UNIREMINGTON®
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

INTRODUCCION AL DESARROLLO DE SOFTWARE

INGENIERÍA DE SISTEMAS

FACULTAD DE CIENCIAS BÁSICAS E INGENIERÍA

Vicerrectoría de Educación a Distancia y virtual

2016

El módulo de estudio de la asignatura Introducción al Desarrollo de Software es propiedad de la Corporación Universitaria Remington. Las imágenes fueron tomadas de diferentes fuentes que se relacionan en los derechos de autor y las citas en la bibliografía. El contenido del módulo está protegido por las leyes de derechos de autor que rigen al país.

Este material tiene fines educativos y no puede usarse con propósitos económicos o comerciales.

AUTOR

Javier Ospina Moreno
Docente- Aspirante a Magister
javier.ospina@uniremington.edu.co

Nota: el autor certificó (de manera verbal o escrita) No haber incurrido en fraude científico, plagio o vicios de autoría; en caso contrario eximió de toda responsabilidad a la Corporación Universitaria Remington, y se declaró como el único responsable.

RESPONSABLES

Jorge Mauricio Sepúlveda Castaño
Decano de la Facultad de Ciencias Básicas e Ingeniería
jsepulveda@uniremington.edu.co

Eduardo Alfredo Castillo Builes
Vicerrector modalidad distancia y virtual
ecastillo@uniremington.edu.co

Francisco Javier Álvarez Gómez
Coordinador CUR-Virtual
falvarez@uniremington.edu.co

GRUPO DE APOYO

Personal de la Unidad CUR-Virtual
EDICIÓN Y MONTAJE
Primera versión. Febrero de 2011.
Segunda versión. Marzo de 2012
Tercera versión. noviembre de 2015

TABLA DE CONTENIDO

Pág.

1 MAPA DE LA ASIGNATURA

INTRODUCCION AL DESARROLLO DE SOFTWARE

PROPÓSITO GENERAL DEL MÓDULO

Como propósito principal del área, se tiene el complemento de temas relacionados con áreas lógicas, lenguajes de programación y de toma de decisión, dando una visión global a todos los puntos principales de la carrera, orientado su propósito, alcances y logros durante los próximos semestres que dure esta.

OBJETIVO GENERAL

Detallar pormenorizadamente el mundo del desarrollo del software identificando conceptos de ingeniería de sistemas contextualizados en el mundo, sus factores diferenciales a nivel Uniremington, el recorrido evolutivo del computador, su aporte a la sociedad, algunos de los sistemas numéricos, de almacenamiento y sus respectivas conversiones al igual que su aplicabilidad en lo lúdico como en la identificación de situaciones lógicas, incursionando en la elaboración de aplicaciones en pro de la metodología orientada a objetos y algunas herramientas del uml como el diagrama de clase y el de uso específicamente.

OBJETIVOS ESPECÍFICOS

- **Identificar los conceptos de Ingeniería de Sistemas**, su contextualización a nivel global, nacional y local, describiendo los Factores diferenciales en la actualidad a nivel de UNIREMINGTON.

- **Explorar las etapas evolutivas** del computador a través de la historia, el aporte a la sociedad en su cotidianidad, los diferentes sistemas numéricos y de almacenamiento incluyendo la conversión al igual que su aplicabilidad tanto en lo lúdico como en la identificación de situaciones lógicas respecto a las generalidades del desarrollo de software.

- **Describir los conceptos y las diferentes estructuras básicas** del lenguaje de programación basados en la metodología Objetual, su instalación, configuración, manejo del entorno y aplicación; explorando algunas herramientas como el diagrama de clase y de uso que permiten una interpretación, diseño y modelado para la elaboración de aplicaciones en pro de una metodología de orientación a Objeto por medio de proyectos aplicándolos adecuadamente el desarrollar software.

UNIDAD 1:

INTRODUCCIÓN A LA INGENIERIA DE SISTEMAS

UNIDAD 2:

GENERALIDADES DEL DESARROLLO DE SOFTWARE

UNIDAD 3:

PRINCIPIOS DE DESARROLLO DE SOFTWARE

2 UNIDAD 1 INTRRODUCCION A LA INGENIERIA DE SISTEMAS

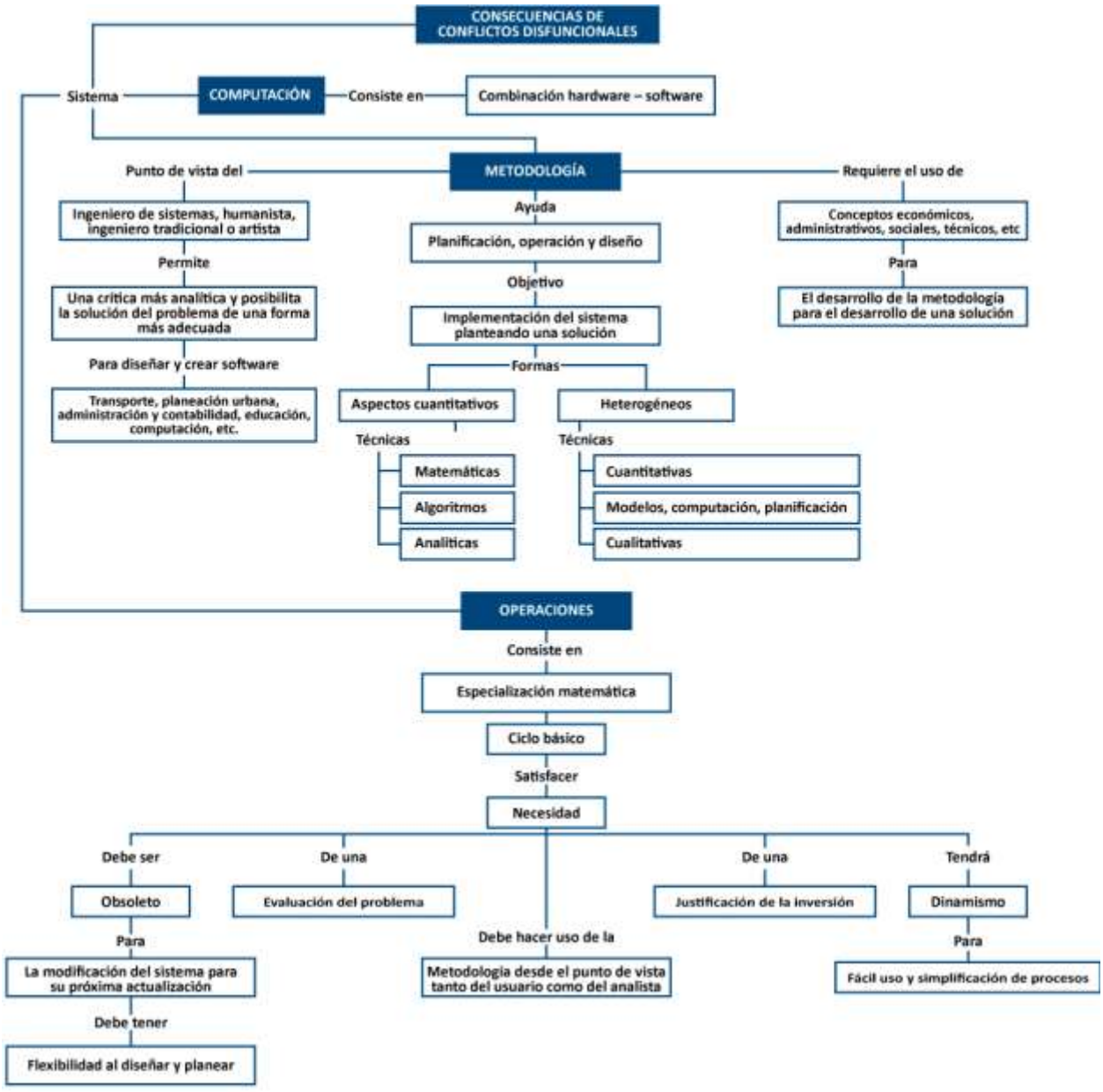
El Proyecto Educativo del Programa es la herramienta fundamental que recoge cada una de las propuestas académicas que se le impartirán al estudiante dentro de su proceso de enseñanza – aprendizaje. La Corporación Universitaria Remington con su Proyecto Educativo Institucional asegura el cumplimiento y puesta en marcha del PEP de las diferentes facultades o escuelas en coherencia absoluta con el PEI. Es así como el programa de Ingeniería de Sistemas tiene la gran responsabilidad de estructurar este proyecto educativo acatando el cumplimiento de las políticas y normativas que tiene fijada la Institución dentro de su marco legal – educativo y con base a las exigencias del Ministerio de Educación Nacional.

El PEP del programa de Ingeniería de Sistemas que propone la Facultad de Ciencias Básicas e Ingeniería está fundamentado en cada uno de los principios legales del programa curricular y del diseño estructural educativo estipulado en el PEI, el cual garantiza la viabilidad de los procesos y actividades ejecutadas durante el proceso de enseñanza – aprendizaje docente – discente, y por supuesto, el reconocimiento de parte de la comunidad educativa perteneciente al área de Ingeniería de Sistemas de las normas, reglamentos y políticas que se deben cumplir para ser verdadero integrantes de la comunidad educativa del programa de Ingeniería de Sistemas de la Corporación Universitaria Remington.

2.1 DEFINICIÓN DE INGENIERÍA DE SISTEMAS Y CONTEXTUALIZACIÓN GLOBAL, NACIONAL Y LOCAL.

La gente relaciona las profesiones con conceptos y son en parte los profesionales quienes dejan la imagen de esos conceptos en la imagen de los demás. Por ejemplo un abogado está asociado a la ley y la gente sabe que los puede usar para defenderse, para demandar, para redactar un contrato, etc. La gente sabe que un chofer “opera” un automotor y que un mecánico automotriz lo arregla. Pero cuando se trata del ingeniero de sistemas a veces parece que todo lo que la gente sabe es que “arreglan computadores”.

En este orden de ideas es menester tener claro el concepto de ingeniero de sistemas y lo que necesitamos proyectar para vender esta profesión como lo que debe ser ante la gente del común, ante nuestros jefes, las empresas y la sociedad en general. Para comprender la definición de ingeniero de sistemas es necesario recordar que el ingeniero debe ser un investigador por excelencia.



Fuente: http://francisco724.blogs.com/mapa_conceptual/2006/02/mapa_conceptual.html

2.1.1 CONTEXTO HISTÓRICO: ¿CÓMO SURGIÓ LA INGENIERÍA DE SISTEMAS?

De acuerdo a (Pico, 2013) la primera referencia que describe ampliamente el procedimiento de la Ingeniería de Sistemas fue publicada en 1950 por Melvin J. Kelly, entonces director de los laboratorios de la Bell Telephone, subsidiaria de investigación y desarrollo de la AT&T. Esta compañía desempeñó un

papel importante en el nacimiento de la Ingeniería de Sistemas por tres razones: la acuciante complejidad que planteaba el desarrollo de redes telefónicas, su tradición de investigación relativamente liberal y su salud financiera. Así, en 1943 se fusionaban los departamentos de Ingeniería de Conmutación e Ingeniería de Transmisión bajo la denominación de Ingeniería de Sistemas. En opinión de Arthur D. Hall, "la función de Ingeniería de Sistemas se había practicado durante muchos años, pero su reconocimiento como entidad organizativa generó mayor interés y recursos en la organización". En 1950 se creaba un primer curso de postgrado sobre el tema en el MIT y sería el propio Hall el primer autor de un tratado completo sobre el tema [Hall, 1962]. Para Hall, la Ingeniería de Sistemas es una tecnología por la que el conocimiento de investigación se traslada a aplicaciones que satisfacen necesidades humanas mediante una secuencia de planes, proyectos y programas de proyectos.

2.1.2 ¿POR QUÉ ES TAN IMPORTANTE LA INGENIERÍA DE SISTEMAS?

> La Cumbre Mundial sobre la Sociedad de la Información declaró en el año 2003 el deseo y el compromiso de todas las naciones por enfrentar uno de los más importantes desafíos a escala global, **la construcción de una sociedad de la información centrada en la persona, integradora y orientada al desarrollo.** Los arquitectos de **esta nueva sociedad de la información deben ser principalmente los Ingenieros de Sistemas,** los cuales tienen ante si grandes retos como:

- Promover las Tecnologías de la Información y las Comunicaciones para el desarrollo de las naciones.
- Construir la infraestructura de la información y las comunicaciones como factor habilitador de la Sociedad de la Información.
- Facilitar el acceso a la información y al conocimiento a todos los hombres y mujeres.
- Crear la capacidad humana para que cada persona tenga la posibilidad de adquirir las competencias y conocimientos para comprender la Sociedad de la Información.
- Fomentar la confianza y seguridad en la utilización de las Tecnologías de las Información y las Comunicaciones.
- Desarrollar un entorno propicio en nivel nacional e internacional en el marco de las Tecnologías de la Información y las Comunicaciones como una herramienta para el buen gobierno.
- Aplicar las Tecnologías de la Información y las Comunicaciones para lograr beneficios en todos los aspectos de la vida de los seres humanos.

> Se puede decir entonces que el futuro de la Sociedad, depende en gran medida de la profesión de la Ingeniería de Sistemas.

2.1.3 EL VALOR SOCIAL DE LA INGENIERÍA DE SISTEMAS

Aunque el enfoque de una ingeniería no va directamente ligada con el área humana y las relaciones sociales, el ingeniero debe establecer una relación social para un adecuado desempeño, en especial el ingeniero de sistemas, pues su objetivo principal es la solución de problemas al usuario. Un ingeniero de sistemas debe poseer la capacidad de interrelacionarse para conocer las necesidades del cliente, sus conocimientos, sus capacidades y posibilidades y resolver el problema de la mejor manera posible.

En la Ingeniería de Sistemas existen una gran cantidad de ares especializas en el campo laboral, cada una de ellas cumple una función específica y requiere de unas cualidades especiales, así como brinda a la comunidad un bien específico. **La ingeniería de sistemas aporta a la sociedad**:

- Facilidades para cumplir una tarea especifica
- Automatización de los procesos
- Garantiza calidad en el sistema
- Disminuye trabajos que requieren de tiempo, conocimientos y espacios especiales.
- Reemplaza trabajos de riesgo
- Permite una mejor administración de la información

> Existe una gran cantidad de ayudas que la ingeniería de sistemas puede aportar a la sociedad dependiendo de la función específica que cumpla:

2.1.3.1 ROBÓTICA:

- Facilita funciones que ponen en peligro la vida del hombre
- Minimiza el esfuerzo físico de cualquier trabajo
- Crea sistemas que facilitan el trabajo de personas discapacitadas 2.

2.1.3.2 SISTEMAS DE INFORMACIÓN:

- Facilita la organización empresarial
- Analiza las necesidades del cliente
- Interrelaciona a los diferentes componentes empresariales

2.1.3.3 INGENIERÍA DE SOFTWARE:

- Facilita trabajos empresariales complejos
- Se interrelaciona directamente con el usuario que requiere el producto de software
- Relaciona la administración desde la parte técnica y comercial de un producto.

2.1.3.4 COMPUTACIÓN GRÁFICA:

- Creación de animaciones con objetivos educativos y de entretenimiento.
- Avance en la realidad virtual como aplicación en las ciencias médicas y arquitectónicas.

2.1.3.5 SISTEMAS DE INFORMACIÓN GEOGRÁFICA:

- Regularización y análisis de desastres naturales
- Adecuados estudio de suelos para construcción de viviendas
- Estudios viales para la ciudad.

2.1.3.6 REDES:

- Facilitan la comunicación a nivel mundial.
- Permite el acceso libre de información
- Minimiza los costos de una empresa.

2.1.4 EL ROL Y DEFINICIÓN DEL INGENIERO DE SISTEMAS EN COLOMBIA

No cabe duda que hay una gran diferencia entre el concepto de Ingeniero de Sistemas en Colombia y en la mayoría de los demás países. En Colombia el ingeniero de sistemas se desempeña principalmente en área de desarrollo de software, de redes y telecomunicaciones y de soporte técnico. Dentro de las organizaciones su rol no es modelar sistemas complejos, de hecho hay más ingenieros industriales que desempeñan ese tipo de funciones. Su rol está relacionado con los sistemas informáticos que la organización a la que pertenece utilice y los que deseen implementar.

La definición del ingeniero de sistemas en Colombia está bastante ligada al computador y a los sistemas de información. Tomemos por ejemplo un fragmento del perfil profesional del ingeniero de sistemas de la universidad de los andes:

> El ingeniero de sistemas colombiano se mueve, sobre todo, en 3 grandes áreas de conocimiento: Construcción de software, Tecnologías de información y Sistemas de información

En otros países el ingeniero de sistemas colombiano podría llamarse ingeniero de software, ingeniero en computación o ingeniero informático. Para bien o para mal se hace necesario apartarnos del concepto oficial de la ingeniería de sistemas para definir el verdadero rol que cumple en nuestra sociedad.

La definición de la ingeniería de computación de acuerdo con Wikipedia es:

La ingeniería en computación estudia el desarrollo de sistemas automatizados y el uso de los lenguajes de programación; de igual forma se enfoca al análisis, diseño y la utilización del hardware y software para lograr la implementación de las más avanzadas aplicaciones industriales y telemáticas.

Conclusión: Una posible definición de ingeniería de sistemas en Colombia

> La ingeniería de sistemas en el contexto colombiano puede definirse como la ingeniería que estudia las tecnologías de la información y se ocupa de su análisis, diseño, desarrollo, operación, implementación y documentación.

2.2 INGENIERÍA DE SISTEMAS EN LA ACTUALIDAD

Hoy en día los ingenieros en sistemas cumplen una función muy importante en el área tecnológica (aunque no solo se desempeñan en esta área, ya que el planeta se ha venido haciendo dependiente de la tecnología y los avances de esta y aquí entra en protagonismo la parte de sistemas, donde hay los ingenieros tienen que hacer que estos funciones de acuerdo a lo que se necesite mediante la programación de estos sistemas y que nos hagan esta tarea más sencilla. No solamente los adaptamos si no que los mantenemos y los mejoramos, para ello debemos tener el conocimiento apropiado de estos sistemas para poder realizar todo esto.

> La sociedad a veces no tiene ninguna idea de los Ingenieros de Sistemas porque creen que solo se trabajan en el área de computadoras cosa que no es así ya que puede cubrir muchos campos de trabajo, como las telecomunicaciones, gerencia, mantenimiento, informática, redes y afines.

El ingeniero de sistemas debe tener cualidades únicas la más única seria pensar con pensamiento e ideales sistémicos y no con el método científico, pero también debe ser innovador, creativo, diseñador, entre otras.

En conclusión el Ingeniero de Sistemas a lo largo de los años remontándonos desde sus inicios siendo una carrera relativamente nueva se ha venido haciendo importante por lo mucho que nos hemos venido independizándonos de la tecnología y los sistemas que cumplan y satisfagan nuestras exigencias.

Figura 2 Estructura de la Ingeniería de Sistemas

2.2.1 ACTUALIDAD

El mundo está en constantes cambios, para lo cual toda sociedad debe estar en la capacidad de afrontar los nuevos retos que conlleve, de lo contrario entrará en un caos sin precedentes, tal como ocurrió recientemente, con el denominado efecto dominó, que inicio en EEUU y luego se expandió como una ola gigantesca, por todas las economías, de los países del hemisferio.

Ahora bien, **uno de los pilares fundamentales de toda economía, está orientada desde el papel que juega el Ingeniero de Sistemas, desde una óptica enfocada al conocimiento e interacción de las diversas tecnologías aplicadas al desarrollo de las potencias y de los países**, con un alto nivel profesional, en el manejo de un ente económico como lo es la empresa o la entidad o persona particular que se esté asesorando, con el fin de que el barco “empresa” o “negocio”, navegue en la dirección correcta, aplicado al mundo de los sistemas de información.

> Para ello, se hace necesario realizarnos los siguientes cuestionamientos, aplicando a su vez un contraste, basado con ejemplos de la realidad del quehacer cotidiano del Ingeniero de Sistemas y su entorno en la actualidad: ¿No hay ingenieros de sistemas? ¿Colombia está importando ingenieros de la India? ¿Los disponibles son pocos, no saben inglés? ¿No tienen la formación suficiente? ¿Es esto cierto?, ¿Cuál es su punto de vista? ¿Qué crees que pueden hacer las universidades?, ¿Cuáles son los problemas más relevantes de estudiar la ingeniería en Colombia? Y ¿Realmente la ingeniería de Sistemas está impulsado el desarrollo tecnológico de la sociedad?

En primera instancia, Colombia al igual que algunos países del hemisferio se están preparando ante los constantes avances tecnológicos y donde el principal punto de apoyo es el conocimiento y aplicabilidad del mismo en los diversos campos del saber, a través de los sistemas de información. Sin embargo, estos avances son lentos, ya el acceso a la educación superior está limitado por diversos factores socioeconómicos, tales como falta de apoyo financiero de los educandos, falta de compromiso por parte de las personas en edad escolar u/o formación superior. Y para colmo, el gobierno, a través de los medios de información ha lanzado una campaña para incentivar a los jóvenes a realizar estudios técnicos, en vez de apoyarlos con mayores recursos para el acceso a la educación superior.

De igual forma, se puede visualizar que los pocos que logran ingresar a la educación superior optan por programas diferentes a las ingenierías y más a la de sistemas, presentando al mismo tiempo falta de compromiso en su formación ético-profesional por parte de algunos de ellos. Tal es el caso, que han tenido que traer ingenieros de países como la India “Para el gerente del Centro de Desarrollo de Software de Open Systems, Benito Pardini, la crisis por falta de gente preparada en Colombia es tan notorio, que ya han tenido que importar algunas veces personas de países como India, con el fin de suplir sus necesidades”. Y para anexar al problema algunos expertos en el tema afirman que: **“Hoy Colombia enfrenta un gran déficit de ingenieros de sistemas. No solo hay problemas de cantidad de profesionales, sino que muchos de ellos trabajan para compañías en el exterior y que otro gran grupo no tiene las calificaciones que se requieren”.**

Pero que piensan algunos expertos en el tema: “Algunos en el país, como la Asociación Colombiana de Ingenieros de Sistemas, no han visto el problema. En diálogo con Dinero.com, uno de sus directivos manifestó que encuentran que la situación en el mercado es normal”. Será cierto esta afirmación, nos queda una inquietud tal como se puede vislumbrar en la siguiente afirmación: “Así mismo, hay dificultades con la actualización de los profesionales y del enfoque de los programas académicos. Con la velocidad que se mueve la tecnología, **un profesional que deje de actualizarse en dos años pierde vigencia.**

El otro punto del problema, que también resulta definitivo, es que los pocos jóvenes mejor preparados están encontrando formas fáciles de engancharse con empresas extranjeras que les ofrecen una mejor remuneración que las nacionales”. Ante este panorama, es importante anotar que el gobierno

colombiano, debe implementar políticas educativas más potentes que permitan a los jóvenes tener una mejor calidad de vida, y así estos no tengan que orientar su perfil profesional hacia otros rumbos o por el contrario que estos tengan que conformarse con la culminación de los estudios básicos, media académica o una técnica, dejando así, las carreras profesionales en manos o empresas extranjeras.

De allí que el gobierno Colombiano, deba apoyar a las instituciones superiores, es decir las universidades, y que estas asuman, junto con los futuros profesionales el verdadero papel que deban asumir, para que de esta forma, el país salga de ese letargo o estancamiento competitivo en que se encuentra y así estar a la vanguardia ante el mundo y no estar en desigualdad ante otros países, tal como lo podemos analizar con el TLC ante EEUU y Canadá.

Ahora bien, universidades como la Universidad Simón Bolívar, se están preparando al respecto "Actualmente, la Educación Superior concibe y potencia la función de EXTENSIÓN como una de las principales vertientes de trabajo de la Universidad, apreciándosele como el elemento más dinámico e integrador del vínculo UNIVERSIDAD-SOCIEDAD.

Se pretende participar activamente en los procesos permanentes de interacción e integración con las comunidades locales, en aras de garantizar la presencia de la Universidad en la vida social y aportar al desarrollo de las comunidades, especialmente aquellas que en la ciudad, el departamento y la región se consideran en desventaja socioeconómica y/o más vulnerables; Por medio de la extensión, la Institución entra en contacto con el medio externo en donde se pretende tener un impacto positivo al generar proyectos que lleven implícito el mejoramiento de la calidad de vida de la comunidad afectada.

Consecuentemente el programa de Ingeniería Sistemas ha venido realizando para el presente semestre diversas actividades de proyección social contempladas principalmente dentro de un proyecto de Extensión Dirigido a Colegios localizados en zonas vulnerables, específicamente en el Instituto Técnico "Padre Álvaro Gutiérrez de Paz y Futuro" ubicado en el Barrio Simón Bolívar, donde se realizó una brigada de mantenimiento tanto preventivo como correctivo de computadores. Con la participación de un grupo de 12 estudiantes de la asignatura de Soporte Tecnológico, Dirigidos por su Docente el Ingeniero Johel Rodríguez y el Coordinador de Proyección social del Programa, el Ingeniero Raúl Rodríguez".

> En última instancia, es compromiso de todos y cada uno de los actores sociales, económico y políticos implementar verdaderas políticas educativas que permitan estar en igualdad de condiciones a Colombia con los demás países del hemisferio, así como ayudar a implementar con mayor rigor la carrera profesional de Ingeniería de Sistemas, para así no tener que importar profesionales de otros países y evitar de esta forma el monopolio del conocimiento por parte de las grandes multinacionales (Johaspot, 2010).

2.2.2 ACTUALIDAD DE LA INGENIERA EN COLOMBIA

La carencia de apoyo a las ciencias, se vio reflejado en los primeros intentos por hacer universidad, pues 1968 se creó COLCIENCIAS, como una entidad para formar el desarrollo científico en el país, lo que demuestra que no ha realizado un esfuerzo coherente y sostenido para crear una infraestructura científica y tecnológica.

La Ingeniería Colombiana es escasa: no obstante ha hecho aportes significativos al país; tales eventos se pueden describir en diversos niveles: A nivel de formación están las actividades ingenieriles por la jerarquía y la creatividad. En el nivel uno está la investigación tecnológica científica, sobre nuevos procedimientos del cálculo. En el nivel dos la creación de nuevos trabajos de proyectos y obras de ingeniería. En el nivel tres obras proyectadas de ingenieros del nivel dos o del mantenimiento de industrias establecidas. En el nivel cuatro la realización de tareas de ensayos, mediciones, control, ejecutados por ingenieros investigadores de alta formación.

Según el ICFES el porcentaje de título de ingeniería en 1976 era casi del 0% y las maestrías no han pasado del 1%

A nivel de ambiente de trabajo para la ingeniería colombiana, se ve un clima altamente inseguro por la obsolescencia de las empresas, la recesión. Continuamente asesinan y secuestran ingenieros por lo tanto las obras civiles, las telecomunicaciones, la distribución eléctrica, la ingeniería y la minería han sufrido grandes atrasos.

2.2.3 INGENIERÍA COLOMBIANA Y MUNDIAL

Dentro de las ventajas hay coincidencia en que existe un mejor conocimiento del medio geográfico y cultural y la exigencia de salarios de menos costos. **Y dentro de las desventajas hay cinco aspectos que resaltan como son: La debilidad del país en ciencia y tecnología e investigación, para tecnología de punta, las limitaciones financieras por el escaso acceso a créditos. La ingeniería extranjera que tiene alianzas nacionales con los gobiernos de origen. En gestión tecnológica las empresas presentan grandes debilidades de adaptación e innovación; los sistemas de comunicación e información precarios y el no manejo de un idioma extranjero y la poca estructura tecnológica, la privatización de empresas estatales y la debilidad gremial.**

Es el análisis y definición de los puestos de trabajo de la organización, lo que incluye a los colaboradores a cargo de ellos, por lo tanto también se analiza el perfil profesional de estos trabajadores.

2.2.4 NUEVAS FORMAS DE BUSCAR CALIDAD DE LA INGENIERÍA COLOMBIANA

> La implementación del modelo económico neoliberal y la globalización de los mercados, ha tenido efectos significativos en la vida empresarial colombiana y exige nuevos retos a las organizaciones dedicadas a proyectos de ingeniería. Por lo tanto se requiere la formulación de nuevos esquemas de financiación y comercialización así como la capacidad ingenieril.

Con relación a la Universidad se reclama la maestría y doctorados en la planta docente que eleven la formación académica.

En gestión tecnológica una mayor capacidad, diseñando alianzas con firmas extranjeras de esta manera progresando en la internacionalización. Se requiere al igual de exponer de banco de datos, comunicaciones, e información inteligente (Unad).

2.3 FACTORES DIFERENCIALES DE LA INGENIERÍA DE SISTEMAS UNIREMINGTON

2.3.1 MISIÓN DEL PROGRAMA

El programa de Ingeniería de Sistemas de la Corporación Universitaria Remington tiene como misión formar profesionales altamente competentes en las áreas de Ingeniería en sistemas computacionales con alto nivel académico y tecnológico, promoviendo el liderazgo en investigación, con el fin de que el egresado pueda desempeñarse en forma eficiente, eficaz y efectiva, en las organizaciones donde preste sus servicios, identificando e interpretando problemas y planteando soluciones óptimas en los diferentes campos empresariales, con el objetivo de aportar al crecimiento social dentro del marco de la ética y los valores institucionales.

2.3.2 PRINCIPIOS Y VALORES

- Responsabilidad y compromiso profesional.
- Ética profesional
- Respeto y tolerancia
- Lealtad
- Superación y liderazgo
- Honestidad y transparencia al ejercer la profesión
- Objetividad en la toma de decisiones

- Optimismo al emprender los retos
- Seguridad y privacidad en el manejo de la información.

2.3.3 JUSTIFICACIÓN DEL PROGRAMA

> La dinámica de la informática y los sistemas a nivel mundial es asombrosa. La causa principal del aumento continuo de la productividad de la economía de Estados Unidos se fundamenta en el uso adecuado y racional de la informática, las comunicaciones y la tecnología. Este avance permanente de innovación y uso acertado de la tecnología es y seguirá siendo el motor del desarrollo. Los continuos lanzamientos de nuevas tecnologías, las mejoras frecuentes a las metodologías exigen una gerencia alerta y activa que defina estrategias adecuadas de cómo integrar estos cambios a la sociedad.

Un ejemplo claro de la necesidad de profesionales en tecnologías de la información es el informe dado por Cisco System en Enero 2008, El informe dice: "De no darse un cambio de ahora, la ausencia de profesionales especializados en soluciones de redes y conectividad será de 126.000 personas o, dicho de otra forma, existirá un déficit de un 27% de este tipo de técnicos en Latinoamérica. Según el informe elaborado por la consultora IDC Latino América para Cisco, el 2007 ya presentó una carencia de unos 84.000 profesionales de red en la región.

El estudio demuestra que en los dos países con mayor crecimiento económico en la región, Brasil y México, el faltante de profesionales irá en aumento conforme crecen sus mercados. Los otros países que incluyen el estudio, Argentina, Colombia, Chile, Costa Rica y Venezuela, reflejan la misma situación. (Ver **¡Error! No se encuentra el origen de la referencia.**)

Tabla Estudio sobre la necesidad de profesionales de Tecnologías de la Información en América Latina

País	2007	2010
Argentina, Colombia, Chile, Costa Rica y Venezuela	32%	33,5%
Brasil	27%	29%
México	21%	24%

FUENTE: Cisco System, enero de 2008.

Este informe se hizo público en Argentina y Costa Rica, países en los cuales se contó con la presencia de representantes de empresas, universidades, prensa y gobierno, entre otras instituciones".

Sin lugar a dudas estamos atravesando por un momento de éxito, para una profesión como la Ingeniería de Sistemas que venía de un tiempo de rezago. El panorama en Colombia no es diferente, ya que como lo menciona Diego Molano Vega, Ministro de las TICS: **"Necesitamos aplicaciones hechas por colombianos para colombianos, que presten servicios que incentiven el uso de las TIC y que a su vez ayuden a generar empleo y disminuir la pobreza",** a propósito del lanzamiento de la convocatoria de Talento Digital, la cual busca fomentar la formación de capital humano especializado en el uso de tecnologías de la información, el fortalecimiento de Gobierno en línea y el desarrollo de la competitividad, la investigación, la innovación y la proyección internacional. Por otro, esta tendencia a mejorar se relaciona con que en el 2011 el sector de las tecnologías de la información y las comunicaciones, tuvo un crecimiento que supera en cerca de dos puntos al de la economía nacional, convirtiendo a este sector en uno de los que más progreso tuvo el año anterior.1

> La necesidad es urgente, más cuando pensamos en cómo hacer tecnología y quiénes son los profesionales más idóneos para construir un ecosistema digital lleno de campos de investigación y de mundos por descubrir. La preocupación entonces se centra en que no se están formando suficientes graduados en Ciencias de Computación para satisfacer la creciente demanda mundial, como lo menciona Michael Buryk, Gerente de Desarrollo de Negocios en el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE).

Un estudio que se titula "Los profesionales colombianos en el Siglo XXI" realizado por el Observatorio del Mercado de Trabajo y la Seguridad Social de la Universidad Externado de Colombia, menciona que dentro de las 10 carreras más rentables que tiene el país en estos momentos se encuentra la Ingeniería de Sistemas2, según la demanda que hacen las compañías por profesionales en esta área del conocimiento. Pero más allá de la rentabilidad de la carrera, lo importante es comprender el impacto y la importancia de formar profesionales en ingeniería de sistema que quieran asumir los retos y desafíos que exige la era de la información en temas como salud, entretenimiento, banca, entre otros.

En uno de los comentarios que se puede encontrar en el artículo The Nation's Best Jobs In Engineering & Information Technology de la IEEE Job Site se lee "***Los ingenieros de software son las estrellas de rock del mundo laboral de hoy, y los analistas de sistemas, incluso informáticos y los desarrolladores web pueden reclamar parte de ese reconocimiento, ya que la demanda de los profesionales de TI es tan profunda.***"3

[1] http://www.vanguardia.com/actualidad/colombia/162161-el-sector-tic-crecio-mas-que-la-economia-nacional-mintic
[2] http://noticias.universia.net.co/en-portada/noticia/2012/06/19/944104/carreras-mas-rentables.html
[3] http://careers.ieee.org/article/bestjobs_0612.php

> A nivel curricular en el mundo IEEE y ACM definen la currícula para Ingeniería de Sistemas, uno de los trabajos es la curricula computing de 2013 donde se definen dieciocho áreas temáticas fundamentales en Ciencias de la Computación, que representan el cuerpo de conocimiento de esta disciplina a nivel de pregrado. Cada una de las áreas se divide en unidades temáticas y representa un campo particular de la disciplina. Internacionalmente, este modelo es la base para la construcción de currículos relacionados con ciencias de la computación, o para llevar a cabo procesos de acreditación internacional.

Estas temáticas son:

1. Algoritmos y Complejidad
2. Arquitectura y Organización
3. Ciencias de la Computación
4. Estructuras Discretas
5. Computación Gráfica y Visual
6. Interacción Humano-Computador
7. Seguridad y Aseguramiento de la Información
8. Gestión de la Información
9. Sistemas Inteligentes
10. Redes y Comunicaciones
11. Sistemas Operativos
12. Plataformas de Desarrollo
13. Computación Paralela y Distribuida
14. Lenguajes de Programación
15. Fundamentos de Desarrollo de Software
16. Ingeniería de Software
17. Fundamentos de Sistemas
18. Temas Sociales y Profesionales

Dando continuidad a estos trabajos las entidades anteriormente descritas en Computer Science Curriculum 2008 definen objetos de aprendizaje, Naturaleza de objetivos de aprendizaje, características de graduados, consideraciones internacionales, entre otros elementos de programas de Informática, sistemas y afines.4

[4] Computer Science Curriculum 2008: An Interim Revision of CS 2001. Report from the Interim Review Task Force includes update of the CS2001 body of knowledge plus commentary December 2008. ACM-IEEE.

Otro trabajo importante es IS 2010 realizado por Association for Computing Machinery (ACM) Association for Information Systems (AIS), que fundamenta sus conclusiones en los grandes cambios en la tecnología y prácticas en la industria, que incluyen procesos de desarrollo de sistemas de información, introducción de la tecnologías web, nuevas arquitecturas emergentes, computación móvil y ubicua, además de buenas prácticas en TI como ITIL y COBIT.5

A nivel Nacional por la concentración de los conflictos sociales en las áreas rurales, Colombia ha venido desarrollándose alrededor de las grandes ciudades. Esta situación ha ocasionado que la educación superior en el país se caracterice como un fenómeno más de condiciones urbanas que rural. Por ello no es de extrañar que al respecto se materialicen grandes desequilibrios entorno a la educación mirado desde el punto de vista de crecimiento de la población y la distribución de la misma tanto a nivel regional, como en general del país. Por esta razón cuando se analiza la distribución de las matriculas en educación superior, se observa que la concentración gira alrededor de las grandes urbes. En este sentido es importante resaltar que, de acuerdo con esta caracterización y proyección, en el inmediato futuro, la evolución de la educación superior estará ligada a la dinámica del crecimiento de las mismas ciudades.

Colombia en las dos últimas décadas, progresivamente ha venido insertándose en el concierto mundial, mediante la formalización de diversos acuerdos económicos con otras naciones. A pesar que ellos giran en torno a los intercambios comerciales, no es de dejar pasar desapercibido que existen múltiples variables relacionadas con el manejo y la interacción que la transferencia del conocimiento tiene, y las particularidades que puede presentar la prestación de servicios, entre ellos los relacionados con la educación, y dentro de estos, como es el manejo que se puede dar a los títulos y certificaciones que de la prestación de los servicios se derive.

Asociado a los intercambios comerciales, fuera de la captación de capitales externos en inversiones productivas y el creciente intercambio comercial de bienes y servicios, se producen otros efectos como lo es el relacionado con la transferencia de conocimiento tecnológico asociado a la producción de bienes y servicios y el componente implícito de mayor productividad y competitividad de las plantas de producción, lo que se traduce en un conocimiento aplicado de nuevas tecnologías.

[5] Curriculum Guidelines for Undergraduate Degree Programs in Information Systems Association

> La misma dinámica de la situación ha generado que estén apareciendo nuevos fenómenos como pueden ser el avance acelerado de los conocimientos científicos, humanísticos y tecnológicos, la creciente participación del sector privado en la educación en todos los niveles, la cada vez mayor escolaridad de la población urbana y rural, en los niveles de la educación básica y sobre todo, en los avances en las tecnologías de la información y la comunicación.

Los sustentos desde el gobierno para los programas de Ingeniería de Sistemas son el plan Nacional de Tecnologías de la Información y la Comunicaciones 2008-2019 busca que, al final de este período todos los colombianos se informen y se comuniquen haciendo uso eficiente y productivo de las TIC, para mejorar la inclusión social y aumentar la competitividad.6

A nivel curricular de acuerdo con el estudio de ACIS, "Caracterización de la Ingeniería de Sistemas en Colombia", los programas de ingeniería de sistemas se distribuyen entre los siguientes perfiles:

- **Ciencias de la computación, 17%**
- **Ingeniería de Software, 39%**
- **Sistemas de Información, 12%**
- **Tecnología de la información, 5%**
- **Otro, 27%**

ACOFI también define lo que buscan los programas de ingeniería de sistemas y lo que la gente usualmente entiende por Ingeniería de Sistemas en Colombia: "podemos decir que ésta se refiere a los aspectos humanos y organizacionales y a la tecnología relacionados con la planeación, el análisis, el modelado, la captura, la transmisión, la presentación y la seguridad de la información, en cuanto que éste es un recurso estratégico de las organizaciones; esto implica elementos importantes de modelado y diseño de sistemas. **Un Ingeniero de Sistemas debe tener capacidad para: diagnosticar, diseñar, construir, evaluar, auditar y mantener sistemas y procesos de información dentro de un marco administrativo, empresarial y humanista."**

A nivel regional la población objetivo de la Corporación Universitaria Remington es aquella población de estratos 2,3 y 4 que sin recursos económicos para acceder a universidades privadas de alto costo, y debido

[6] Plan Nacional de Tecnologías de la Información y las comunicaciones. Marzo 2008.

a su ubicación geográfica socioeconómica la calidad de su educación secundaria no le permite alcanzar el nivel exigido en los exámenes de admisión para lograr su ingreso a las universidades públicas.

Es a este nivel regional donde se percibe una necesidad de ofrecer programas que permitan comprender, asimilar y adecuar a nuestra realidad los avances de la tecnología.

> Las propuestas del Plan Estratégico de Antioquia y el Plan de Desarrollo, Visión 2019 han establecido una serie de metas que comprometen el sector de la educación superior: Cobertura, Calidad, Ciencia y Tecnología y las propias de la educación Básica y Media, requieren que las regiones de Antioquia y Colombia dispongan de profesionales de informática y sistemas preparados para mejorar nuestra competitividad y nuestro progreso. Colombia, requiere hoy mejorar su competitividad para incrementar su nivel de vida. El enfoque de sistemas, los sistemas de información, los adelantos en comunicación son herramientas imprescindibles para acelerar el ingreso a los mercados globales.

En el nivel superior se presenta la diferenciación entre las universidades clásicas y las instituciones de formación profesional y tecnológica, y sus respectivos programas de formación.7

En diferentes momentos en la historia educativa colombiana han sido propuestas instituciones y modelos curriculares claramente orientados a la formación de una cultura técnica como sustento de políticas de desarrollo y modernización del sector productivo, como estrategia de mayor empleabilidad de los jóvenes y como alternativa a la educación académica general.8

Pero estas iniciativas han sido posteriormente canceladas y reemplazadas por otras políticas de predominio de la educación general, académica, cuya principal función es promover el acceso a la educación superior al pequeño porcentaje del grupo de edad que puede acceder a dicho nivel.

A nivel de proyectos de región y ciudad que sustentan la pertinencia de programas de Ingeniería de Sistemas son:

Plan Activa 2011-2021: La Secretaría de Productividad y Competitividad de la Gobernación de Antioquia, lideró en alianza con el Departamento Administrativo de Ciencia, Tecnología e Innovación -COLCIENCIAS, y el apoyo metodológico de la Universidad de Antioquia, la construcción participativa Plan Departamental

[7] GOMEZ Victor Manuel, "Modalidades de Educación Secundaria y Formación de Actitudes y Disposiciones frente al Conocimiento, en Colombia" C. Depto. de Sociología. Universidad Nacional de Colombia. Mayo 2004.
[8] Los Institutos Técnicos Industriales, creados a finales de la década de los 40s, son la institución más característica de estos propósitos.

de CTI a través de la Encuesta de Percepción de Áreas de Conocimiento, el cual será ratificado por la Asamblea departamental como hoja de ruta para Antioquia en los aspectos de CTI.9

Plan de Ciencia, Tecnología e Innovación 2011-2021: La Alcaldía de Medellín a través de Ruta N y en alianza con Colciencias, construyó de manera participativa el Plan de CTI de Medellín 2011 – 2021, el cual propone una cartera de proyectos que permiten cerrar las brechas tecnológicas, de innovación y competitivas que tienen las cadenas productivas (Salud, energía y TIC). Este Plan será ratificado por el Concejo Municipal como hoja de ruta para la Ciudad en los aspectos de CTI.10

Clúster de las tecnologías de la información y comunicación (Ver): El Clúster Tecnología, Información y Comunicación –TIC-, se define como la concentración geográfica en Medellín y Antioquia, de empresas e instituciones especializadas y complementarias en las actividades de: Consultoría TIC, contact center, contenidos digitales, data center, desarrollo de infraestructura, desarrollo y comercialización de software, producción y distribución de hardware, electrónica y servicios de telecomunicaciones; las cuales interactúan entre sí, creando un clima de negocios en el que todos pueden mejorar su desempeño, rentabilidad y competitividad empresarial.11

> La Red TI (Tecnologías de la Información) Colombia, integrada por los seis clústeres ubicados en las regiones del Triángulo Cafetero, Costa Caribe, Medellín, Santander, Bogotá y Pacífico, cuenta con la participación de 208 empresas de software, en las que se generan 12 mil empleos y representan 1.3 billones de pesos en ventas anuales. Más de $1.600 millones invertirá este año el Ministerio de Tecnologías de la Información y las Comunicaciones, MinTIC, para el fortalecimiento de clústeres de la industria del software en el país.

Entre enero y mayo de este año -2013-, Proexport ha apoyado y asesorado 14 nuevos proyectos de inversión extranjera que llegarán a Colombia a los sectores de BPO, software y servicios de TI y que se instalarán en Pereira, Manizales, Medellín, Barranquilla y Bogotá.

Así lo manifestó la presidenta de la entidad, María Claudia Lacouture, quien señaló que en los primeros cinco meses del año, siete empresas extranjeras del sector de software y servicios de tecnología reportaron que realizarán inversiones en Colombia por US$63.188.487 y generarán la contratación directa 239 personas.

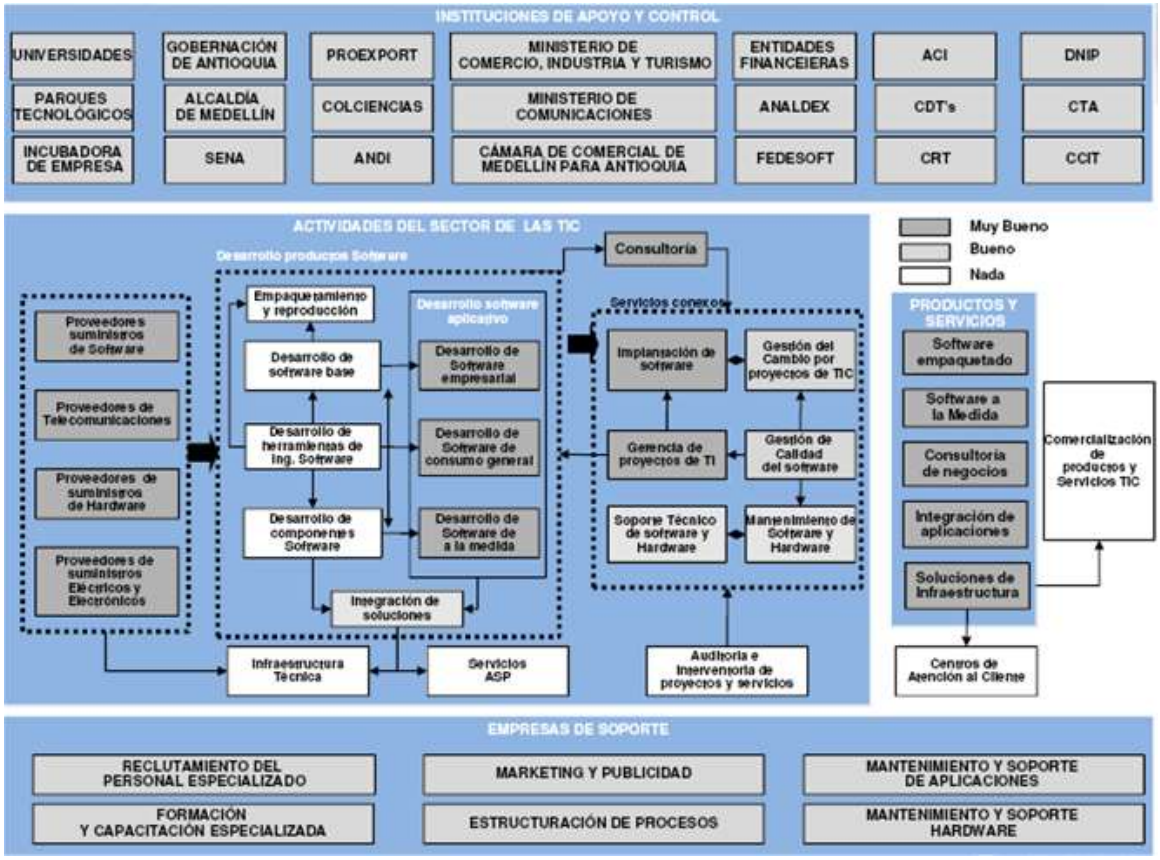
[9] Aspectos básicos para la presentación de propuestas al macro proyecto. “desarrollo de conocimientos y nuevos negocios en tic”

[10] Proyecto corporación ruta N Medellín. Plan estratégico de ciencia, tecnología e innovación de Medellín 2011-2021. http://www.rutanmedellin.org/plancti/Documentos%20compartidos/Plan-de-CTi-de-Medellin.pdf

[11] http://www.camaramedellin.com.co/site/Cluster-y-Competitividad/Comunidad-Cluster/Cluster-TIC.aspx

Por otro lado, cuatro inversionistas del sector de BPO de Estados Unidos, uno de India, España y Reino Unido, respectivamente, que han sido apoyados y asesorados por Proexport realizarán inversiones por US$37.500.000. Para estos siete proyectos se contratarán 4.795 personas en Medellín, Barranquilla, y Bogotá.

Gráfico Clúster Tecnología, Información y Comunicación - FUENTE: Cámara de Comercio de Medellín, julio 30 de 2013



Cluster TIC. Fuente Cámara de Comercio de Medellín

En 2012, la inversión extranjera en Colombia de sector de BPO, que contó con el acompañamiento de la entidad, llegó a los US$16.522.433, representados en cuatro proyectos, los cuales se espera generen la contratación de 3.562 personas.12

[12] http://www.proexport.com.co/noticias/bpo-software-y-servicios-de-ti-sobresalen-en-inversion-en-colombia

2.3.4 COORDINACIÓN DE RECURSOS

De acuerdo al Estatuto General de la UdeG[13] y Reglamento Interno de la Administración General de la UdeG, la Coordinación o Gestión de Recursos Humanos, es el área encargada de diseñar, planear, coordinar, supervisar y evaluar el ingreso, promoción, permanencia, capacitación y desarrollo del personal colaborador ; así como implementar, regular y administrar servicios, prestaciones y programas especiales en beneficio de ellos; con el fin de impulsar y fortalecerla cultura y calidad en sus trabajadores.

2.3.5 RELACIÓN DEL PROGRAMA CON EL MODELO PEDAGOGICO

La Corporación Universitaria Remington, es una institución de educación superior de carácter privado. Presta sus servicios a nivel regional y nacional a través del Proceso Integrado de Formación Profesional por Ciclos "PIF", en las modalidades presencial, y a distancia, en ambientes virtuales, orientada por principios de interacción académica institucional con el medio y de generación de nuevos conocimientos y competencias para el desarrollo.

Cumple sus funciones de docencia, investigación, extensión y relaciones internacionales, para la formación integral de la persona plena en valores éticos, morales, políticos, económicos, ambientales, culturales y transcendentes, a través de sus programas académicos de pregrado y postgrado, en un contexto de alto nivel de excelencia científica, tecnológica y de investigación aplicada orientada al sector productivo, interactuando comprometidamente con la comunidad.

Busca de manera permanente ampliar el acceso y la participación en la educación, con base en la capacidad intelectual, centrada en la responsabilidad, la sinceridad y la autogestión

2.3.6 RELACIÓN DEL PROGRAMA CON EL MODELO CURRICULAR

El programa Ingeniería de Sistemas de la Corporación Universitaria Remington toma el modelo curricular institucional brindando el camino que debe seguir la construcción del programa con respecto a la formulación del macrocurrículos y los microcurrículos con los respectivos créditos, y sobre todo la decisión sobre la metodología pedagógica que se utiliza para la construcción de conocimiento, habilidades y valores que representen una formación profesional de calidad.

Los medios didácticos de docencia tradicional y tutorías presenciales tradicionales se mantienen, sin embargo la Corporación Universitaria Remington ha desarrollado una plataforma (uso de Tecnologías de la Información y la comunicación), que permite el contacto directo de los estudiantes con los currículos, la administración y el seguimiento docente. Además de la construcción de módulos que le ofrecen el direccionamiento de estudio al individuo, estando en cualquier ligar del país.

El programa ha tenido varias versiones curriculares, actualmente opera una versión desde principios de 2011 que realiza cambios en la organización de las asignaturas del área humanística y del direccionamiento a líneas de profundización que ofrezcan las posibilidad a los estudiantes de los dos

[13] UdeG: Universidad de Guatemala.

últimos semestres opciones frente a la dinámica del área de formación y a los constantes cambios tecnológicos.

Las líneas de profundización son:

LINEAS DE ENFASIS
INGENIERÍA DE SOFTWARE
INFORMÁTICA

2.3.7 FUNDAMENTACIÓN TEÓRICA DEL PROGRAMA

La docencia se concibe como la función misional que aborda el proceso formativo en torno al conocimiento teórico-práctico que demanda el objeto de formación profesional, mediado por la enseñanza para el logro de un aprendizaje significativo, a la luz del Modelo Pedagógico Institucional.

En los siguientes cuadros se detalla los principales criterios a tener en cuenta en la fundamentación teórica del programa.

ASPECTOS ACADÉMICOS	CRITERIOS
Currículo	▰ Concepción. El currículo se concibe como el diseño del proceso formativo que se desarrolla en espacios formales a través del plan de estudios y en espacios de formación complementaria ▰ Calidad del Currículo. La calidad del currículo está garantizada por su pertinencia social y su pertinencia académica. La primera está determinada por los requerimientos de desarrollo del entorno económico y social y, la segunda, está determinada por la coherencia entre las competencias académicas y los requerimientos del objeto de formación del programa ▰ Indicadores de Pertinencia Social. Son indicadores de pertinencia social: o El campo de intervención, determinado por el objeto de conocimiento que le da identidad al programa

ASPECTOS ACADÉMICOS	CRITERIOS
	○ El contexto determinado por el escenario donde adquiere significación el objeto de conocimiento ○ El objeto de formación está determinado por las perspectivas desde las cuales se aborda el objeto de conocimiento, en el contexto que le da significado. Este objeto de formación está expresado en las competencias profesionales del egresado ○ Las competencias de desempeño profesional, determinadas por las habilidades de desempeño para el ejercicio del Contador Público en cada una de las competencias profesionales • Indicadores de Pertinencia Académica. Definidos desde lo que debe saber el egresado para el logro de un desempeño profesional de calidad: ○ Los campos del saber que deben fundamentar el objeto de formación ○ Las competencias académicas, expresadas en lo que debe saber de cada área de conocimiento, esto es, la profundidad con la que se debe abordar cada una de las áreas, dados los requerimientos del objeto de formación ○ Plan de estudios, estructurado con fundamento en las áreas de formación y créditos académicos, determinados por la normatividad vigente • Macrodiseño Curricular. Los indicadores de pertinencia social y de pertinencia académica constituyen el diseño macro del currículo y direccionan la intencionalidad de su implementación en el aula, (metodología presencial) o en el módulo (metodología a distancia), en el trabajo independiente del estudiante y en los espacios de formación complementaria. • Microdiseño Curricular. Expresado en el diseño de las asignaturas, con sujeción al macrodiseño curricular y estructurado en torno a estrategias pedagógicas que, en el marco de la investigación formativa, garanticen el acercamiento adecuado del estudiante con el conocimiento y que desarrollen competencias de conocimiento y de pensamiento crítico. • Integralidad, interdisciplinariedad y flexibilidad del currículo. La integralidad, expresada en un plan de estudios fundamentado en lo básico y lo específico de la profesión, en lo complementario desde el punto de vista ético, estético, económico y social para el logro de una formación integral del estudiante, en coherencia con la misión institucional y del programa, así como con los objetivos del mismo. La interdisciplinariedad expresada en actividades curriculares que tienen un carácter explícitamente interdisciplinario para abordar con rigor el campo de

ASPECTOS ACADÉMICOS	CRITERIOS
	intervención propio del objeto de formación. La flexibilidad expresada en opciones académicas, pedagógicas y curriculares para la consolidación del objeto de formación
Competencias	▰ Concepto. Se entiende por competencia un saber-hacer desde el conocimiento en un contexto socio cultural determinado, para resolver problemas reales a través de la elaboración de productos o servicios, tangibles o intangibles, significativos para su entorno. En el campo de la formación integral, la educación por competencias desarrolla herramientas conceptuales, aptitudinales, actitudinales y valores, que le permiten al individuo desempeñarse en los diversos contextos de la cotidianidad ▰ Clasificación. El currículo del programa está desarrollado con base en las siguientes competencias: ○ Competencias profesionales que estructuran el objeto de formación del programa ○ Competencias de desempeño profesional que se expresan en el perfil del egresado ○ Competencias académicas que expresan la complejidad con la que deben ser abordados los campos del saber que fundamentan el objeto de formación ○ Competencias de conocimiento desarrolladas a través de los contenidos relevantes de cada campo de saber ○ Competencias de pensamiento desarrollas a través de estrategias pedagógicas que, en el marco de la investigación formativa, estructuran métodos de razonamiento en torno a los contenidos de los diferentes campos de saber propios del objeto de formación ▰ Formulación. Toda competencia debe formularse con los siguientes referentes: un verbo de desempeño, un objeto, una finalidad y una condición de calidad.
Evaluación	▰ Una formación por competencias requiere de una evaluación por competencias. Toda competencia tiene como referente dos clases de contexto con los cuales debe ser confrontada: uno disciplinar, que le da significado a la lógica y a la dinámica del objeto de estudio y fundamenta el actuar, y otro, los contextos particulares de desempeño que definen su intencionalidad.

ASPECTOS ACADÉMICOS	CRITERIOS
	En este ámbito, la evaluación de una competencia requiere, por tanto, tener previamente identificados los contextos que, a la luz del objeto de formación, son lo suficientemente relevantes para la consolidación de dicho objeto - El desarrollo de una competencia académica remite a la relación sobre lo que se debe saber para saber hacer y saber actuar, relación que determina la relevancia del contexto disciplinar sobre el contexto específico de desempeño, puesto que, desde los contextos disciplinares se explican e interpretan las realidades de los contextos específicos de desempeño para intervenirlos adecuadamente, en procura de su desarrollo, de su mejoramiento, de su transformación, de la solución de problemas y necesidades. En síntesis, desde el contexto disciplinar se le imprime inteligencia al hacer y al actuar - Formulada una competencia en términos de la relación que existe entre un saber disciplinar y un contexto particular de desempeño, identificados los conceptos básicos del saber que deben fundamentar el hacer y diseñadas las estrategias metodológicas, se diseña la forma como se realizará la evaluación de la competencia, para lo cual se tienen como referentes fundamentales los indicadores de logro. Para su formulación, el indicador de logro identifica un contexto con unas determinadas condiciones para que el estudiante aplique los conceptos adquiridos
Desarrollo curricular	- El desarrollo curricular debe ser estructurado desde los microdiseños, en el contexto de la investigación formativa y a la luz del Modelo Pedagógico - El desarrollo microcurricular de cada asignatura debe contener como mínimo la identificación de las competencia y la elaboración para cada una de ellas de la red de conceptos básicos, los indicadores de logro, las estrategias de aprendizaje, los medios didácticos y los criterios de evaluación. - El seguimiento se realiza sobre cada una de las competencias diseñas en el micro de cada asignatura, lo cual implica hacer seguimiento a la apropiación de la base conceptual que ha de fundamentar el hacer o el actuar, y a la aplicación de dichos conceptos en contextos particulares específicos de significativa relevancia para la profesión.

ASPECTOS ACADÉMICOS	CRITERIOS
	▰ Para poder garantizar el desarrollo de las competencias, el seguimiento debe ser permanente y continuo y mediado por espacios de refuerzo, recuperación y formación complementaria.
Diseño metodológico del currículo	▰ El diseño metodológico del programa está elaborado con sujeción a la normatividad vigente, sobre características específicas de calidad para los programas de Contaduría Pública, Créditos Académicos y Lineamientos para la Acreditación del Programa. ▰ La formación integral del Contador Público comprende las siguientes áreas y componentes de formación: Área de formación básica y área de formación profesional. El área de formación básica está conformada por el pensamiento epistemológico, pensamiento matemático, pensamiento administrativo, pensamiento económico y pensamiento jurídico. El área de formación profesional está conformado por: pensamiento contable, pensamiento tributario-regulativo, pensamiento financiero, pensamiento organizacional, pensamiento de información, pensamiento humanista, pensamiento en investigación y electivas. La formación profesional será complementada con el desarrollo de competencias comunicativas básicas en una segunda lengua.
Espacios de formación integral	▰ Garantizar la creación de un ambiente académico propicio para el desarrollo de actividades que complementen la intencionalidad de la formación integral con calidad del estudiante, tales como: proyectos de investigación, semilleros de investigación, actividades culturales, artísticas y deportivas, promoción de la salud y prevención de la enfermedad.
Perfil del docente	▰ La selección y vinculación de docentes estará centrada en la naturaleza académica del programa; en la idoneidad ética, pedagógica y profesional; en el espíritu crítico de sus académicos y en su potencial creativo. ▰ Creación de condiciones académicas y financieras para el desarrollo integral del profesorado. ▰ Docentes inscritos en una dinámica de interacción académica y disciplinar con pares nacionales e internacionales, que mantengan la relación profesión, contexto, disciplinas y con producción intelectual que soporte los desarrollos del programa.
Permanencia y deserción	▰ Establecimiento de estrategias pedagógicas y actividades en espacios complementarios, para optimizar la tazas de retención y de graduación de los estudiantes, manteniendo la calidad académica del programa.

ASPECTOS ACADÉMICOS	CRITERIOS
	▰ Elaboración de estudios periódicos para identificar y evaluar las causas de la deserción.
Entorno académico	▰ El programa se desarrollará en un entorno académico que favorezca la consolidación del objeto de formación, mediante la incorporación adecuada, actualizada y suficiente de recursos bibliográficos, informáticos, de comunicación, de apoyo docente, de infraestructura técnica y tecnológica

La Ingeniería, según la Junta de Acreditación de Programas de Ingeniería y Tecnología de Accreditation Board for Engineering and Tecnology -ABET-, es la profesión en la cual los conocimientos de las ciencias naturales y matemáticas adquiridas mediante el estudio, la experiencia y la práctica se aplican con buen criterio para desarrollar los medios de aprovechar económicamente los materiales, los recursos y las fuerzas de la naturaleza, para el crecimiento y prosperidad de la humanidad.

La Ingeniería de Sistemas, según la profesora Mary Shaw de la Universidad de Carnegie Mellon "Es la construcción de soluciones efectivas y eficientes aplicando conocimiento científico a problemas prácticos para la creación de soluciones informáticas que estén al servicio de la humanidad".

El programa que se presenta corresponde a la tradición y fundamentación teórica de la Ingeniería, en particular de la Ingeniería de sistemas. En tal sentido y como se señala en el Resolución 2773 de 2003, los componentes de la formación comprenden los aspectos de la fundamentación básica de programas en ciencias aplicadas, fundamentación básica profesional del área de conocimiento de la Ingeniería y formación específica congruente con el nivel de formación y las competencias profesionales propias de la disciplina.

2.3.8 PROPÓSITOS GENERALES DEL PROGRAMA

Proporcionar una educación de excelente calidad para el ejercicio profesional en el campo de la Ingeniería de Sistemas , que apoye a la docencia universitaria y a la investigación en universidades, centros de investigación, empresas e incubadoras de empresas y que de igual forma permita desarrollar en el programa las funciones de la docencia, la investigación, la extensión o proyección social y la internacionalización, en el contexto de la Misión y el Proyecto Educativo Institucionales y de las tendencias nacionales e internacionales de desarrollo teórico y práctico del campo disciplinar de la Ingeniería de Sistemas

2.3.9 PERFIL DE FORMACIÓN PROGRAMA

2.3.9.1 PERFIL PROFESIONAL

El Ingeniero de Sistemas de la Corporación Universitaria Remington es un profesional con un alto sentido de responsabilidad social, humanista, competente e idóneo, con gran capacidad de análisis, diseño, y gestión de sistemas de información debido a su formación integral en las áreas de las ciencias básicas y computacionales, aplicando estándares internacionales y las mejores prácticas empleadas en la ingeniería del software permitiendo alcanzar el logro de los objetivos estratégicos en las organizaciones, desde una visión ética y responsable con el medio ambiente, a través del uso racional de recursos y la optimización de procesos empresariales.

2.3.9.2 PERFIL OCUPACIONAL

Las competencias anteriormente descritas, habilitan al Ingeniero de Sistemas, egresado de la Corporación Universitaria Remington para su desempeño en las siguientes áreas:

- Jefe departamento de sistemas
- Arquitecto de software
- Gestor de proyectos informáticos
- Auditor de sistemas
- Desarrollador de software
- Asesor y consultor de proyectos de sistemas informáticos
- Ingeniero diseñador, implementador y soporte de redes de datos.
- Diseñador e implementador de redes de datos
- Administrador de centro de cómputo
- Docente en el área de sistemas
- Analista de sistemas

2.3.10 PLAN DE ESTUDIOS

El programa de Ingeniería de Sistemas está sujeto al siguiente plan de estudio actualizado para el año en curso:

CARRERA PROFESIONAL

INGENIERÍA DE SISTEMAS

Registro Calificado MEN.

Resolución 2965 del 30 de mayo de 2007 - SNIES: 4118.

Título: Ingeniero de Sistemas.

Duración: 9 semestres

PLAN DE ESTUDIOS

NIVEL	ASIGNATURA	CREDITOS
1	ALGORITMOS 1	4
	TEORIA GENERAL DE SISTEMAS	2
	INTRODUCCION AL DESARROLLO DE SOFTWARE	4
	ADMINISTRACION Y ORGANIZACIONES	3
	MATEMATICAS DISCRETAS	4
	COMPETENCIAS COMUNICATIVAS	2
	TOTAL SEMESTRE	**19**

NIVEL	ASIGNATURA	CREDITOS
2	ALGORITMOS 2	4
	FUNDAMENTOS DE PROGRAMACIÓN	3
	CALCULO DIFERENCIAL	3
	ALGEBRA LINEAL	3

	METODOLOGIA DE LA INVESTIGACIÓN	2
	CONSTITUCIÓN POLÍTICA Y DERECHOS HUMANOS	2
	TOTAL SEMESTRE	**17**

NIVEL	ASIGNATURA	CREDITOS
3	ESTRCUTURA DE DATOS	4
	LENGUAJE DE PROGRAMACIÓN 1	3
	FISICA 1	3
	CALCULO INTEGRAL	3
	PROGRAMACION LINEAL	3
	ECOLOGIA HUMANA Y AMBIENTAL	2
	TOTAL SEMESTRE	**18**

NIVEL	ASIGNATURA	CREDITOS
	LENGUAJE DE PROGRAMACION 2	3

4	BASES DE DATOS 1	3
	INGENIERIA DE SOFTWARE 1	4
	ELECTRONICA	3
	FISICA 2	3
	ECUACIONES DIFERENCIALES	3
	TOTAL SEMESTRE	**19**

NIVEL	ASIGNATURA	CREDITOS
5	LENGUAJE DE PROGRAMACION 3	3
	BASES DE DATOS 2	3
	INGENIERIA DE SOFTWARE 2	4
	ARQUITECTURA DE COMPUTADORES	3
	INGLES 1	2
	ETICA	2
	TOTAL SEMESTRE	**17**

NIVEL	ASIGNATURA	CREDITOS
6	COMPILADORES	3
	INGENIERIA DE SOFTWARE 3	4
	REDES DE DATOS 1	3
	METODOS NUMERICOS	3
	INGLES 2	2
	ESTADISTICA Y PROBABILIDADES	3
	TOTAL SEMESTRE	**18**

NIVEL	ASIGNATURA	CREDITOS
7	SISTEMAS OPERATIVOS	3
	LENGUAJE DE PROGRAMACIÓN AVANZADO 1	3
	ARQUITECTURA DE SOFTWARE	3
	REDES DE DATOS 2	3

	MODELOS DE SIMULACIÓN	3
	LINEA DE ENFASIS	2
	TOTAL SEMESTRE	**17**

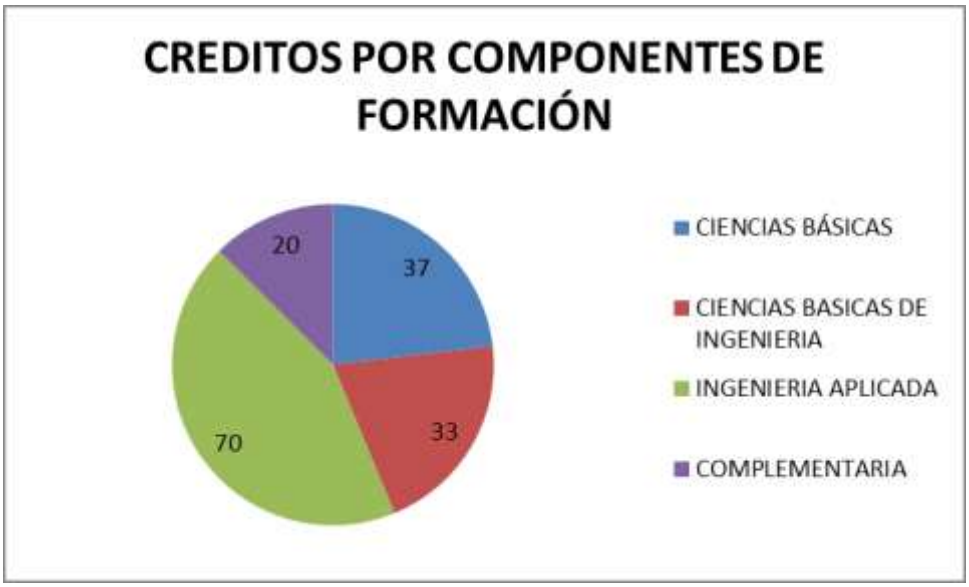
NIVEL	ASIGNATURA	CREDITOS
8	SISTEMAS DE INFORMACION	3
	LENGUAJE DE PROGRAMACIÓN AVANZADO 2	3
	GESTION Y EVALUACIÓN DE PROYECTOS	3
	AUDITORIA DE SISTEMAS	3
	INVESTIGACION DE OPERACIONES	3
	LINEA DE ENFASIS	2
	TOTAL SEMESTRE	**17**

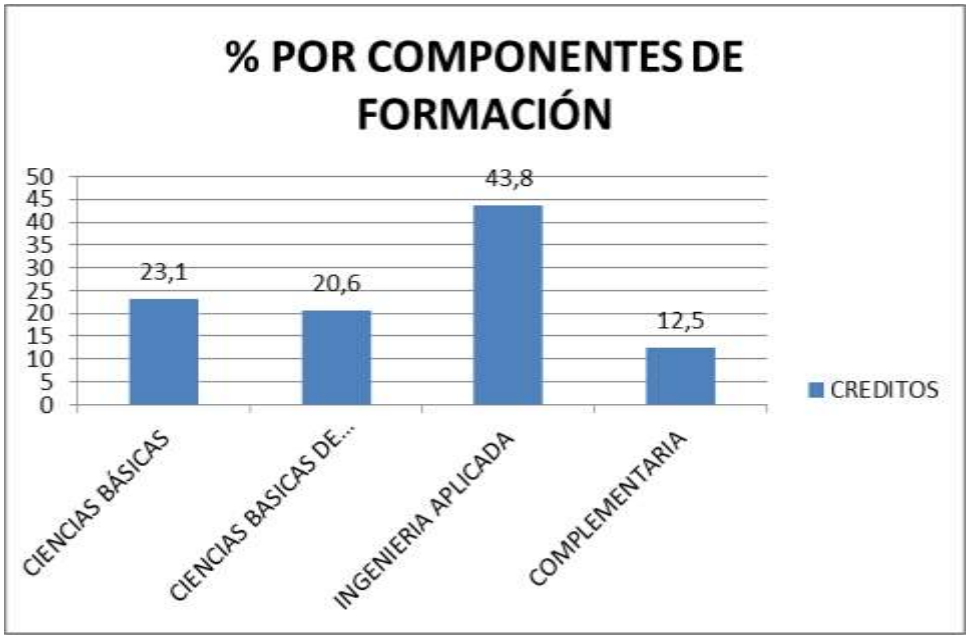
NIVEL	ASIGNATURA	CREDITOS
	GESTION DE PROYECTOS INFORMÁTICOS	4

9	ADMINISTRACION DE PERSONAL	3
	DERECHO INFORMATICO	3
	PRACTICA EMPRESARIAL	6
	LINEA DE ENFASIS	2
	TOTAL SEMESTRE	**18**
	TOTAL PROGRAMA	**160**

2.3.11 COMPONENTES DE FORMACIÓN

COMPONENTE DE FORMACION	CREDITOS	% COMPONENTES
CIENCIAS BÁSICAS	37	23.1%
CIENCIAS BASICAS DE INGENIERIA	33	20.6%
INGENIERIA APLICADA	70	43.8%
COMPLEMENTARIA	20	12.5%
TOTAL CRÉDITOS	**160**	





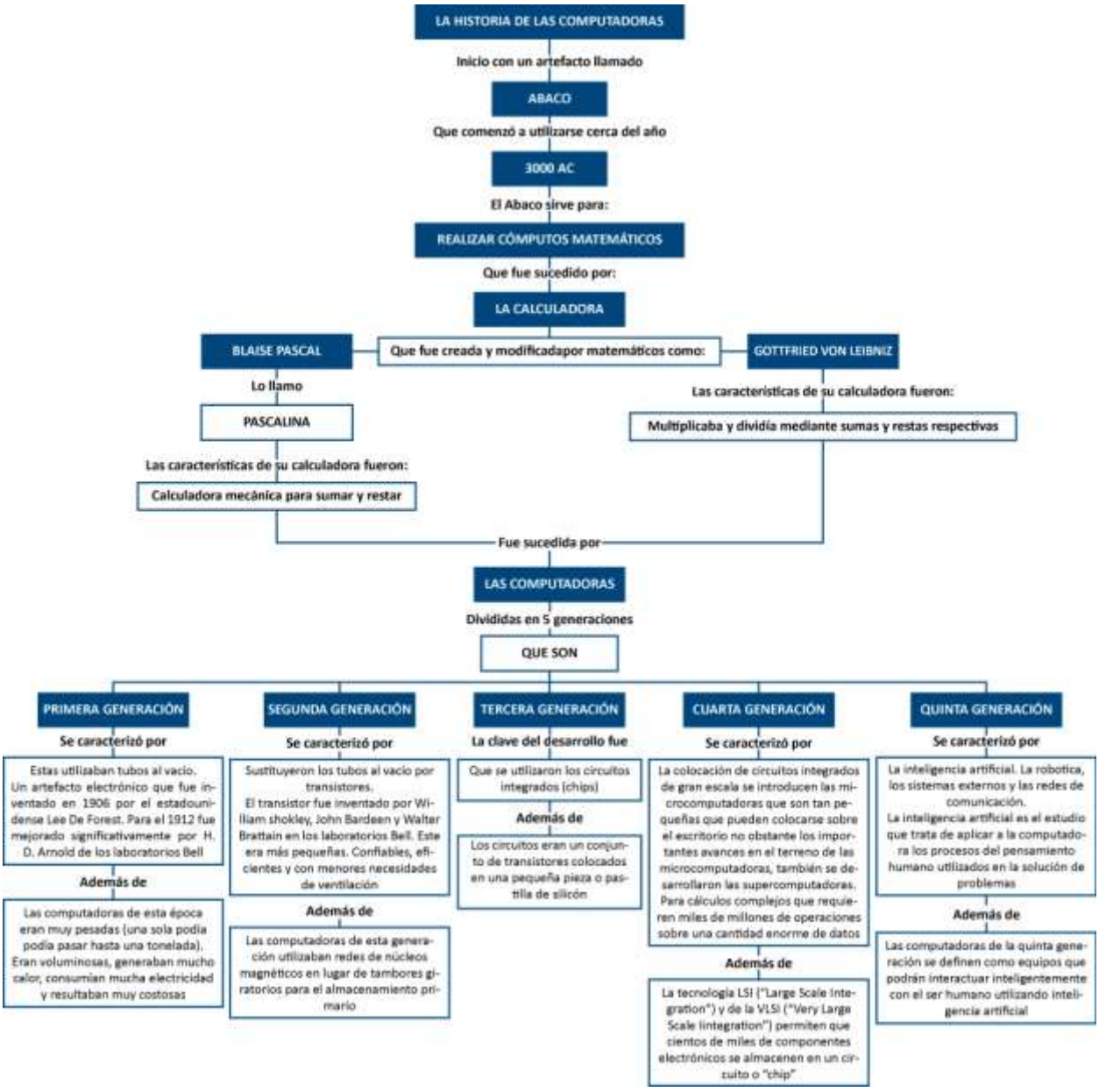
Ejercicio de Entrenamiento

Investigar sobre la actualidad de la ingeniería de sistemas en Latinoamérica y realizar un cuadro comparativo con la actualidad Colombiana.

3 UNIDAD 2 GENERALIDADES DEL DESARROLLO DE SOFTWARE

3.1 EVOLUCIÓN DEL COMPUTADOR

Para poder comprender adecuadamente el tema de la evolución del computador, es fundamental y esencial que se aborden los elementos que le dieron nacimiento y generaron su evolución, partiendo de lo que conocemos como historia de los computadores.



Geraldine, (2012). Historia y evolución del Pc [mapa conceptual]. Recuperado de http://geraldine75.blogspot.com/

La **historia de los sistemas computacionales** se remonta los años 40's, durante la segunda guerra mundial, no sin antes mencionar que estos procesos se venían realizando años atrás, es en esta época donde se presentan los grandes avances en esta área, con sistemas de cómputo como el Colossus, y su función inicial era decodificar mensajes de radio cifrados de los alemanes, en la misma década en los Estados Unidos un grupo de científicos desarrollo un sistema calculador e integrador numérico que se **bautizó como el ENIAC,** este fue uno de los primeros sistemas más populares, pese a su tamaño y limitantes es considerado por muchos como el inicio de este tipo de máquinas.

En los años 50's, aparecen los **transistores**, que eran componentes más pequeños, más rápidos y versátiles que sus antecesores, los tubos al vacío, estos permitían uso de menor energía y su vida útil era mucho más larga que los tubos, para este entonces se tomó el nombre de ordenador o computadoras de segunda general.

A principios de los años 60's, aparecen los **circuitos integrados**, esto facilito la fabricación de transistores y el espacio, y el valor era mucho más asequible para las industrias que los requerían, más adelante los microprocesadores aparecen en la década de 1970, esto ahorro mucho más los costos y la fabricación de componentes y equipos, aunque su costo era aún elevado **el tamaño era considerablemente más pequeño** se sus antecesores.

> En 1964, **aparece por primera vez el termino PC por sus siglas de Computador Personal),** y se obtuvo la primera patente con este nombre para identificar este tipo de dispositivos.

En la década del 80, es donde se dispara el concepto de PC en el mercado, cuando IBM, y su competidor más cercano Apple, crean de manera muy similar equipos que pueden ser transportados y manejados por una persona desde su hogar, en este momento Microsoft, ya ha prestado a IBM el sistema Operativo D.O.S, que se promueve y se populariza en este tipo de equipos electrónicos.

Estos dispositivos tienen un crecimiento constante hasta 1995, **cuando Microsoft lanza al mercado el sistema operativo Windows 95**, y se tiene un cambio en todos los aspectos informativos, haciendo que sean más asequibles, más económicos, las livianos y de mayor facilidad de uso, tanto en tareas de oficina, juegos, herramientas de comunicación**, la propagación de internet**, la comuniones entre usuario de distintas partes del mundo, este crecimiento ha sido constante durante los últimos años y a ha permitido que en cambio en la informática sea muy drástico en comparación con sus años previos.

Posterior a este gran lanzamiento aparecen otros sistemas operativos con mayor o menor aceptación como fueron Windows 98, Windows Milenium Edition que no tuvo éxito en el consumidor, luego aparece Windows 2000, **Windows XP, Windows 2003,** Windows vista con muchas críticas por su desempeño, aparece Windows 2008 y Windows 7. Todos estos han sido parte de la evolución del sistema operativo por parte de Microsoft, sin descuidar su rival de los años 70's y 80's, **Apple**, que tiene un sistema operativo grafico desde 1984, cosa que Microsoft alcanzo en 1995, esta empresa que ha evolucionado en mercado con sus productos exclusivo sigue a la vanguardia de creaciones novedosa, como son **en los último años el IPod, IPhone, Ipad**, y otros tantos que han hecho del mercado un gran fortín en la evolución de dispositivos electrónicos.

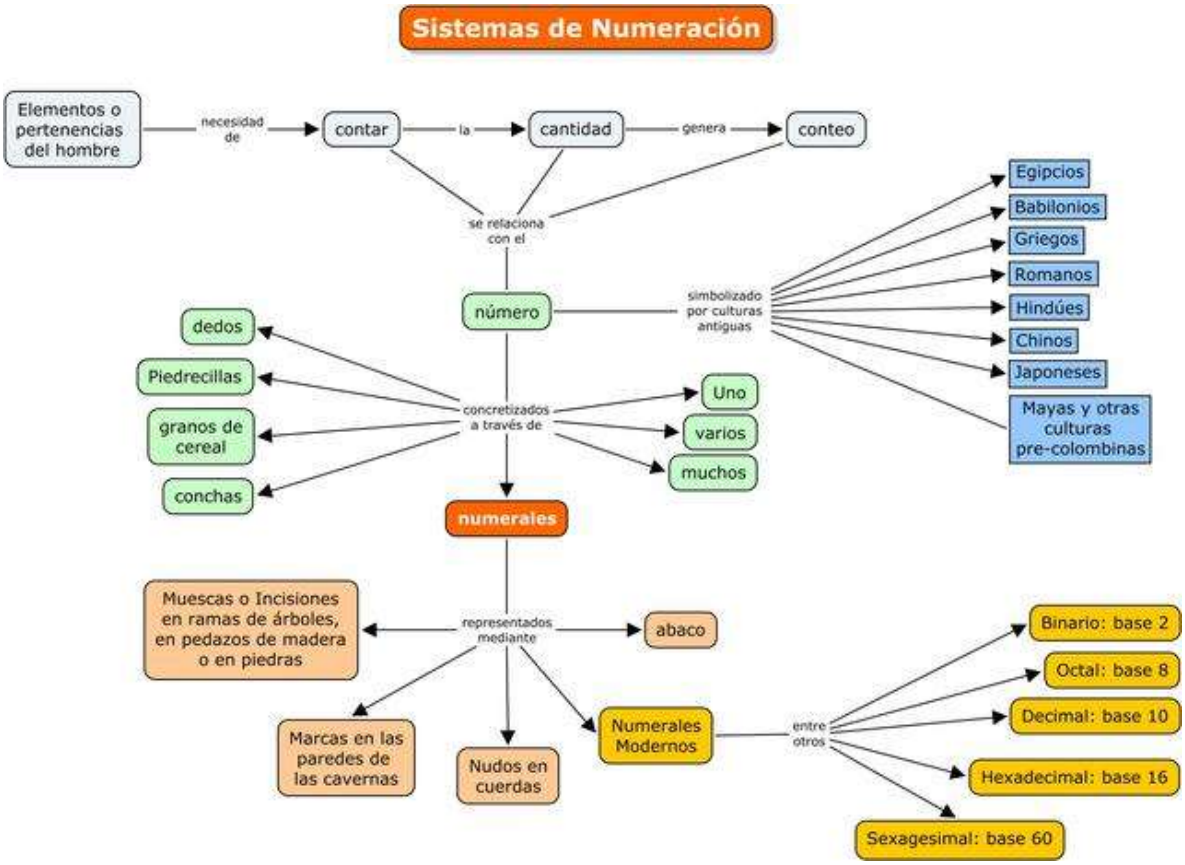
Los cambios más significativos que se han presentado **en los últimos años** han sido sin lugar a duda, **el acceso a internet, las redes sociales,** la mayor cercanía con dispositivos, al punto que el teléfono celular hace parte de esta tecnología por su uso y alcance, la mezcla de componentes como teléfono, cámara fotográfica, cámara de grabación de videos, agenda, navegación a internet, chat y otros **muchos recursos que han hecho que sea casi que indispensable en su portabilidad.**

Otro aspecto destacado ha sido sin duda, el software, esta componente lógico hace parte fundamental del manejo del PC, y ha tenido grandes retos partiendo de los sistemas operativos en los que hemos mencionado solo por parte de Microsoft, hay que tener presente que existen muchas más fabricantes y variedad de estos, dentro los que podemos mencionar el **Macintosh, sistema operativa de Apple, Linux, que hace parte de una gran comunidad de software libre, Solaris, creado especialmente para grandes dispositivos de industria como el internet, creado por Sun Microsistema**, pero además existe software ofimático, como el Office, el Corel Word Pro, el Lotus, y de ahí depende una gran cantidad de utilidades partiendo de juegos, herramientas de comunicación, de reparación , de recuperación, cada vez más encontramos herramientas de prevención y seguridad, entretenimiento, **lenguajes de programación y software a la medida de las empresas según su necesidad**.

CASTIBLANCO, J (2010). HISTORIA DE LA PC [IMAGEN]. RECUPERADO DE HTTP://SISTEMASKMILO.BLOGSPOT.COM/2010/08/HISTORIA-DE-LA-PC-MAPA-CONCEPTUAL.HTML

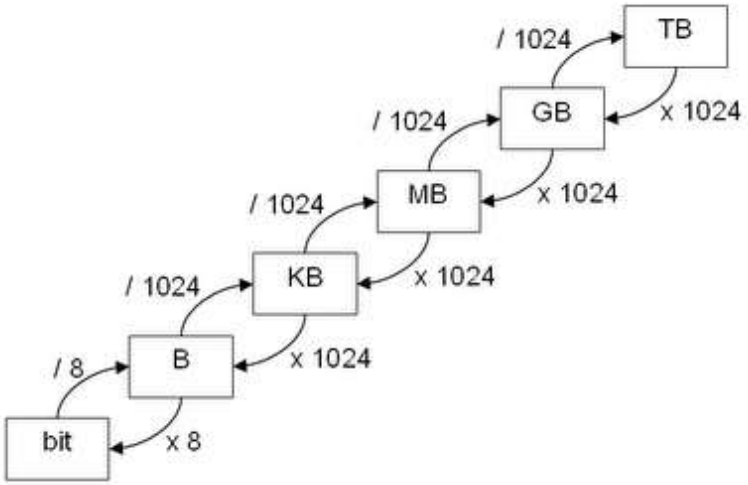
FECHAS Y HECHOS IMPORTANTES	
40’s	Aparece el sistema calculador e integrador numérico que se bautizó como el ENIAC
50’s	aparecen los transistores
60’s	Aparecen los circuitos integrados
1964	Aparece por primera vez el termino PC por sus siglas de Computador Personal), y se obtuvo la primera patente con este nombre para identificar este tipo de dispositivos.
80’s	IBM y Apple, crean de manera muy similar equipos que pueden ser transportados y manejados por una persona desde su hogar.
1995	Microsoft lanza al mercado el sistema operativo Windows 95

3.2 SISTEMA NUMÉRICOS Y DE ALMACENAMIENTO



Matemáticas, Fs (2014). Sistema de numeración - un concepto / opinión personal [Mapa conceptual]. Recuperado de http://matematicasfps.wikispaces.com/Sistemas+de+Numeraci%C3%B3n

Mapa 3 Conversiones de Medidas 3-Unidad 2



DumitruAlcantara, (2011). Unidades de almacenamiento en informática [Imagen]. Recuperado de http://dumitrualcantara.blogspot.com/2011/10/unidades-de-almacenamiento-en.html

> Durante toda la historia del PC, **ha existido la necesidad de almacenar la información procesada** en el, en algún momento esta información era volátil, posteriormente se presentaron **las tarjetas perforadas**, los discos duros, los discos flexibles, las unidades ópticas y las memorias portátiles, todas ellas de gran ayuda **desde el inicio de los computadores hasta nuestros días,** y encontramos una variedad creciente de estos dentro de los cuales podemos destacar.

Bit

Unidad mínima de procesamiento conformada por un impulso que **puede ser 1 o 0**

Byte

Unidad mínima de almacenamiento, representa un carácter y se conforma por 8 bits, una secuencia de 1 o 0, ejemplo 01000001

Kilo byte (KB)

Es equivalente a 1024 bytes o caracteres

Mega byte (MB)

Es equivalente a 1024 KB

Giga byte (GB)

Es equivalente a 1024 MB

Tera byte

Es equivalente a 1024 GB

Peta byte

Es equivalente a 1024 TB

Exa byte

Es equivalente a 1024 PB

Zetta byte

Es equivalente a 1024 EB

Yotta byte

Es equivalente a 1024 ZB

Estas son las medidas de uso de almacenamiento más comunes, dentro del PC, o computador personal existen limitantes con estas medidas, pero en otros sectores de la industria es muy común hablar de todas estas, a nivel de PC, se pueden mencionar dispositivos como la memoria RAM, que puede almacenar hasta **GB,** discos duros que puede almacenar hasta **TB**, unidades ópticas pueden almacenar desde **MB** hasta **GB,** al igual que las memorias portátiles.

La Aplicación de estas unidades **es muy variada**, en este curso se presentaran ocasiones de conversión entre diferentes tipos de ellas.

3.2.1 EJERCICIO DE APRENDIZAJE

3.2.1.1 CONVERTIR

Según la necesidad del usuario de almacenar datos en distintos dispositivos se deben tener presente que **pasar de una unidad pequeña a una más grande implica división(es)** para obtener el resultado esperado, **si por el contrario se desea convertir de valores grandes a pequeños estos se multiplican por la unidad en común que es presenta que es 1024.**

- Pasar 24 GB a KB

El paso de esta unidad equivale que de una unidad grande como son los GB pasamos a una pequeña como son los KB, **esto implica que existe una multiplicación de valores**

24 x 1024 = 24.576 MB, como se nos indica que es KB **debemos de multiplicar nuevamente el valor para obtener el resultado esperado.**

24.576 MB x 1024 = **25.165.824 KB**

Este sería el resultado esperado para este punto en particular

- Pasar 125.908.854.098 KB a TB

En este caso estamos pasando de una unidad pequeña como es el KB a una unidad grande como es el TB, **esto implica divisiones para obtener el resultado esperado**

125.908.854.098 KB / 1024 = 122.957.865, 33 **MB**

120.076,04 MB / 1024 = 117, 26 **GB**

117, 26 GB / 1024 = **0.11 TB**

Estos son los tipos de conversión más comunes pero se presenta la necesidad de convertir según el medio de almacenamiento que se requiera

- Se desea almacenar 1.2 GB en unidades ópticas de 210 MB, cuántas de estas unidades se requieren.

1.2 GB x 1024 = 1.228.8 MB

1.228.8 / 210 = **5.85 dispositivos ópticos**

Respuesta 6 dispositivos ópticos de 210 MB

- 204 CD's de 700 MB se desean almacenar en DVD's de 4.7 GB, cuántos de estos DVD's se requieren.

204 x700 =142.800MB

142.800 MB / 1024 = 139.45

139.45 / 4.7 = **29.67**

Respuesta 30 DVD's de 4.7 GB

Además de este tipo de conversiones es muy común encontrarnos que deseamos **descargar información de la web,** y requerimos **conocer sea tiempo, tamaño del archivo o velocidad de descarga,** esto implican cálculos similares a los anteriores.

- Se desea descargar un archivo de 5 GB a una velocidad "Constante" de 103 KB/s, indique el tiempo en horas de la descarga de dicho archivo

Para este tipo de casos encontraremos unas unidades comunes para cualquier tipo de operación similar, si hablamos de tamaño este debe ser especificado en KB y si se mencionan tiempos o duración esta se expresa en Segundos

5 GB x 1024 = 5.120 MB

5.120 MB x 1024 = 5.242.880 KB

Teniendo el valor del archivo en KB, se procede a determinar los tiempos de descarga, partiendo de que existe una velocidad "Constante" de 103 KB/s

5.242.880 / 103 = 50.901 **Segundos** se tarda este archivo en descargar

50.901 / 60 = 848 **Minutos**

848 / 60 = 14.13 **Horas** es el tiempo de descarga del archivo de 5 GB

- Se descarga un archivo en 14 horas, 32 minutos 56 segundos, a una velocidad "Constante" de 154 KB/s, cual es el peso del archivo descargado en GB.

Como las **unidades en común son KB** y **Segundos**, para este caso debemos hallar los segundos que tarda la descarga.

14 x 60 = 840 Minutos

840 x 60 = 50.400 Segundos

32 x 60 = 1.920 Segundos

50.400 + 1.920 + 56 = 52.376 Segundos en tiempo total

52.376 x 154 = 8.065.904 KB

8.065.904 / 1024 = 7.876.85 MB

7.876.85 MB / 1024 = **7.69 GB**

- Se descarga un archivo de 240 MB, en 1 hora, 27 minutos, 23 segundos, cual fue la velocidad promedio de descarga de este archivo.

240 MB x 1024 = 245.760 KB

1 x 60 = 60 Minutos

60 x 60 = 3600 Segundos

27 x 60 = 1.620 Segundos

3600 + 1620 + 23 = 5.243 Segundos es el tiempo de descarga

245.760 / 5243 = **46.87 KB/s**

PISTAS DE APRENDIZAJE

Traer a la memoria:

Tenga en cuenta que al manejar medidas de almacenamiento debe saber cuáles son estas, que nombres reciben y cuáles son sus equivalencias en las otras medidas.

Tenga cuidado cuando convertimos de una medida mayor a una menor se multiplica y cuando convertimos de una medida menor a una mayor se divide.

No olvide que la mínima medida de medida en informática es el byte y todas las demás medidas se construyen teniendo esta como base.

Tenga presente que siempre que vaya a realizar una conversión debe saber:

1. Que nos dieron y a que lo vamos a convertir
2. Qué medida es mayor y cual es menor
3. Que operación debe realizar si una multiplicación o una división.
4. La regla tres simple o una operación directa

Dentro de la informática es muy común escuchar el tema de **los sistemas numéricos** como herramienta de funcionamiento del PC, téngase en presente que **todo lo que interpretan los sistemas de cómputo son números**, independientemente que esta información esté representada en imágenes, videos, música, texto, juegos, o cualquier otro tema, aunque existen varias capas para esto **son 4 los sistemas más comunes dentro del manejo computacional.**

Binario
Sistema con **base 2**, conformado por **1 y 0**

Octal
Sistema con **base 8,** conformado por valores que **van de 0 a 7**

Decimal
Sistema con **base 10**, conformado por valores que **van de 0 a 9**

Hexadecimal
Sistema con **base 16,** conformado por valores que **van de 0 a 9 y de A a F**

Si tomamos como ejemplo casos comunes como representar el **símbolo @** mediante sistema numérico, en muchas ocasiones encontramos que para suplir este carácter dentro del teclado **utilizamos Alt + 64 = @,** este número 64 pertenece a un grupo de caracteres denomina **ASCII**, que

comprende los 256 caracteres especificados para el manejo de información del PC, estos **van del 0 al 255** y representan caracteres en mayúscula y minúscula, números y símbolos.

Pero este valor 64 no es reconocido internamente por el PC, que para ser **interpretado** adecuadamente deberá **convertirse en binario** para su uso correcto.

> El **sistema binario** es una secuencia de números que **contiene unos y ceros**, así; 1000000, **todos los valores** que se manipulan internamente para ser interpretados por el PC **sufren esta transformación**

3.2.2 EJERCICIO DE APRENDIZAJE

3.2.2.1 CONVERSIÓN

3.2.2.1.1 Decimal a Binario

Para pasar decimales a binarios, **se realizan división sucesivas por 2**, sin tener en cuenta los decimales hasta **que el divisor sea 0**, luego **se toman los resultados del último al primero** formando una secuencia de valores **de izquierda a derecha**.

- Convertir el valor 89 de decimal a binario

89 / 2 = 44 Residuo 1

44 / 2 = 22 Residuo 0

22 / 2 = 11 Residuo 0

11 / 2 = 5 Residuo 1

5 / 2 = 2 Residuo 1

2 / 2 = 1 Residuo 0

1 / 2 = 0 Residuo 1

Observemos que en todos los **resultados de residuo esta conformado por unos por ceros**, teniendo esta operación terminada procedemos a ubicar los valores **1011001** y obtenemos esta cifra.

3.2.2.1.2 Binario Decimal

Para convertir valores de binario a decimal, **existen varios métodos**, uno de ellos es **elevar a base 2**, y tener presente que **solo se suman los valores que contengan en binario un uno**, esta base **inicia con el valor más a la derecha** de la serie y **termina** en el mas a la **izquierda** de este.

2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	**0**	**1**	**1**	**0**	**0**	**1**
64	32	**16**	**8**	4	2	**1**

En **la primera fila obtenemos la base 2 elevado al valor de 0 a la N**, en la segunda fila los valores referentes al binario que se desea convertir, **en la cuarta fila, los valores equivalentes a la base 2 a la N**, para **obtener el valor real** oculto dentro de este binario **realizamos la suma de los valores que contienen 1 en el binario**

1 + 8 + 16 + 64 = 89

- Conversión a sistema Octal

Para las conversiones de los 2 sistemas numéricos faltantes existen múltiples formas, para un mayor aprovechamiento sin hacer uso de procesos complejos se utilizara el método binario como base de los demás sistemas mencionados

3.2.2.1.3 Decimal a Octal

- Pasar 294 de decimal a octal

Para esta conversión se realizara de decimal a binario y luego de este a octal.

294 / 2 = 147 Residuo 0

147 / 2 = 73 Residuo 1

73 / 2 = 36 Residuo 1

36 / 2 = 18 Residuo 0

18 / 2 = 9 Residuo 0

9 / 2 = 4 Residuo 1

4 / 2 = 2 Residuo 0

2 / 2 = 1 Residuo 0

1 / 2 = 0 Residuo 1

Valor Binario

100100110

Teniendo el valor **binario** se precede a utilizar la siguiente tabla que será útil tanto para **binario a octal como de octal a binario**

	4	2	1
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Para realizar la conversión del valor binario, **se toman grupos de 3 cifras**, de **derecha a izquierda** y cada bloque se **reemplaza** con la tabla antes creada

100	100	110
4	**4**	**6**

Este resultado es **446**, indicando que esta es **la equivalencia de 294 del decimal**

3.2.2.1.4 Octal a Decimal

- Para un valor octal téngase presente que solo se compone con cifras entre 0 y 7
- Octal 4567

> Para la conversión de este a **decimal,** primero se convierte a **binario,** para esto tomamos cada cifra del número **octal** y **se reemplaza con la tabla antes mencionada.**

4	5	6	7
100	**101**	**110**	**111**

El resultado de este valor es

100101110111

Teniendo este valor convertido procedemos a pasarlo a **decimal**

1	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**
2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
2048	1024	512	**256**	128	**64**	**32**	**16**	8	**4**	**2**	**1**

Para finalizar **se suman** los valores de la potencia y el valor del **binario en uno**

1 + 2 + 4 + 16 + 32 +64 + 256 + 2048 = **2423**

3.2.2.1.5 Hexadecimal

Para estos sistemas número que **está conformado por números de 0 a 9 y de A a F**, se tiene la siguiente tabla.

	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10 - A	1	0	1	0
11 – B	1	0	1	1
12 – C	1	1	0	0
13 – D	1	1	0	1
14 – E	1	1	1	0
15 - F	1	1	1	1

3.2.2.1.6 Hexadecimal a Decimal

- 2F4A

> Dentro del **manejo hexadecimal**, se tienen **números y letras,** es muy común encontrar valores como el anterior 2F4A, hay que **tener presente** que valores como el **10 se representa con la A, el 11 con la B y así hasta el número 15 que se representa con la F.**

Para pasar este valor de hexadecimal a decimal, **primero se pasa a binario** y luego al valor solicitado, esta conversión se realiza seleccionando valor por valor y reemplazando con la tabla antes vista.

2	F	4	A
0010	**1111**	**0100**	**1010**

Valor en binario es **0010111101001010**

Si se desea convertir este valor **binario** a **decimal** se procede de la manera tradicional de elevar 2 a la n.

0	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**0**	**1**	**0**	**0**	**1**	**0**	**1**	**0**
2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

2 + 8 + 64 + 256 + 512 + 1024 + 2048 + 8192 = **12106**

3.2.2.1.7 Hexadecima l a Octal

- Pasar FF9 a Octal

F	F	9
1111	**1111**	**1001**

Valor en binario

111111111001

- Paso a octal

111	111	111	001
7	**7**	**7**	**1**

Resultado de pasar

FF9 de hexadecimal a Octal es **7771**

- Pasar de Octal a Hexadecimal

3541

3	5	4	1
011	**101**	**100**	**001**

Resultado en binario

011101100001

Para pasar de **binario a hexadecimal**, selecciona **grupos de cuatro dígitos** y los reemplaza con la tabla.

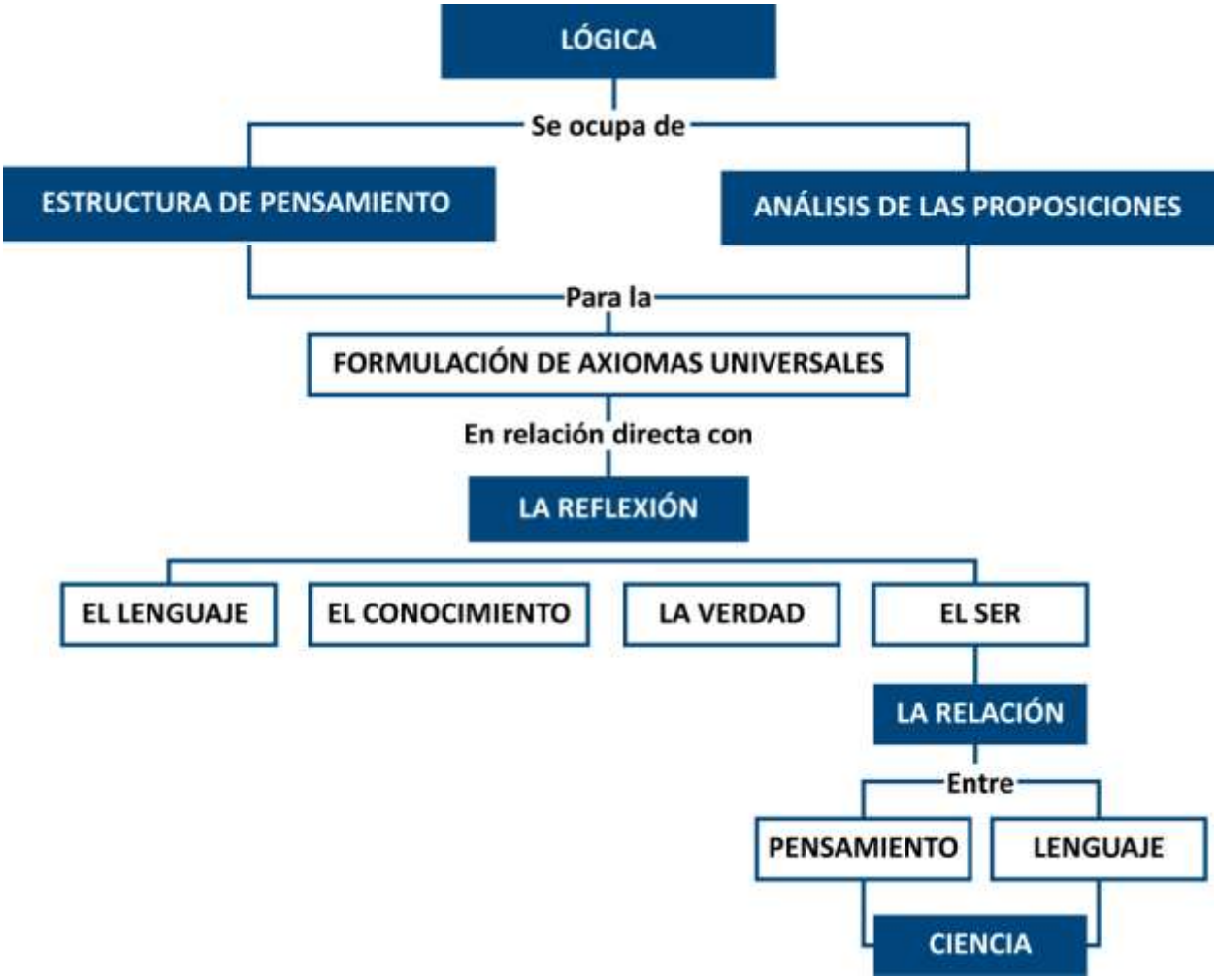
0111	0110	00001
7	**A**	**1**

Resultado de pasar 3541 de octal a hexadecimal es **7 A 1**

3.3 LÚDICA Y RAZONAMIENTO LÓGICO

La algoritmia vista desde punto de ciencia que nos **propone la lógica que se debe aplicar a diferentes situaciones problemitas computacionales**, tiene una serie de **etapas básicas** para su desarrollo y complemento para áreas afines dentro de la carrera.

El mapa 1 de la perspectiva 1-Unidad 2



Unad, (2014). Mapa Conceptual Del Curso Académico De Lógica Formal Y Simbólica [imagen]. Recuperado de http://datateca.unad.edu.co/contenidos/103300/103300exe/mapa_conceptual.html

> Toda esta **ciencia** tiene unos **pasos** que son y serán **fundamentales** durante toda la **vida profesional** del **desarrollador.**

3.3.1 PLANTEAMIENTO DEL PROBLEMA

Consiste en una "**conversación**" simple, en la que se **especifica** que **se tiene y que se quiere solucionar o mejorar**, es quizás la **etapa más importante** porque determinamos **cuales son**

las características del entorno, las entradas de la información y los resultados esperados.

3.3.2 ANÁLISIS DE LA PROPUESTA

Después de tener el planteamiento, nos centramos en analizar esta situación, **determinar todos sus pormenores, sus características, su beneficio**, este análisis nos **permitirá llegar a conclusiones y realizar posiblemente un nuevo planteamiento** más cercano a la parte computacional y **ver diferentes alternativas de solución**.

3.3.3 DESARROLLO DEL PROBLEMA

Cuando se tiene un análisis muy centrado, y enfocado en una tarea particular, se puede iniciar el proceso de desarrollo o **darle solución a la propuesta inicial**, de esta se desprenderán algunas características básicas dentro de los algoritmos como pueden ser, la información que entra, la información que se procesa, y la información que sale, el desarrollo **puede contener múltiples métodos y múltiples herramientas que permitan hacer un proceso más dinámico**.

3.3.4 CODIFICACIÓN

Si el problema tiene como condición sistematizar computacionalmente, este pasa a la codificación, que **no es otra cosa que interpretar el desarrollo anterior y plasmarlo en papel con todas las características previamente expresadas**, esta codificación ya está **interpretada** mediante un **lenguaje de programación**, que tiene su normas y métodos de trabajo muy similares a las de el algoritmo creado.

3.3.5 DIGITACIÓN

La digitación **es el método de pasar la información a un editor de un lenguaje de programación**, donde se buscaran opciones y alternativas prácticas para hacer del problema una forma fácil y ágil de operar, esta digitación **se realiza con comando e instrucciones propias que dependen del lenguaje y se interpretara por el PC y mostrar los procesos específicos**.

3.3.6 COMPILACIÓN

Terminada la digitación, el paso siguiente es compilar, este es un **proceso interno del lenguaje que determina si lo digitado está dentro de las normas y las características del lenguaje de programación**, esta verificación se encargara básicamente de las **sintaxis** más no de la lógica del programador.

3.3.7 EJECUCIÓN

Esta etapa es la que **permitirá ver el programa funcionando**, se realiza en primeras etapas con el fin de ver la lógica funcionando del programa, recuerde que **la compilación solo se encarga de la sintaxis, mas no de lógica**, cuando se ha probado la lógica, ingresando, procesando y obteniendo resultados, ya se podrá considerar un programa.

Hay que tener presente que esta no es la última etapa, **existe varios métodos adicionales que complementan esta tarea**, como pueden ser el desarrollo grafico que permitirá dar una presentación, distribución, color, fondos, tipos de letra, tamaños de esta, etc., y la opción de validación que se encargara de control todos los procesos que tengan un ingreso correcto con el fin de determinar un correcto funcionamiento del programa solucionado.

> **Esta última etapa será fundamental para cualquier tipo de aplicativo desarrollado, pues con el tendremos la certeza de controlar todo lo que el usuario ingrese y que los resultado sean los esperados y no un conjunto de errores que no se tenían presupuestados.**

Este conjunto de opciones harán del desarrollo una tarea completa, que con el tiempo y la experiencia se irán depurando y realizando cada día mejores opciones.

3.3.8 LÚDICA

Las **herramientas lúdicas,** tiene por **objetivo desarrollar, afianzar, mejorar las condiciones lógicas del individuo,** con el fin de tener **mejores amas de trabajo cuando esta se enfoque en conceptos computaciones, la relación de esto es "LOGICA",** no solo se desarrolla está al frente de un PC, sino que se tiene opciones del común que permitirán este desarrollo apropiado.

3.3.8.1 SUDOKU

Es Sudoku, es un **juego que tomo su poderío en Japón** a principio de los años 80's, **es una matriz de 9 x 9,** (el formato más clásico de este juego), y **consiste en una colección de 9 filas, 9 columnas, y 9 áreas más pequeñas, en las que no se deben de repetir número, símbolos o letras, la forma más tradicional se trabajó con números del 1 al 9**, estos no se deben de repetir en ninguna forma en las filas o en las columnas y en los cuadros más pequeños, **tiene por condición adicional que el resultado es único,** y de ninguna manera se podrán presentar 2 con los mismos números y distinto resultado.

				1		2		
8	7						4	
	3	4	8			9		
		2	1			7	6	
			7		5			
	4	9			8	5		
		5			7	4	2	
	6						3	5
		3		6				

En este primer grafico **observamos que es un cuadrado de 9 x 9 filas y 9 cuadros enmarcados**, si se observa detenidamente aparecen una serie de números que nos darán una guía, de que números falta por ubicar, **a mayor cantidad de números en el tablero inicial, será más sencillo la búsqueda del resultado definitivo**, y al contrario, a medida que este tenga menos números su solución será más compleja.

> Para iniciar la solución de este ejemplo, tendremos que observar detenidamente cada número y su comportamiento dentro del tablero.

Como los números no se pueden repetir, observemos en la primera fila esta el 1, pero en la segunda y la tercera no está en ninguna posición, este no sería un número ideal para iniciar.

Lo mismo sucede con el número 2, el número 3, 7, y 9, no serían número que permitan la solución inmediata de estos valores.

Si observamos el número 8, este se encuentra en la segunda fila y tercera fila, nos indica que **se podría colocar en la primera fila**, en el tercer cuadro pequeño, observemos el grafico.

				1		2		
8	7						4	
	3	4	8			9		
		2	1			7	6	
			7		5			
	4	9			8	5		
		5			7	4	2	
	6						3	5
		3		6				

El área en gris, nos indican la posición donde debería estar el número 8, pero existe una dificultad, al existir 2 posibilidades es complejo indicar cuál es el correcto, esto nos llevaría a que coloquemos un valor en una posición equivocada, posteriormente tengamos que iniciar nuevamente el juego.

En el segundo grupo de filas, la 4, 5 y 6 serán ahora nuestro nuevo objetivo de trabajo.

Observemos nuevamente que el número 1, 2, 4, 6, 7, 8 y 9 tienen un solo valor en las 3 filas, situación que dificulta su solución, en numero 5 tiene una situación similar al primer intento que realizamos.

				1		2		
8	7						4	
	3	4	8			9		
		2	1			7	6	
			7		5			
	4	9			8	5		
		5			7	4	2	
	6						3	5
		3		6				

> Iniciamos un nuevo recorrido con las filas 7, 8 y 9, esto nos indica que la solución de un Sudoku, no tiene un orden específico en su solución.

En este último grupo de filas encontramos dificultades con el número 2, 4, y 7, pero podríamos encontrar una posible solución en el número 5, observemos que el número 5 se encuentra en la fila 7, en la fila 8, nos quedaría solo la fila 9.

				1		2		
8	7						4	
	3	4	8			9		
		2	1			7	6	
			7		5			
	4	9			8	5		
		5			7	4	2	
	6						3	5
		3		6				

Esto nos indicaría que el número 5 se podría ubicar al lado del número 6, pero tenemos igualmente 2 posibles ubicaciones, si observamos con cuidado, **una de estas columnas está ocupada con el número 5 y de esta manera solo nos quedaría una posición por llenar.**

				1		2		
8	7						4	
	3	4	8			9		
		2	1			7	6	
			7		5			
	4	9			8	5		
		5			7	4	2	
	6						3	5
		3	5	6				

> Esta es la manera más apropiada de solucionar este tipo de juegos lúdicos, descartando opciones hasta llenar los espacios vacíos.

Con el número 6 de las filas 8 y 9 podríamos realizar la misma función

				1		2		
8	7						4	
	3	4	8			9		
		2	1			7	6	
			7		5			
	4	9			8	5		
		5			7	4	2	6
	6						3	5
		3	5	6				

Obsérvese que el número 6 después de descartar la fila 8 y 9 nos permite una sola ubicación la fila 7, este es el ideal para el llenado, debe tener presente que la misma función se aplica a las columnas.

Observemos el resto de la solución del ejercicio

Número 6

				1		2		
8	7					6	4	
	3	4	8			9		
		2	1			7	6	
			7		5			
	4	9			8	5		
		5			7	4	2	6
	6						3	5
		3	5	6				

Número 2

				1		2		
8	7					6	4	
2	3	4	8			9		
		2	1			7	6	
			7		5			
	4	9			8	5		
		5			7	4	2	6
	6						3	5
		3	5	6				

Número 7

				1		2		
8	7					6	4	
2	3	4	8	7		9		
		2	1			7	6	
			7		5			
	4	9			8	5		
		5			7	4	2	6
	6						3	5
		3	5	6				

Número 6

				1		2		
8	7					6	4	
2	3	4	8	7		9		
		2	1			7	6	
			7		5			
	4	9	6		8	5		
		5			7	4	2	6
	6						3	5
		3	5	6				

Número 7

				1		2		
8	7					6	4	
2	3	4	8	7		9		
		2	1			7	6	
			7		5			
7	4	9	6		8	5		
		5			7	4	2	6
	6						3	5
		3	5	6				

Número 7

				1		2		
8	7					6	4	
2	3	4	8	7		9		
		2	1			7	6	
			7		5			
7	4	9	6		8	5		
		5			7	4	2	6
	6	7					3	5
		3	5	6				

Número 2

				1		2		
8	7					6	4	
2	3	4	8	7		9		
		2	1			7	6	
			7		5			
7	4	9	6		8	5		
		5			7	4	2	6
	6	7					3	5
	2	3	5	6				

Número8

				1		2		
8	7					6	4	
2	3	4	8	7		9		
		2	1			7	6	
			7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7					3	5
	2	3	5	6				

Número 5

				1		2		
8	7			5		6	4	
2	3	4	8	7		9		
		2	1			7	6	
			7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7					3	5
	2	3	5	6				

Número 8

				1		2		
8	7			5		6	4	
2	3	4	8	7		9		
		2	1			7	6	
			7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6				

Número 1

				1		2		
8	7	1		5		6	4	
2	3	4	8	7		9		
		2	1			7	6	
			7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6				

Número 8

				1		2		
8	7	1		5		6	4	
2	3	4	8	7		9		
		2	1			7	6	
		8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6				

Número 8

				1		2		
8	7	1		5		6	4	
2	3	4	8	7		9		
		2	1			7	6	8
		8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6				

Número 8

				1		2	8	
8	7	1		5		6	4	
2	3	4	8	7		9		
		2	1			7	6	8
		8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6				

Número 6

		6		1		2	8	
8	7	1		5		6	4	
2	3	4	8	7	6	9		
		2	1			7	6	8
		8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6				

Número 5

		6		1		2	8	
8	7	1		5		6	4	
2	3	4	8	7	6	9	5	
		2	1			7	6	8
		8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6				

Número 7

		6		1		2	8	7
8	7	1		5		6	4	
2	3	4	8	7	6	9	5	
		2	1			7	6	8
		8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6				

> Observe que al final de la fila 3 falta solamente un número, el faltante en este caso en el número 1, luego de colocar este queda un faltante en el cuadro pequeño.

		6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
		2	1			7	6	8
		8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6				

Número 6

		6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
		2	1			7	6	8
6		8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6				

Número 7

		6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
		2	1			7	6	8
6		8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6			7	

Número 8

		6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
		2	1			7	6	8
6		8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6		8	7	

Número 1

		6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
		2	1			7	6	8
6	1	8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6		8	7	

Número 3

		6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3		2	1			7	6	8
6	1	8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6		8	7	

Número 5

5		6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7		5			
7	4	9	6		8	5		
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6		8	7	

Número 1

5	9	6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7		5			
7	4	9	6		8	5	1	
	8	5			7	4	2	6
	6	7		8			3	5
	2	3	5	6		8	7	

5	9	6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7		5			
7	4	9	6		8	5	1	
	8	5			7	4	2	6
	6	7		8		1	3	5
	2	3	5	6		8	7	

5	9	6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7		5			
7	4	9	6		8	5	1	
	8	5			7	4	2	6
	6	7		8		1	3	5
	2	3	5	6	1	8	7	9

5	9	6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7		5			
7	4	9	6		8	5	1	
1	8	5			7	4	2	6
	6	7		8		1	3	5
	2	3	5	6	1	8	7	9

Número 3

5	9	6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7		5	3		
7	4	9	6		8	5	1	
1	8	5			7	4	2	6
9	6	7		8		1	3	5
4	2	3	5	6	1	8	7	9

Número 4

5	9	6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7		5	3		4
7	4	9	6		8	5	1	
1	8	5			7	4	2	6
9	6	7		8		1	3	5
4	2	3	5	6	1	8	7	9

Número 2

5	9	6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7	2	5	3	9	4
7	4	9	6		8	5	1	2
1	8	5			7	4	2	6
9	6	7		8		1	3	5
4	2	3	5	6	1	8	7	9

Número 3

5	9	6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7	2	5	3	9	4
7	4	9	6	3	8	5	1	2
1	8	5			7	4	2	6
9	6	7		8		1	3	5
4	2	3	5	6	1	8	7	9

Número 3

5	9	6		1		2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7	2	5	3	9	4
7	4	9	6	3	8	5	1	2
1	8	5	3		7	4	2	6
9	6	7		8		1	3	5
4	2	3	5	6	1	8	7	9

5	9	6		1	3	2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7	2	5	3	9	4
7	4	9	6	3	8	5	1	2
1	8	5	3	9	7	4	2	6
9	6	7		8		1	3	5
4	2	3	5	6	1	8	7	9

Número 4

5	9	6	4	1	3	2	8	7
8	7	1		5		6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1			7	6	8
6	1	8	7	2	5	3	9	4
7	4	9	6	3	8	5	1	2
1	8	5	3	9	7	4	2	6
9	6	7		8		1	3	5
4	2	3	5	6	1	8	7	9

Número 2

5	9	6	4	1	3	2	8	7
8	7	1	9	5	2	6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1	4		7	6	8
6	1	8	7	2	5	3	9	4
7	4	9	6	3	8	5	1	2
1	8	5	3	9	7	4	2	6
9	6	7	2	8	4	1	3	5
4	2	3	5	6	1	8	7	9

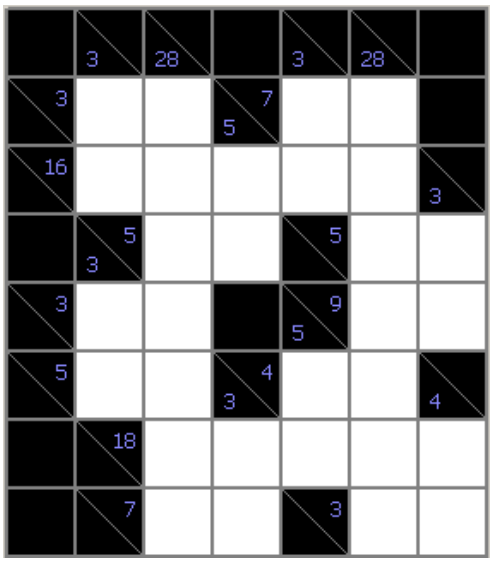
Solución

5	9	6	4	1	3	2	8	7
8	7	1	9	5	2	6	4	3
2	3	4	8	7	6	9	5	1
3	5	2	1	4	9	7	6	8
6	1	8	7	2	5	3	9	4
7	4	9	6	3	8	5	1	2
1	8	5	3	9	7	4	2	6
9	6	7	2	8	4	1	3	5
4	2	3	5	6	1	8	7	9

3.3.8.2 KAKURO

El **kakuro** es otro **juego de origen Japonés**, que nos permite **desarrollar habilidades numéricas**, en un proceso **similar a un crucigrama**, este tiene diferentes tamaños y formas y debe de arrojar un valor que nos pueda ser útil tanto en filas y columnas**, sus principales condiciones es que solo acepta valores entre el 1 y el 9 sin repetir por bloque de números.**

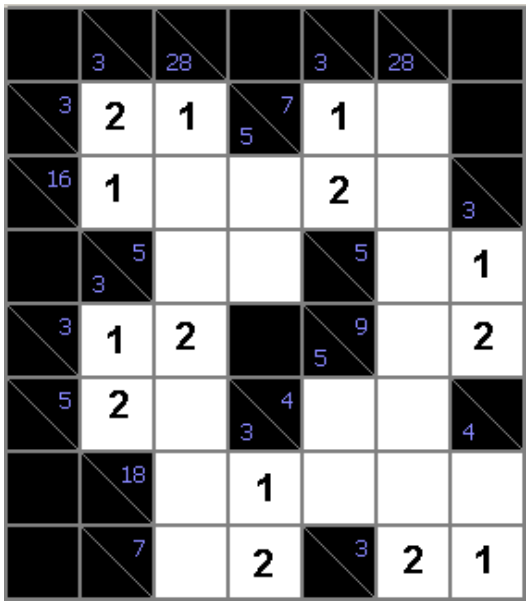
Este es un ejemplo claro de un **kakuro**



Si se observa con detenimiento, encontramos una serie de números, **si el número aparece en la parte inferior del cuadro es que en esa columna debe de sumar ese valor**, **si el valor aparece a la derecha indica que esa fila debe sumar el valor especificado**.

Recomendable a la hora de llenar este tipo de procesos, iniciar con números pequeños es el ideal, de modo que no existan muchas alternativas de valores que puedan complicar el desarrollo del ejercicio, ejemplo si tenemos que encontrar 2 valores que sumen 3 solo tendríamos dos posibles combinaciones, 1 y 2 o 2 y 1, esto porque dependiendo de la ubicación puede interferir con otros resultados, si el valor buscar es un 5, ya tenemos mayor número de combinaciones, 1 y 4, 2 y 3, 3 y 2, 4 y 1.

Resolvamos estos valores iníciales.



En este punto resolvimos todos los que en suma horizontal o vertical debe de dar 3, si vemos en detalle algunos inicial con 1 y otros con 2, es un proceso que se va dando a medida que se tengan más números.

Con estos valores ya podemos colocar algunos que solo tienen 2 opciones.

	3	28		3	28	
3	2	1	7 / 5	1	6	
16	1			2		3
	5 / 3			5	4	1
3	1	2		9 / 5	7	2
5	2	3	4 / 3			4
	18		1			3
	7	5	2	3	2	1

Estos valores ubicados no tenían más combinaciones según el orden especificado al inicio, **para los demás existe un grado de complejidad más amplio**, ya que tenemos que colocar valores que no interfieran con otros resultados.

	3	28		3	28	
3	2	1	7 / 5	1	6	
16	1		4	2		3
	5 / 3		1	5	4	1
3	1	2		9 / 5	7	2
5	2	3	4 / 3			4
	18		1			3
	7	5	2	3	2	1

> En este grafico colocamos la suma de 5, la combinación que utilizamos es 4 y 1, no se podría la otra 1 y 4 porque en esa misma fila ya existía este valor (observar fila 3, columna 4)

	3	28		3	28	
3	2	1	7 / 5	1	6	
16	1	6	4	2	3	3
	5 / 3		1	5	4	1
3	1	2		9 / 5	7	2
5	2	3	4 / 3			4
	18		1			3
	7	5	2	3	2	1

En este grafico anterior, se ubican el 6 y al final el 3, (fila 3, columna 4), **el único orden que teníamos para sumar 16 era este**, si observamos el 3 en la columna 3 ya existía.

	3	28		3	28	
3	2	1	7 / 5	1	6	
16	1	6	4	2	3	3
	5 / 3	4	1	5	4	1
3	1	2		9 / 5	7	2
5	2	3	4 / 3			4
	18	7	1			3
	7	5	2	3	2	1

Al terminar la columna 3, observamos que faltaba un 4 en la fila 4 y un 7 en la penúltima fila**, y con esto tenemos la suma de 28 que se nos solicitaba.**

	3	28		3	28	
3	2	1	7 / 5	1	6	
16	1	6	4	2	3	3
	5 / 3	4	1	5	4	1
3	1	2		9 / 5	7	2
5	2	3	4 / 3	3		4
	18	7	1	2		3
	7	5	2	3	2	1

Para la columna 5, fila 6, colocamos un 3 para la suma de 5 que se solicita, era la única combinación que teníamos en este caso, porque horizontalmente se solicitaba un 4, y esta solo se puede realizar con 3 y 1 o 1 y 3.

	3	28		3	28	
3	2	1	7 / 5	1	6	
16	1	6	4	2	3	3
	5 / 3	4	1	5	4	1
3	1	2		9 / 5	7	2
5	2	3	4 / 3	3	1	4
	18	7	1	2		3
	7	5	2	3	2	1

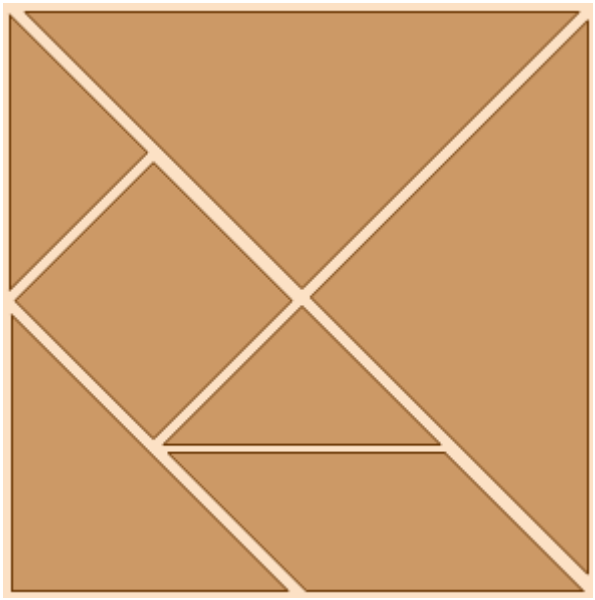
> Nos queda solo una casilla por llenar, horizontalmente se solicita la suma de 18 y verticalmente la suma de 28.

Solución del Kakuro

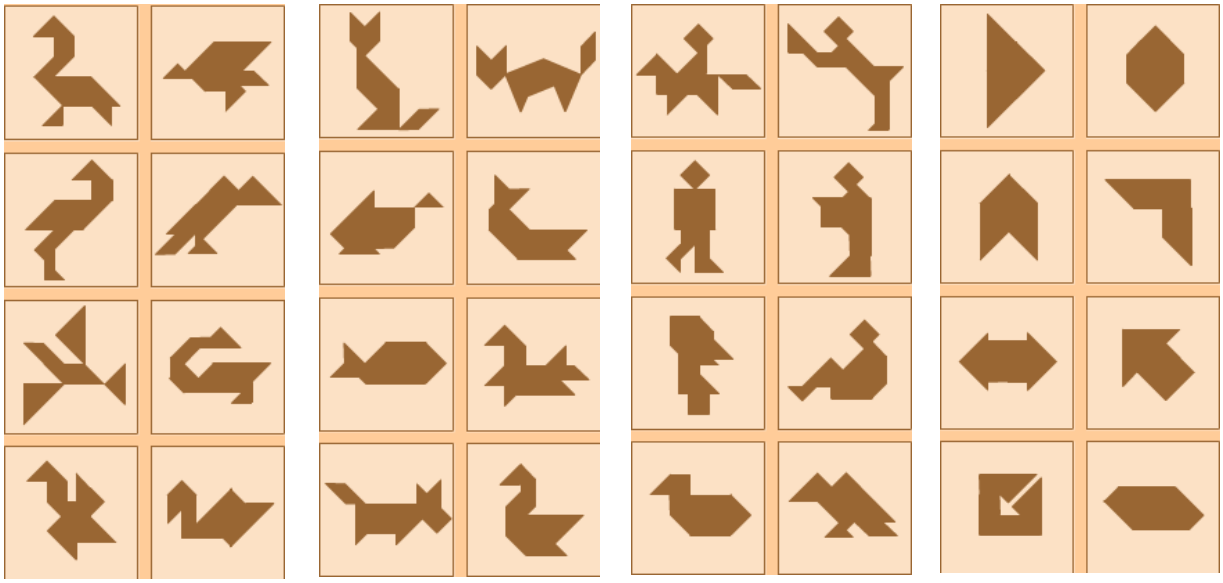
	3	28		3	28	
3	2	1	7 / 5	1	6	
16	1	6	4	2	3	3
	5 / 3	4	1	5	4	1
3	1	2		9 / 5	7	2
5	2	3	4 / 3	3	1	4
	18	7	1	2	5	3
	7	5	2	3	2	1

3.3.8.3 TANGRAM

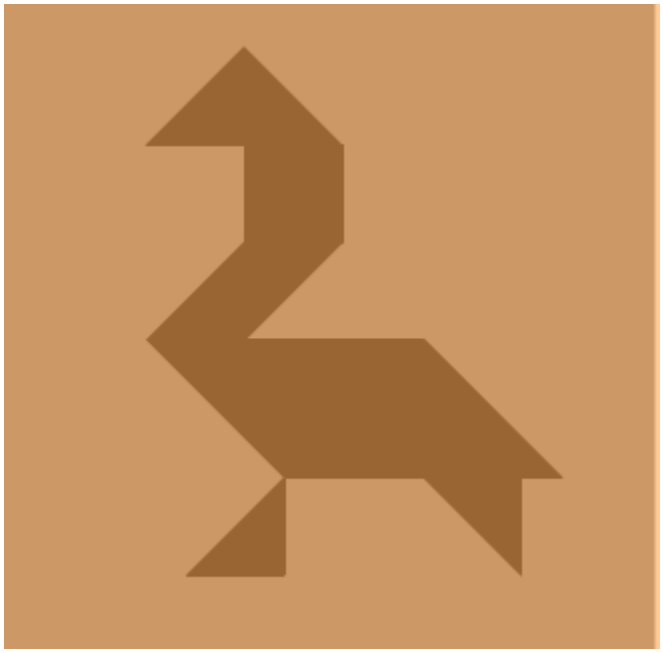
Tangram **es un juego de origen chino**, que **consta de 7 fichas 5 de ellas triángulos** con las que se **pretenden realizar una serie de figuras con la condicional de que no deben sobrar fichas**, siempre se deben de utilizar las 7, dentro de esto tenemos



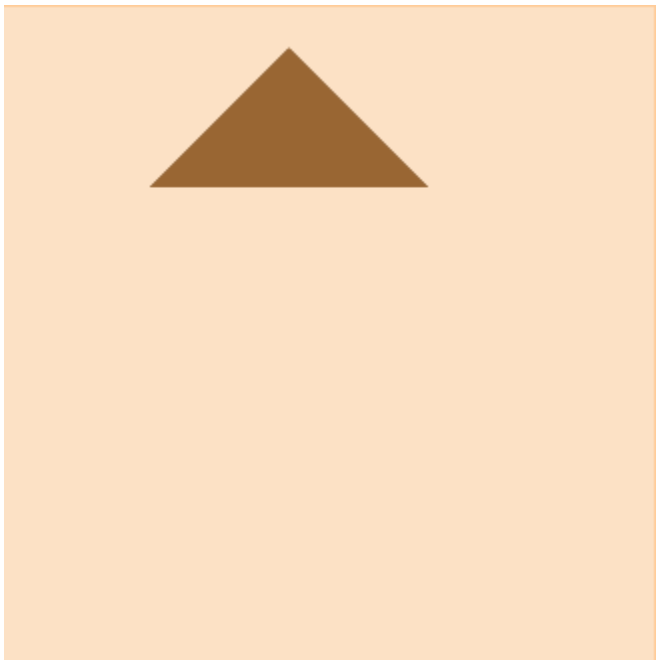
> Esta es su forma inicial, observe las 7 fichas conformando un cuadrado, dentro de las opciones que se tienen se pueden realizar más de 500 figuras dentro de las que encontramos de forma representativa las siguientes.



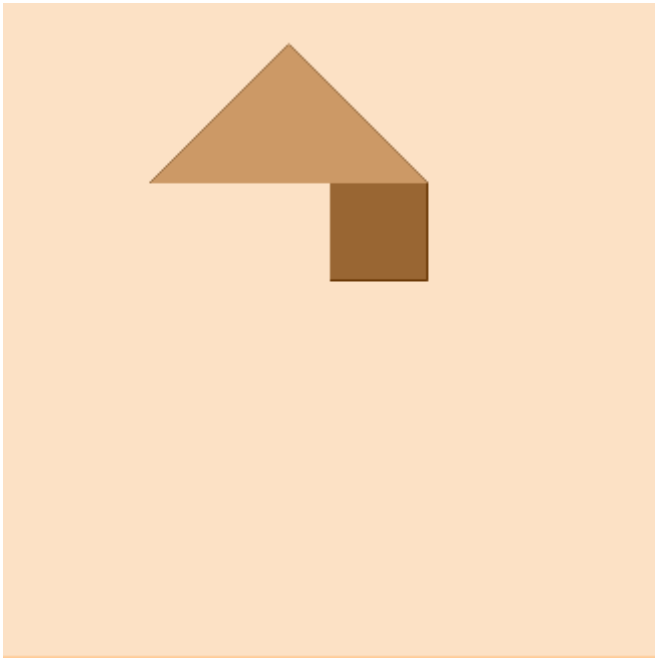
Vamos a darle solución a una de estas graficas mediante el uso de la herramienta del tangram.



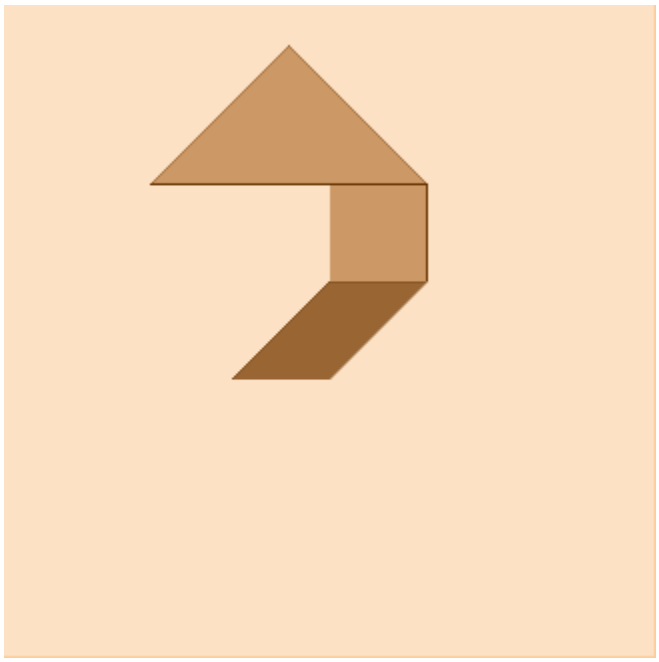
Este grafico está conformado por las 7 fichas del tangram, debemos de ubicarla de la mejor manera para que tenga la misma forma que la muestra que tenemos.



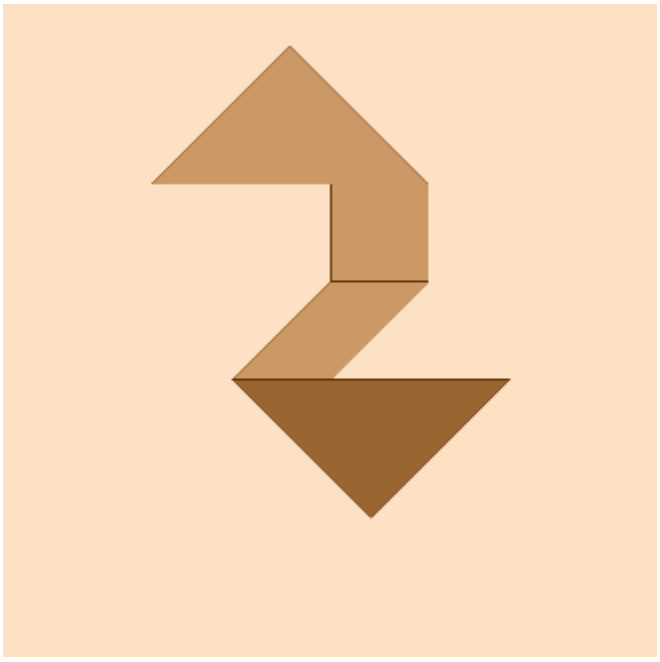
En este grafico ubicamos uno de los triángulos, para este caso el intermedio en la parte superior.



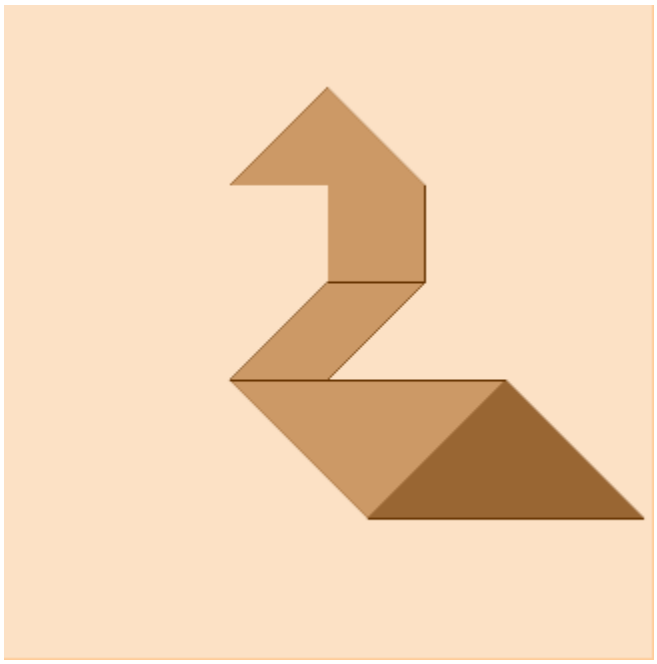
Luego colocamos el cuadrado en uno de los extremos del triángulo.

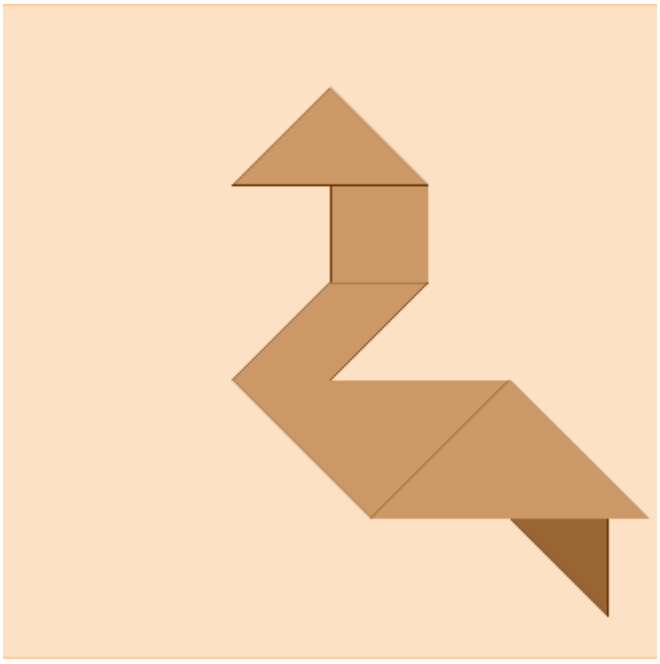


Con esta nueva figura le damos forma al cuello de la figura planteada.



Ubicamos el otro triangulo, en este caso uno de los más grandes de la figura, iniciamos la construcción del cuerpo.





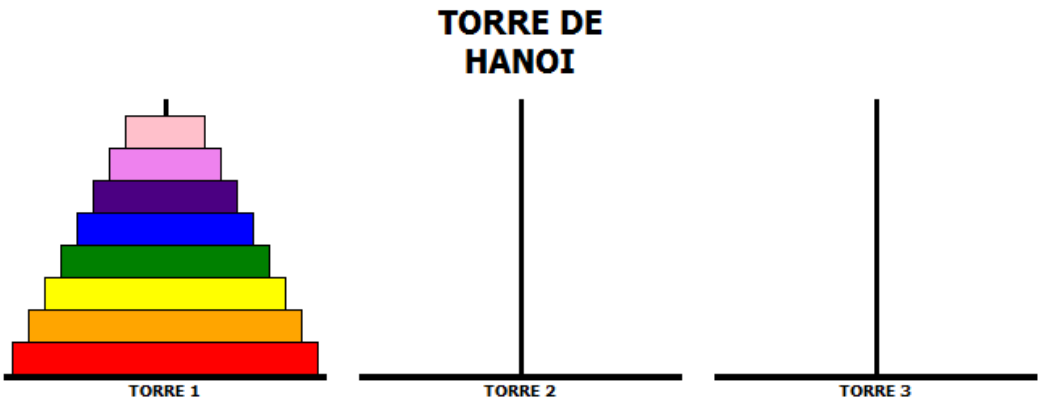
Solución del ejemplo.

> De esta manera, utilizando las 7 fichas, podrá armar un sinnúmero de gráficos, que permitirán el desarrollo lógico, creativo y recursivo que se requiere en la parte de programación.

3.3.8.4 HANOI

La torre de Hanoi, **es una de las más representativas dentro de los aspectos lógicos por su apoyo a la recursividad**, es un **juego matemático** que a medida que se ubican más discos, estos duplican la cantidad de movimientos de la opción anterior, si esta se inicia con 3 discos, los movimientos mínimos para solucionarla es de 7, **pero si se colocan 4 discos su solución mínima es de 15 movimientos.**

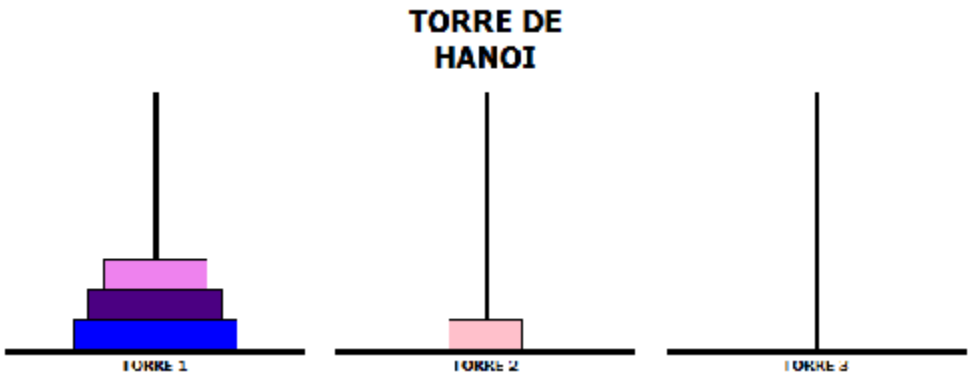
Dentro de las **normas más significativas** que tiene este juego encontramos que, **solo podremos mover ficha a ficha**, se debe formar la torre inicial en uno de los postes adiciones que se tienen y **nunca debe ir un disco grande sobre uno pequeño.**



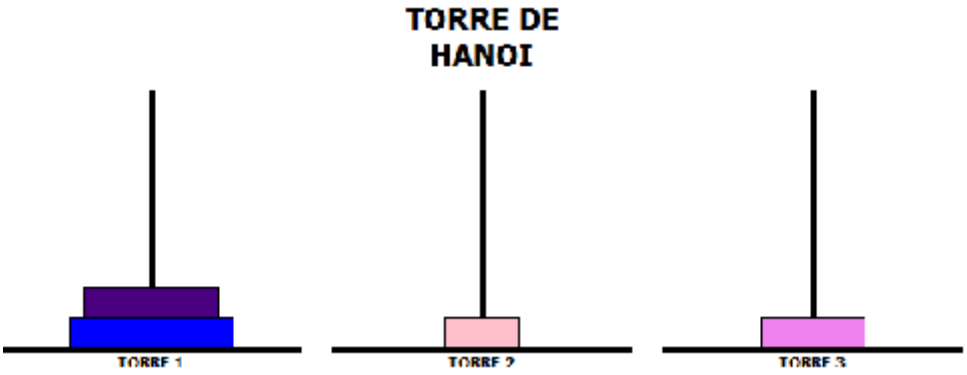
Podemos observar que consta de 3 torres, en su máxima expresión consta de 8 discos para un mínimo de movimientos de 255, si este valor se sobrepasa, debemos de intentar hasta disminuir al máximo la cantidad de movimientos, es una herramienta de aporte a la recursividad de los procesos.

Recreemos un ejemplo con 4 discos, un mínimo de 15 movimientos.

Movimiento 1



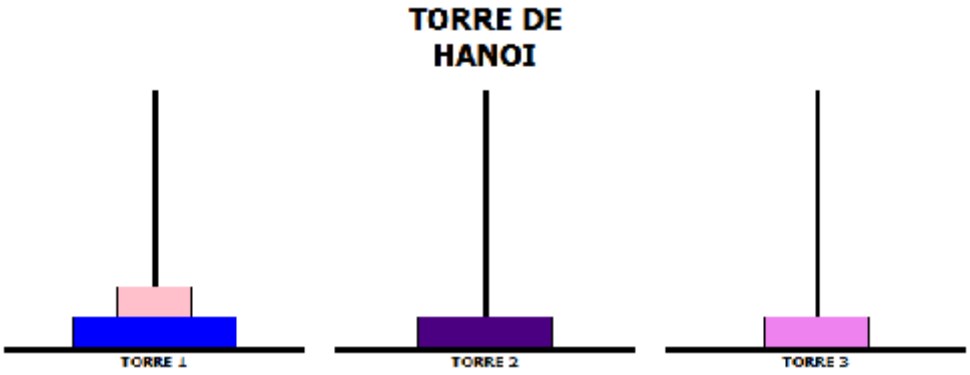
Movimiento 2



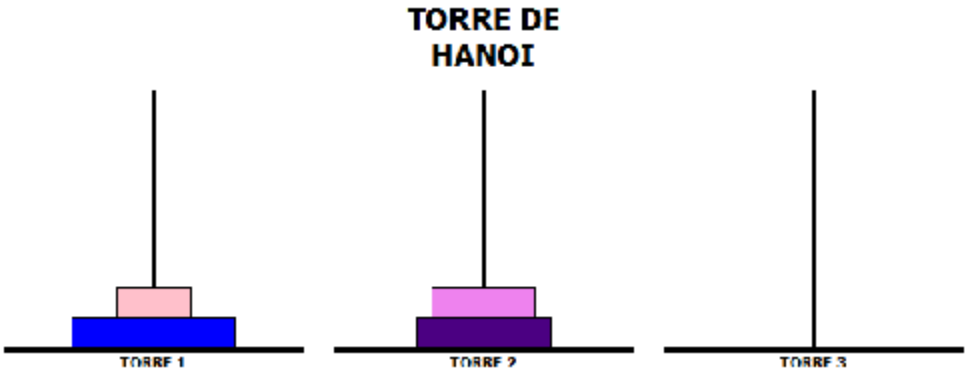
Movimiento 3

Movimiento 4

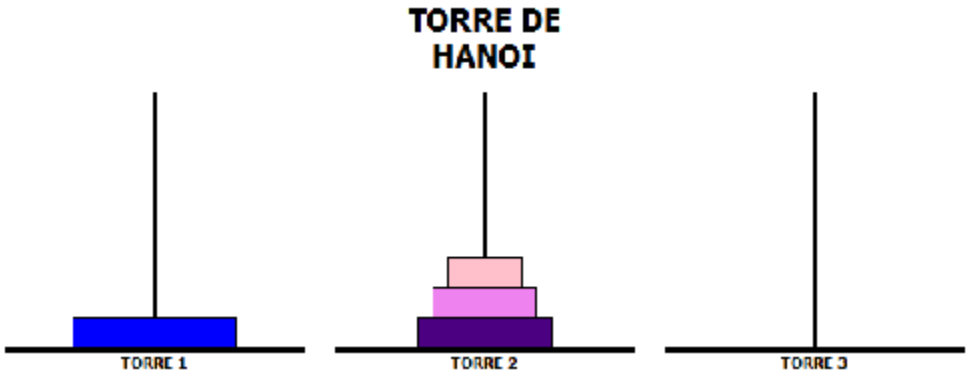
Movimiento 5



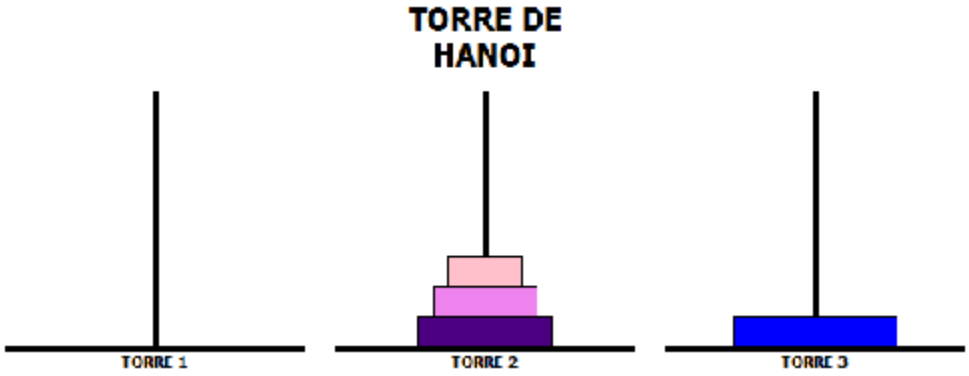
Movimiento 6



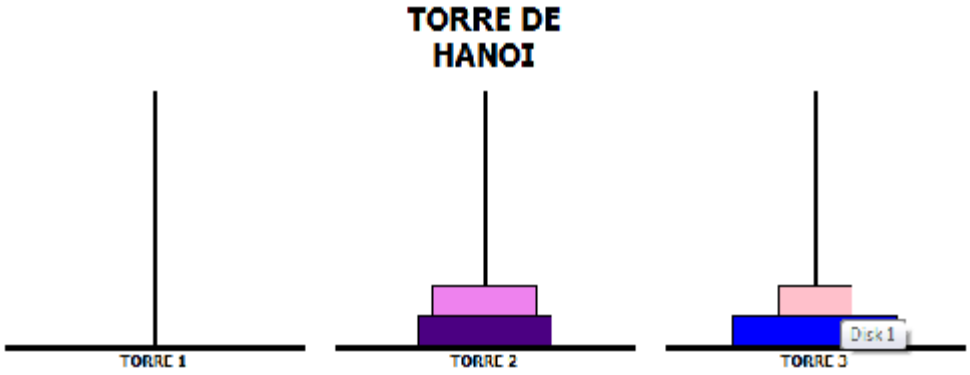
Movimiento 7



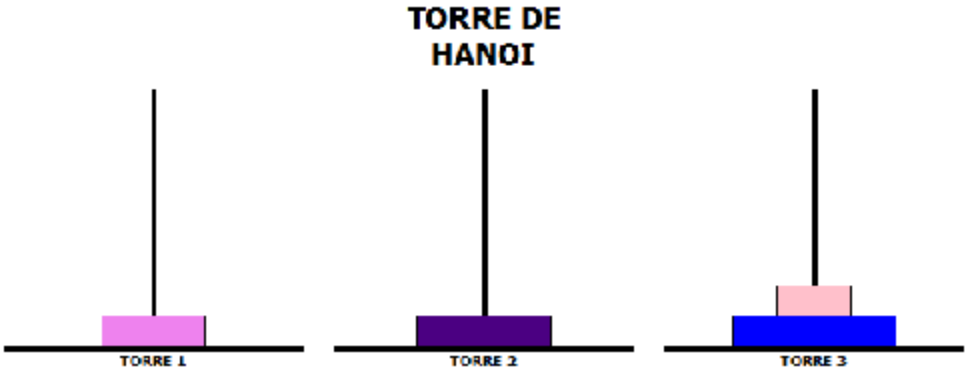
Movimiento 8



Movimiento 9

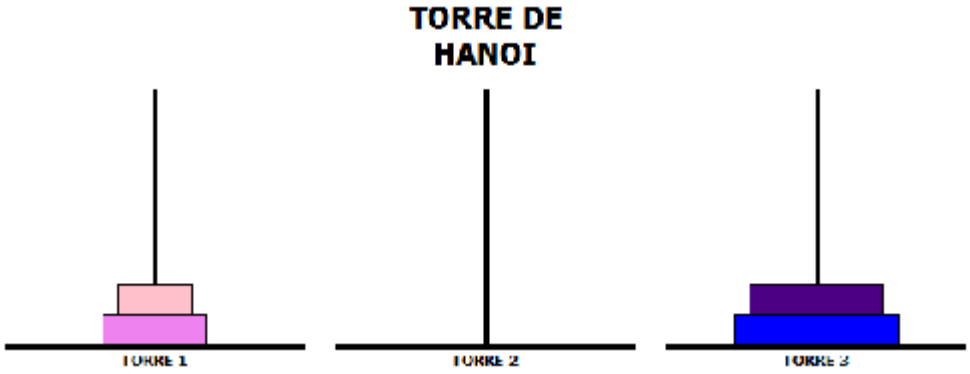


Movimiento 10

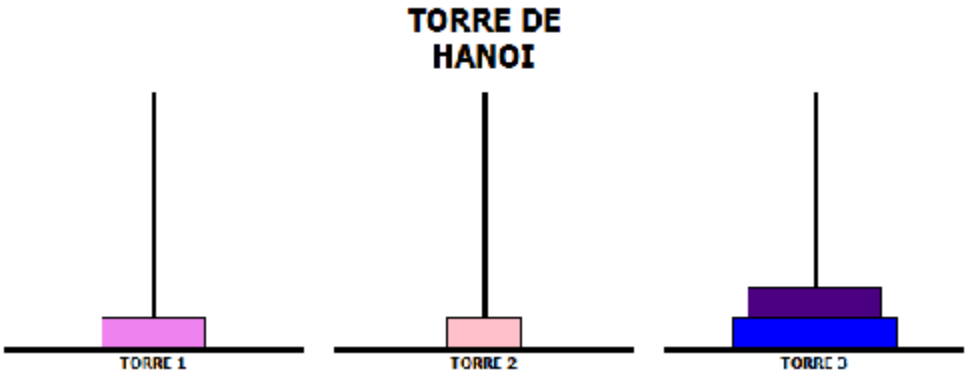


Movimiento 11

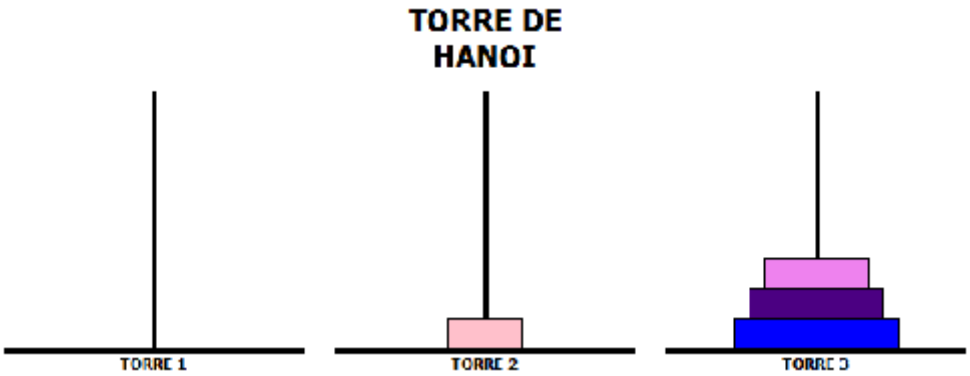
Movimiento 12



Movimiento 13



Movimiento 14



Movimiento 15 (Solución)

PISTAS DE APRENDIZAJE

Traer a la memoria:

Tenga en cuenta que los juegos lógicos deben ser vistos como algo divertido.

Tenga cuidado al realizar o desarrollar cada juego seguir los pasos indicados.

No olvide leer y entender en consiste cada juego lúdico antes de sentarse a realizarlos o jugarlos.

Tenga presente que con los juegos lúdicos se pretende desarrollar habilidades de pensamiento para utilizarla en el desarrollo de software.

Ejercicio de Entrenamiento

Mostrar el proceso que se realizo para llegar a la respuesta.

1. Para pasar de hexadecimal a binario

a) 86BF Solución: 1000011010111111

b) 2D5E Solución: 0010110101011110

2. Para pasar de octal a decimal

a) 106 Solución: 70

b) 742 Solución: 482

3. Para pasar de decimal a octal:

a) 236 Solución: 354

b) 52746 Solución: 147012

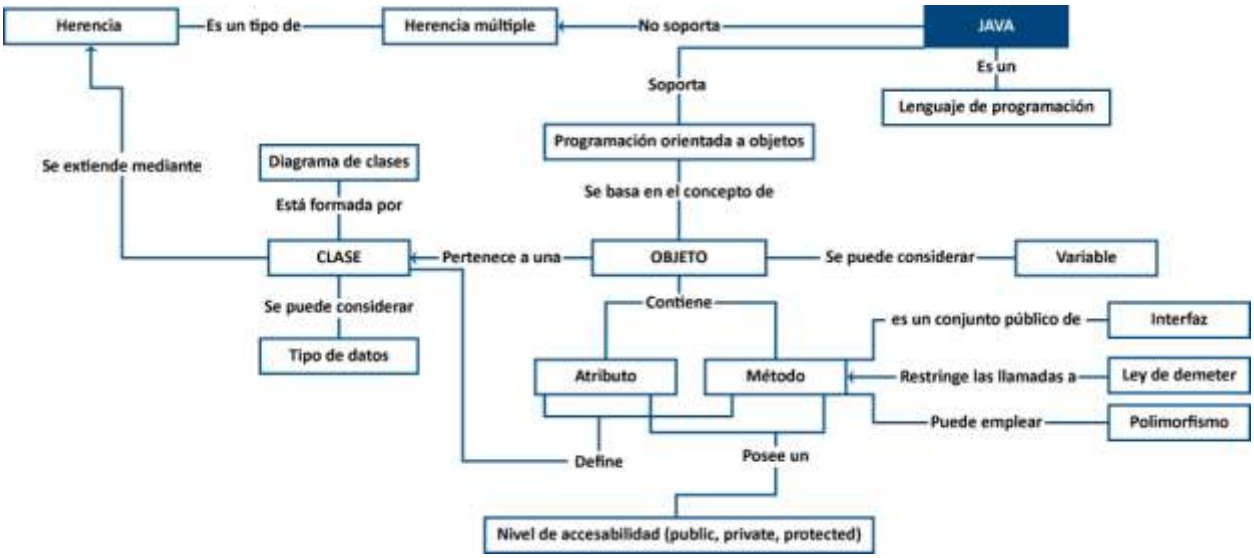
4.Un disco duro posee una capacidad de almacenamiento de 88.276,80MB y su espacio utilizado es 10,7GB con programas instalados y 1.048.231 KB en archivos de textos y en otros archivos 3.045.760.996 Bytes.
cual es la cantidad de GB libres en dicho disco?
Convierta los gigabytes libres a Mb, Tb, Kb

5. DESARROLLAR EL SIGUIENTE SUDOKU

		3				9		7
		1		5				2
	7	9	6		2	4		
5	8		7				3	
			5	2	8			
		2					5	
3	9		8	4		5		6
	6	8	2					
			3			7		

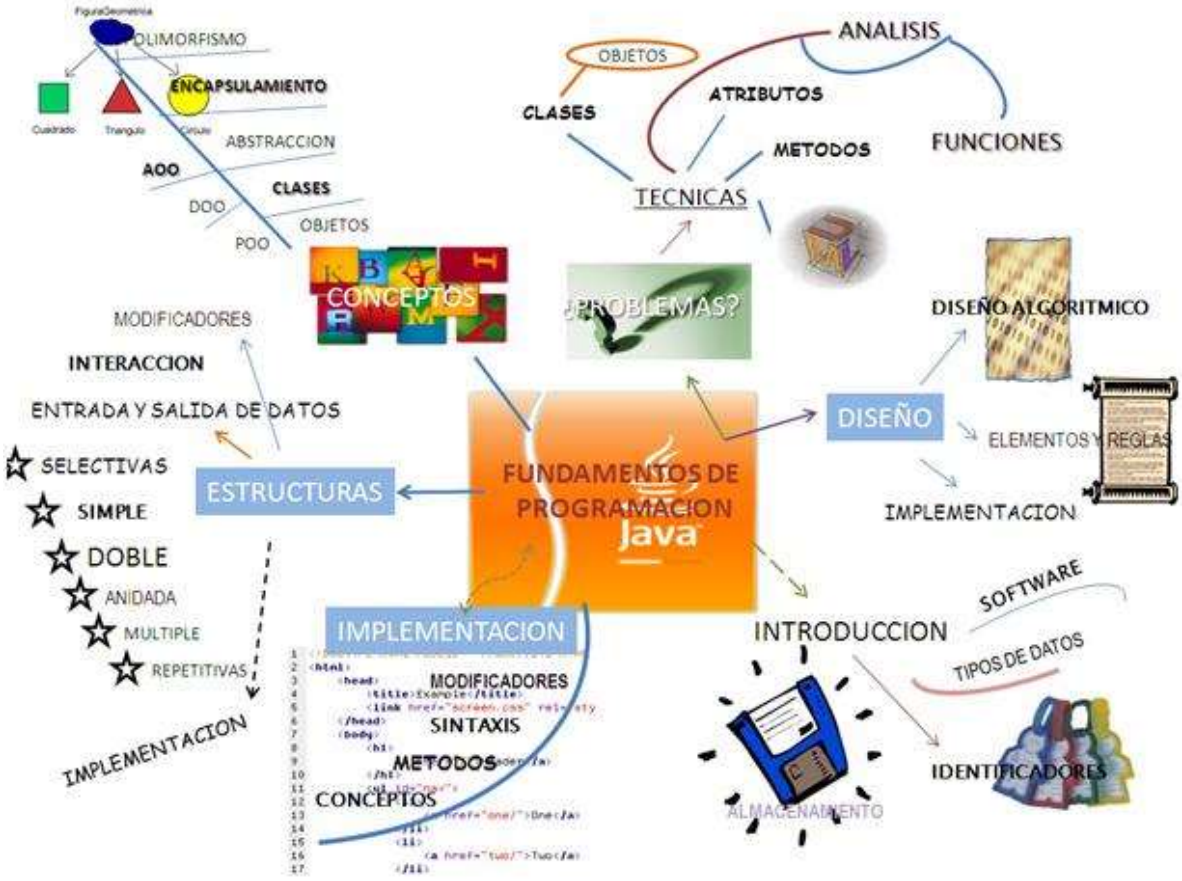
4 UNIDAD 3 PRINCIPIOS DEL DESARROLLO DE SOFTWARE

Mapa 1 de perspectiva 1-Unidad 3



Ihmc, T. (2014). Mapa Conceptual Programación Orientada a Objetos [imagen]. Recuperado de http://cmapspublic.ihmc.us/rid=1191521763968_1644716976_9540/Programaci%C3%B3n%20orientada%20a%20objetos.cmap

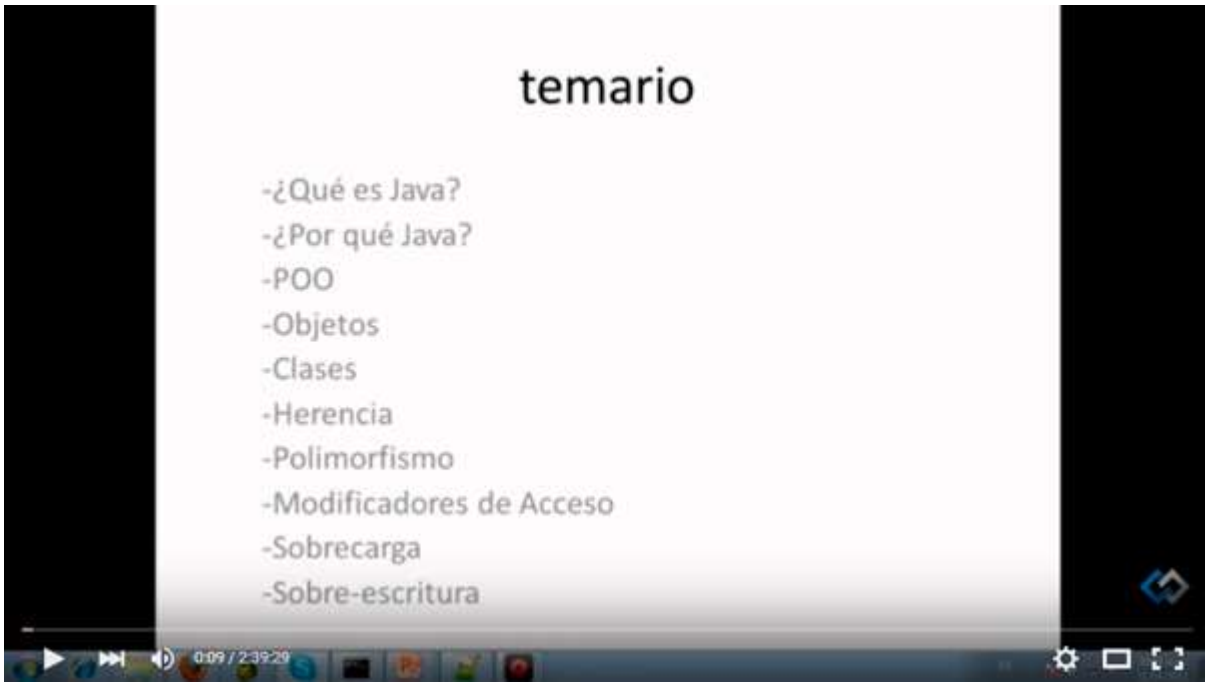
Mapa 2 de perspectiva 2-Unidad 3



Jabry, (2014). Mapa mental conceptual que explique la materia de fundamentos de programación [imagen]. Recuperado de http://users6.nofeehost.com/kaos07/examen1.html

Uml Diagramas de clases, Universidad de los Andes por Demian Gutierrez: Explica cómo trabajar con Uml yn sus diferentes diagramas Enlace

Características de los Lenguajes de Programación, Instituto Tecnológico superior de Lerdo por María Guadalupe Flores Luevano. Enlace



Define las características de los lenguajes orientados a objetos y su implementación en Java por James Gosling Enlace

http://www.adictosaltrabajo.com/tutoriales/instalacion-jvm/

VIDEO PARA INSTALAR jdk PARA NRTBEANS https://www.youtube.com/watch?v=asQl7ZNQtsw

La asignatura de Introducción al Desarrollo de Software en la carrera de Ingeniería de sistemas de la UNIREMINGTON desde este nivel inicial pretende enfatizar la conceptualización de los diferentes componentes del lenguaje de programación como herramienta fortalecedora para su implementación, esta unidad contendrá la instalación y configuración, los elementos y estructuras básico, como estructuras para la elaboración de la aplicación, algunos diagramas básicos del modelado como también se realizara definiciones entradas procesos y salidas propias del análisis para la solución de aplicaciones complejas o no mediante la codificación.

4.1 CONCEPTOS BÁSICOS DEL LENGUAJE DE PROGRAMACIÓN ORIENTADO A OBJETOS, INSTALACIÓN Y CONFIGURACIÓN DE SU ENTORNO.

Al iniciar en un lenguaje de programación sin tener ninguna orientación se hace difícil pero reflexionando se observa que es importante la abstracción para separar el conjunto de dificultades y empezar por uno que se considera elementos básicos tratando de minimizarlo para con la constancia y dedicación alcanzar un peldaño y preparase para los siguientes porque sigue siendo más difícil empezar para poder sostenerse más si se tiene el deseo y la guía .

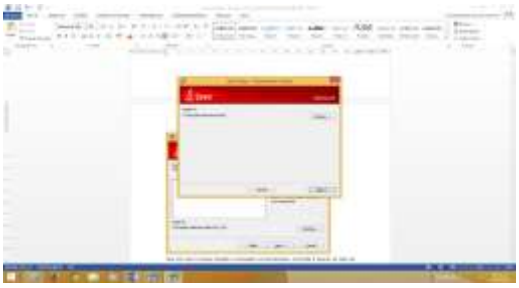
4.1.1 INSTALACIÓN, CONFIGURACIÓN Y MANEJO DEL ENTORNO

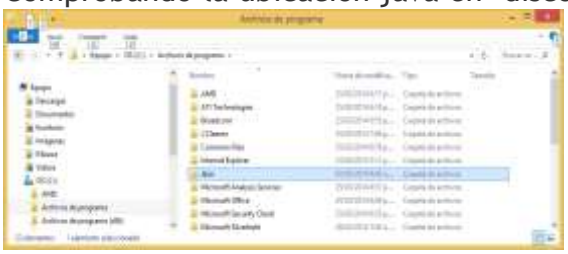
Describir los conceptos básicos del lenguaje de programación orientado a objetos requiere preparar el computador dado el caso de no estar dispuesto para el lenguaje de programación especificado como java en ese caso necesitara instalar la máquina virtual y luego el conjunto de desarrollo de java jdk en cualquier caso es una buena práctica para el principiante en desarrollo de software en , su instalación, configuración y manejo del entorno.

4.1.1.1 INSTALACIÓN Y CONFIGURACIÓN

Para realizar la instalación del El jdk

Lo primero es ir a la página oficial de sun (http://www.java.sun.com) para descargar la última versión del jdk disponible, ir a la opción “Technologies” — “Java SE” y seleccionar la pestaña “Download”, con lo que nos aparecerá una pantalla donde se especifica las características de seadas también se puede bajar del sitio web oficial: Java SE Development Kit 7 Downloads según el siguiente cuadro donde se describen algunas evidencias de pasos realizables al efectuar este procedimiento :

 Seleccionar la descarga desde el sitio.	 Inicializar el proceso según el procesador
 Tomar todas las herramientas Jdk	 Aceptar condiciones y ubicar en la unidad

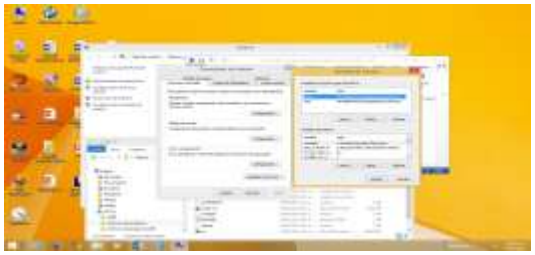
 Terminar el proceso de instalación.	Comprobando la ubicación java en disco. 
 Especificando la versión respectiva.	 Describiendo las diferentes carpetas(bin)

Posteriormente Una vez que lo hayan bajado e instalado correctamente, se procede a buscar la ruta de instalación significativamente se ubica la carpeta de los archivos básicos de java ellos se alojan en la carpeta bin por ejemplo: C:\Program Files\Java\jdk1.8.0_05\bin para luego llevarla al class path C:\Program Files\Java\jdk1.8.0_05\src.zip en el CLASSPATH

C:\Program Files\Java\jdk1.8.0_05\ para el JAVA_HOME

4.1.1.2 MANEJO DEL ENTORNO

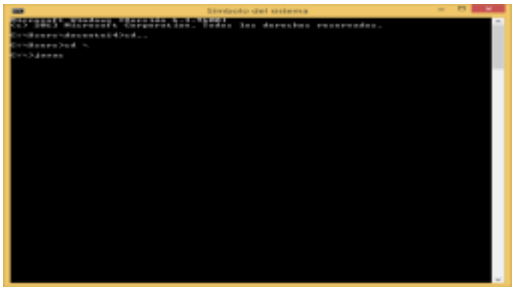
Tener en cuenta la variable de entorno porque es la ruta que tendremos que agregar a la variable de entorno Path, para ello debemos ir a: Equipo -> Click derecho -> Propiedades -> Configuración Avanzada del Sistema -> [Pestaña] Opciones Avanzadas -> Variables de Entorno llegamos al cuadro mostrado la variable de nombre Path, presionamos Editar, en el cuadro mostrado, nos vamos hasta el final del texto que ya está escrito y agregamos lo siguiente sin las comillas: "C:\Program Files\Java\jdk1.7.0_02\bin;" como se describe en la siguiente cuadro numero con algunas evidencias imágenes del proceso realizable.:

Una vez que se haya añadido la ruta, presionar aceptar y se actualiza o registra el proceso.

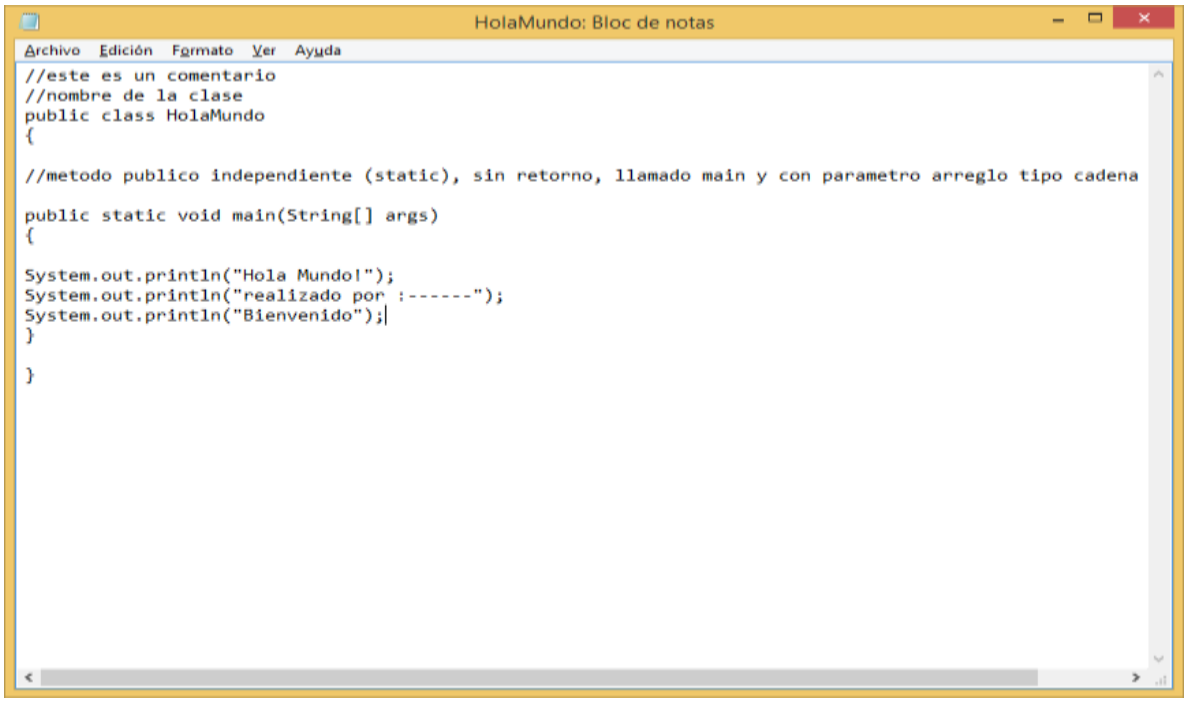
4.1.1.3 LÍNEA DE CONSOLA

La consola (cmd) primero se ubica en la ventana de consola procediendo así como en la tabla siguiente:

 Para llegar a la consola digita cmd	 Desde la consola ubicamos la unidad c:
 Para compilar escribimos javac	 Desde consola, nos debería salir así:

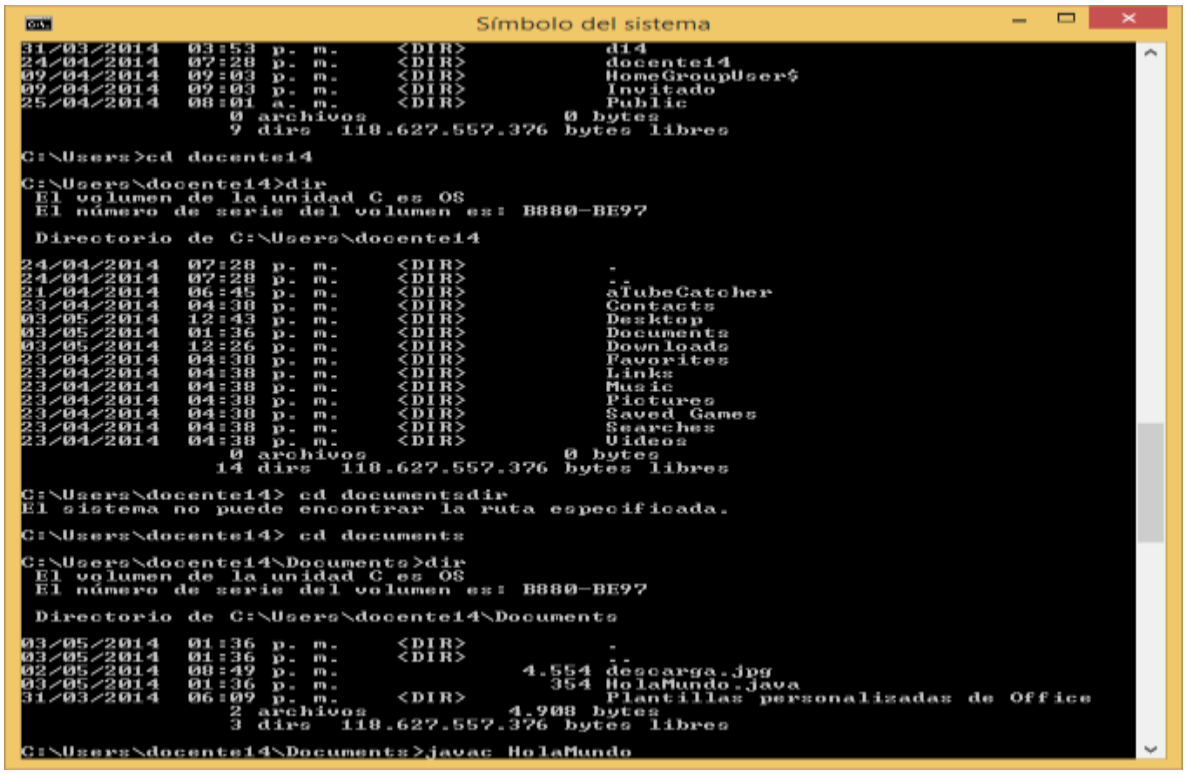
La orden para compilar es javac , esto comprueba el proceso de instalación y configuración ajustado para manejar la aplicación desde java.

Para crear una aplicación llamada HolaMundo, se requiere de un editor o se abre el bloc de notas (para digitar el programar en Java), con o sin clase pero empleando el método principal código fuente.

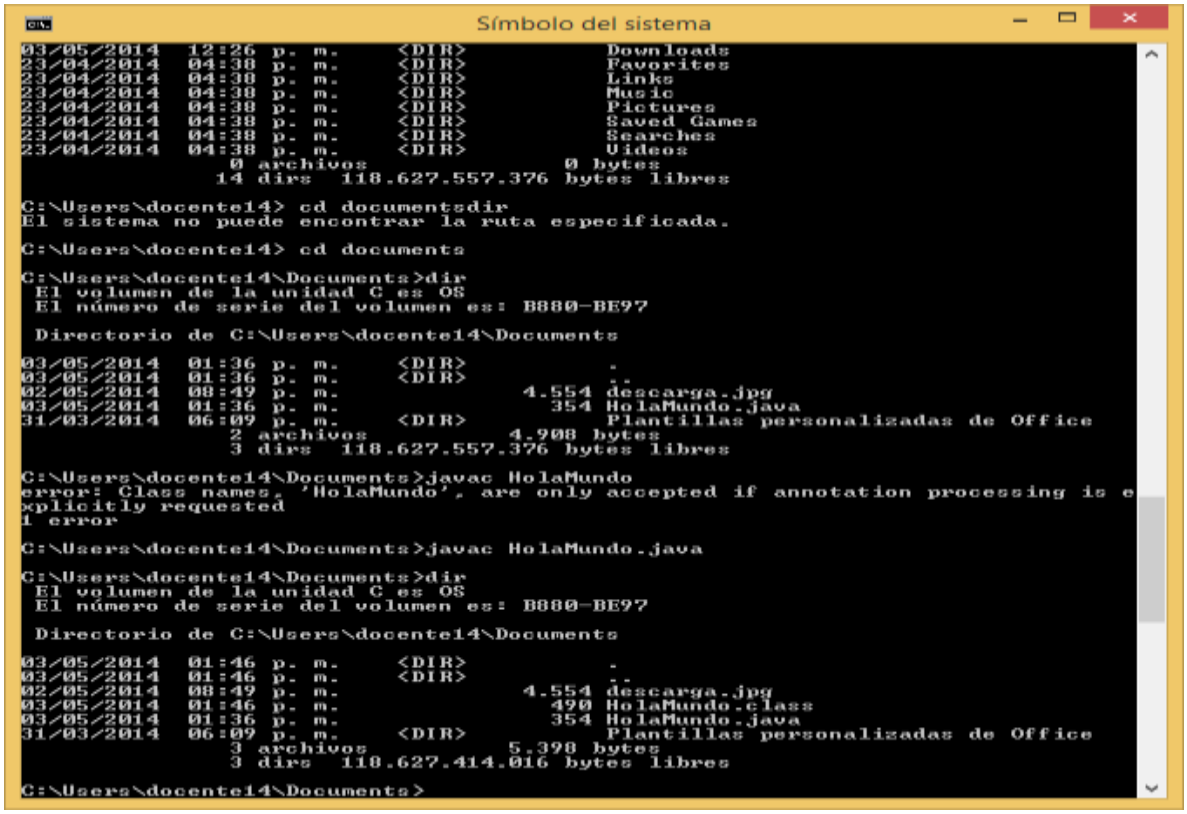


Se guarda el archivo con el nombre "HolaMundo.java", donde java es la extensión del programa fuente debe tener presente la diferencia entre mayúscula y minúscula, en este caso se cambió de ruta, por lo general está en la unidad c:

Para compilarlo verificar la sintaxis y llevarlo a lenguaje de maquina (objeto) bytecode, se digita la siguiente sentencia: ruta\ javac HolaMundo.java mucha atención con la ruta del archivo, como se especifica en el siguiente gráfico.



La compilación correcta genera un archivo con extensión .class. Verificar ….



Ejecutar poner a correr el código objeto, en la Consola escribimos: java HolaMundo

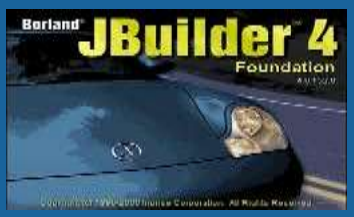
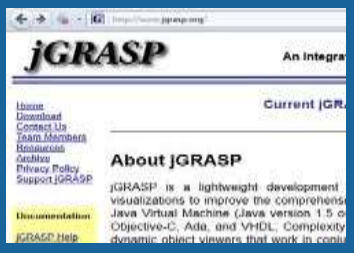
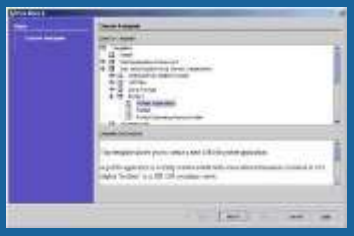
Muy importante, no se coloca la extensión del archivo (es decir el .class)

Tomado de los siguientes link: para mayor información
http://gl-epn-programacion-ii.blogspot.com/2013/01/como-ejecutar-java-desde-la-consola-cmd.html.

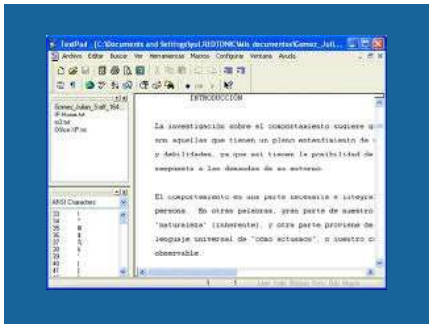
4.1.1.4 ENTORNO DE DESARROLLO

Para la elaboración de una aplicación se presentan dos posibilidades una por comando descrita en el tema anterior y por editores será una muestra de algunos editores de java pero en la UNIREMINGTON se tiene en cuenta a Eclipse y en unos casos a netbeans se describen para su apreciación otros que existen en el mercado en el siguiente cuadro

www.eclipse.org El entorno de desarrollo Eclipse puede usarse para desarrollar código en cualquier lenguaje de programación. Puede descargar el entorno y varios complementos (plug-ins) de Java para desarrollar sus programas en Java.	
www.netbeans.org El IDE NetBeans. Una de las herramientas de desarrollo para Java más populares, de distribución gratuita.	

borland.com/products/downloads/ download_jbuilder.html Borland ofrece una versión Foundation Edition gratuita de su popular IDE JBuilder para Java. Este sitio también ofrece versiones de prueba de 30 días de las ediciones Enterprise y Developer.	
www.blueJ.org BlueJ: una herramienta gratuita diseñada para ayudar a enseñar Java orientado a objetos a los programadores novatos.	
www.jgrasp.org Descargas, documentación y tutoriales sobre jGRASP. Esta herramienta muestra representaciones visuales de programas en Java, para ayudar a su comprensión.	
www.jedit.org jEdit: un editor de texto escrito en Java.	
developers.sun.com/prodtech/javatools/ jsenterprise/index.jsp El IDE Sun Java Studio Enterprise: la versión mejorada de NetBeans de Sun Microsystems.	
www.jcreator.com JCreator: un IDE popular para Java. JCreator Lite Edition está disponible como descarga gratuita. También está disponible una versión de prueba de 30 días de JCreator Pro Edition.	

www.textpad.com TextPad: compile, edite y ejecute sus programas en Java desde este editor, que proporciona coloreo de sintaxis y una interfaz fácil de usar.



Tomado de: http://javapucp.blogspot.com/2012/03/lista-de-editores-y-entornos-de.html

La descarga editor eclipse desde www.eclipse.org procesador de 32 o 64 se o ee.

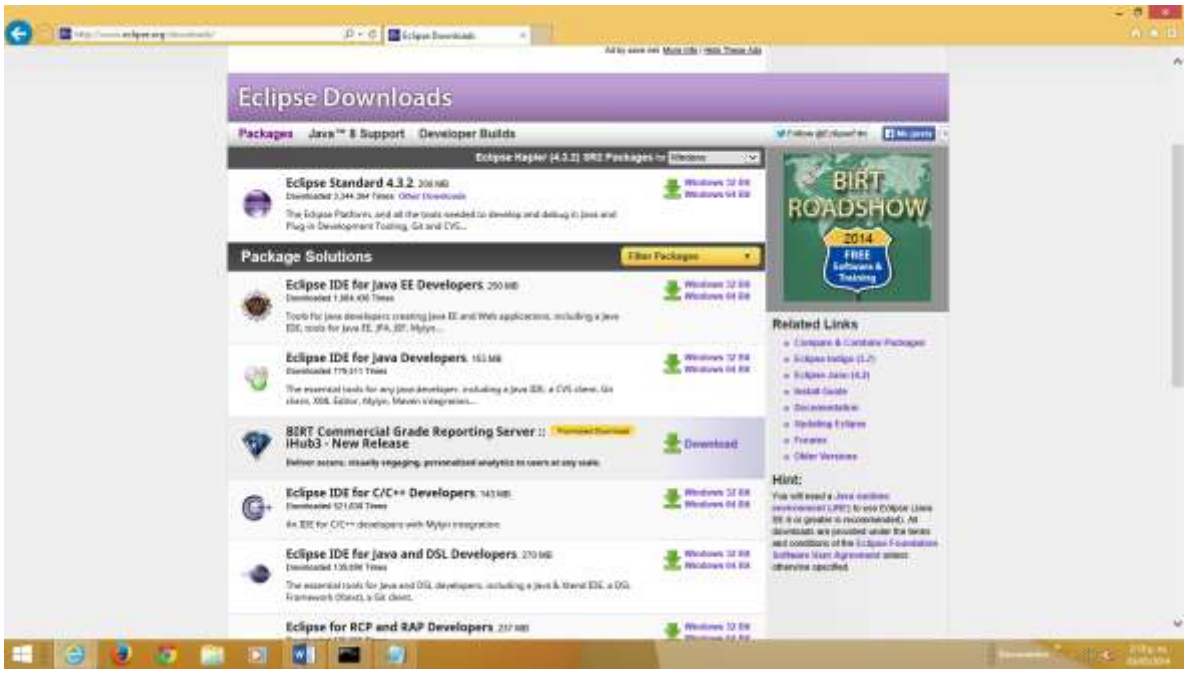
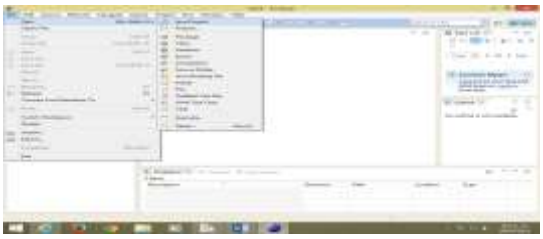
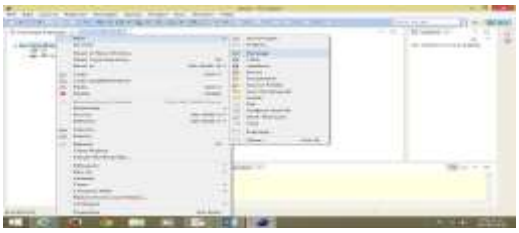
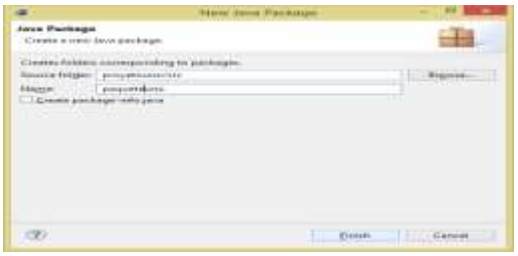
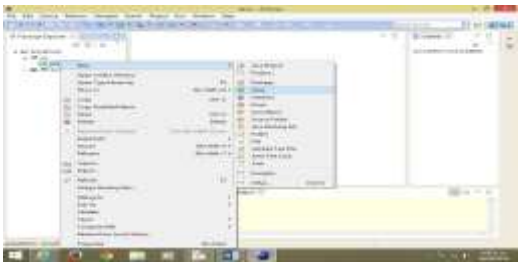
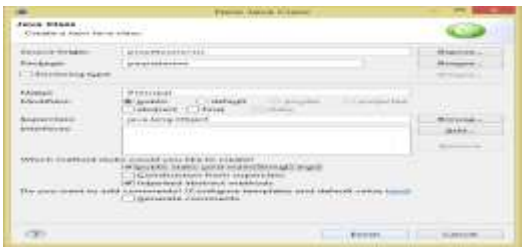
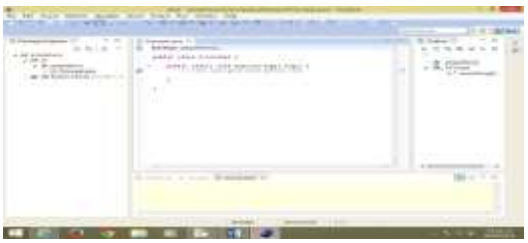


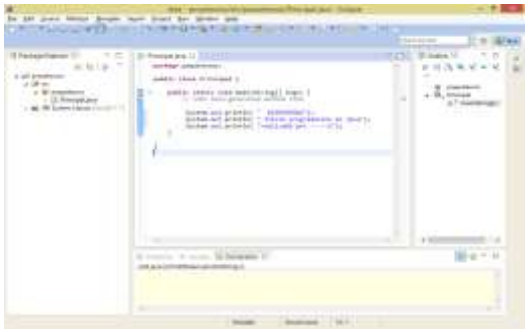
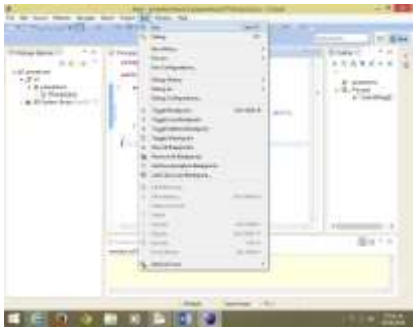
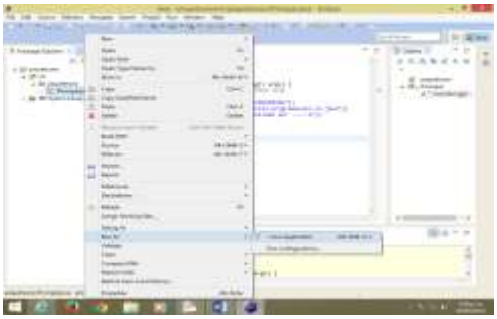
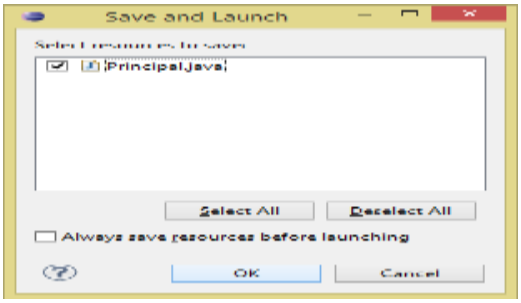
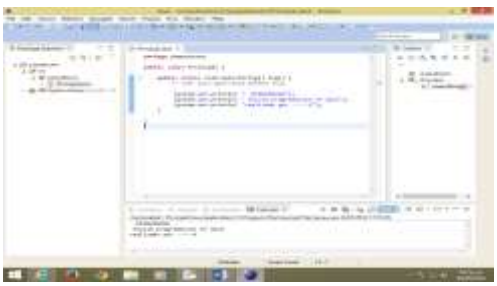
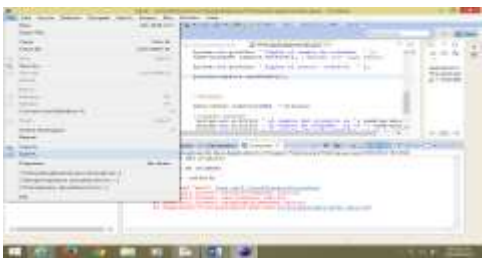
Imagen tomada de: http://www.softonic.com/s/editor-compilador-java-en-espa%C3%B1ol

A partir del editor facilitador del desarrollador para la elaboración del código se debe tener en cuenta para este caso el editor eclipse SE en cualquiera de sus categorías este se describe en el siguiente cuadro hasta Seleccionar java SE (estándar) también existe el EE (Enterprise Edition), tomemos el estándar.

Seleccionar el editor	Eclipse Kepler
 Presenta la Ventana de eclipse keppler	 muestra la ventana para crear el proyecto
 Definir el nombre del proyecto y terminar	 Desde el proyecto obtener una carpeta paquete
 Darle un nombre al paquete y terminar	 Desde el paquete botón derecho nueva clase
 Nombrar la clase(preferiblemente mayuscula)	 Lista la estructura para codificar en java

Seleccionar de la barra de menú, el menú de archivo (file), por medio del comando nuevo la etiqueta proyecto java (project name) muestra la ventana para crear el proyecto como lo describe la tabla en la segunda fila segunda columna. Se debe nombrar el proyecto y terminar el proceso para continuar con el paquete que es el agrupador de los diferentes archivos de este proyecto, clic derecho en él y el comando nuevo pero seleccionar paquete se crea uno debe finalizar el proceso lo indica el editor. Sobre el paquete, clic derecho se pide un nuevo archivo tipo clase a la cual se le asigna un nombre preferiblemente primera mayúscula, este presenta algunas opciones para sus métodos en especial si se requiere el main. (Principal).

Toda su ventana de edición se observa en 4 divisiones como lo describe la imagen uno en la tabla siguiente

 Ventana de la aplicación java procedimental	 Desde la línea de comando RUN correr
 Especificar la aplicación a ejecutar	 Describir los archivos en este caso uno
	

No presenta errores solo los resultados	Los errores se refleja en la ventana inferior en rojo, corregir y de nuevo

4.2 ESTRUCTURAS BÁSICAS PARA LA ELABORAR UN PROGRAMA EN LENGUAJE JAVA.

Describir las diferentes estructuras básicas para la elaboración de una aplicación en lenguaje de programación orientados a Objeto (java) por medio de proyectos, empleando adecuadamente el desarrollar software, con sus tipos de datos, sus expresiones, su forma de acceso a datos, las estructuras de toma de decisiones, los ciclos y desde luego la incursión en la elaboración de aplicación.

4.2.1 TIPOS DE DATOS Y EXPRESIONES

Este tema se relaciona con la tipificación ligada a la abstracción de los datos y sus procedimientos por ende implican sobre la capacidad de almacenamiento en memoria, para java se disponen de dos tipos de datos básicamente

Antes de identificar cada uno de los elementos miembros datos y procedimientos incluso los independientes es importante el reconocimiento del espacio en memoria a utilizar

4.2.1.1 TIPOS DE DATOS

Los tipos de datos nativos en otro caso llamados primitivos se caracterizan por ser tradicionales en la programación de los lenguajes no son objetuales, pero en el desarrollo de la metodología objetual cumplen un papel muy importante se encargan de realizar las acciones aritméticas y combinan sus funciones con las clases numéricas determinando el uso de memoria disponible para cada uno. A continuación en el cuadro número --- se describirán algunos de los tipos nativos con sus capacidades respectivas

Tipos de dato primitivo	Descripción	Capacidad
byte	Numérico entero manejador de 8 bits.	Rango desde -27 hasta 26 (-128 a 127)
short	Numérico entero manejador de 16 bits	Rango desde -215 hasta 214 (-32768 a 32767)

int	Numérico entero manejador de 32 bits.	Rango desde -231 hasta 230 (-2147483648 a 2147483647)
long	Numérico entero manejador de64 bits.	rango desde -263 hasta 262 (-9223372036854775808 a 9223372036854775807)
float	Numérico con decimal de 32 bits. coma flotante de simple precisión.	rango desde (de 1.40239846e–45f a 3.40282347e+38f)
double	Numérico con decimal de 64 bits. coma flotante de doble precisión.	Rango desde (de 4.94065645841246544e–324d a 1.7976931348623157e+308d.)
char (‘ ‘)	Carácter Unicode de 16 bits	
String (“ “)	Cadena de caracteres	
boolean	Tiene un valor de comprobación	Rango true o false.

Es importante identificar cada uno de estos tipos con el primer carácter en minúscula diferente a las clases numéricas

4.2.1.2 TIPOS DE DATOS CLASE

Los tipos de datos clase se caracterizan por pertenecer a la una de los agrupadores de clase y por tanto adquieren procedimientos los cuales les facilitan la conversión a cualquiera de los tipos de datos nativos, combinándose adecuadamente y sobre todo teniendo presente que el flujo de datos se presenta en tipo de dato Cadena o texto o alfanumérico dentro de la clase String (note la primera mayúscula). es notable la clasificación que se establece diferentes tipos de paquetes los cuales agrupan clases que serán utilizadas en el desarrollo de la aplicación según las necesidades, existe un paquete predefinido donde las clase se usan sin necesidad de importar el paquete que está dentro de java este es lang, la sentencia package debe estar en primera línea de la cabeza del programa digitado por el desarrollador con la sintaxis import java.lang.*; realizando una petición a todas las clase encontrados en este paquete o import java.lang.System; solicitando el uso de la clase System, Object, Math entre otros que pertenecen

al paquete lang . Algunos agrupadores se describen en el siguiente cuadro número 2 Tipos y características de los paquetes

PAQUETE	DESCRIPCIÓN
lang	Se asigna sin necesidad de nombrarla agrupa clases como System, Object, Math Object, Thread, Exception, System, Integer, Float, Character, String, Double
io	Se encarga de agrupar las clases del flujo de datos entrada y salida desde el programa clase como InputStreamReader, Exception, BuferedReader. FileInputStream y FileOutputStream, InputStreanReader
util	Agrupas clase de utilería como la clase Date, Random, Dictionary, Stack .
applet	Agrupa las clases para manejar el explorador tiene la clase Applet AppleViewear,AppletContext
java.net	Agrupa las clases para manejar la comunicación redes clase Sokect, URL y URLConnection.
awt	Agrupas las clase propias para el manejo de ambiente grafico tradicionales Buttom, Frame, Panel,Menu, Label,TextField
swing	Agrupas las clase propias para el manejo de ambiente grafico más actualizadas se le antepone la J a cada componente import javax.swing .JButtom , JFrame, JPanel,JMenu, JButton, JLabel, JTextField
bean	Agrupa las clase propias para generación de componentes reutilizables
sql	Agrupa las clase para manejo de base de datos

Cuadro numero tema Descripción de algunos paquetes con sus características (realizado por el autor)

4.2.1.3 DIFERENCIA ENTRE TIPOS

Una diferencia notable es los nativos son tradicionales no clases y los otros son clases poseedores de propiedades y procedimientos.

Los nativos se declaran inicializan asignándole un valor y se utilizan, los clases de declaran instancian requieren del operador new para solicitar espacio en memoria y del método constructor respectivo para inicializarlo y se usan están agrupados siendo la clase String la llamada intermediaria para el acceso a la información y luego se convierte al nativo respectivo acompañado de una clase numérica facilitadora del método en el cuadro siguiente se presentaran algunos tipos de datos clase también llamados envoltorio por pertenecer a las clases.

Tipos de dato Clase	Descripción	procedimiento
Byte	Numérico entero	
Short	Numérico entero	
Long	Numérico entero	
Integer	Numérico entero	
Double	Numérico con punto flotante	
Float	Numérico n punto flotante	
Boolean	Verdadero o falso	
Character	Un solo carácter	
String	Uno o más caracteres “ ”	

Existe la posibilidad de definir unos tipos de datos especiales los cuales son creados por el programador pero están dentro de esta estructura como son las class y las bibliotecas.

4.2.2 EXPRESIONES DELIMITADORES SEPARADORES Y OPERADORES

Contando con la funcionalidad de los caracteres ASCII en el lenguaje de programación en donde la gran mayoría cumple una en este caso de delimitar o separar para conformar la estructura del código se resumen en el siguiente cuadro algunos de los delimitadores o separadores con sus respectiva descripción.

Carácter tipo delimitador separador	Descripción
{ }:	Llaves { }: abierta delimita el inicio principio del bloque de código. cerrada delimita el final del bloque de código.
[ ]:	Corchetes [ ]: juntos abierto y cerrado especifican la posición del arreglo tipo vector.
( ):	Paréntesis (): juntos delimitan la estructura de un método con su lista de parámetros si existen o como condicionales en pregunta, además en expresiones establecen la prioridad y precedencia.
// o /*__ */	Establecen el comentario o documentación en el código. // comentario de una sola línea /* delimita principio de comentario */ finaliza el comentario
;	punto y coma ; delimita la instrucciones indica el final de ella
,	coma ,: delimita cada identificador en la instrucción
.	Punto .: agrupador a los atributos y métodos de una clase.

+	Suma +:agrupador Concatena cadenas variables y/o constantes
\n	Salto de línea
\t	Espaciador en un texto

Operador es otra particularidad de algunos caracteres ascii para realizar acción facilitando la conformación de una expresión al combinarse con operando quien puede ser variable o constante preferiblemente numérica. La diversidad de estos elementos los clasifica de la siguiente manera: aritméticos, relacionales, lógicos, de Asignación, unarios, de Instanceof, concatenado, de nivel de bit o desplazamiento entre otros para resumir se describirá cada grupo en el siguiente cuadro:

Operadores Aritméticos	descripción	ejemplo
+	Suma	A =A+1;
- .	Resta	A =A-1;
* .	Multiplicación.	A =A*1;
/ .	División.	A =A/1;
%	Resto de la División (modulo) .	A =A%1;
Operadores Asignación:	**Combina el = con los aritméticos**	
=		: A = 65
+=		: A += B ; A = A + B
-=		: A -= B ; A = A - B

*= /=' %=		: A *= B ; A = A * B : A /= B ; A = A / B : A %= B ; A = A % B
Operadores Unarios	**Cambiar el signo del operando –o+.**	
++ --	Incrementa la variable una unidad. decrementa en una	A++ A--
Operadores Relacionales:	**Realizan comparaciones validan**	**A comparado con B es:**
> < == ¡= >= <=	Mayor que Menor que Iguales Distintos Mayor o igual que Menor o igual que	Verdadero falso
Operadores Lógicos:	**Realizan comparaciones lógicas.**	**A : con B devuelve**
&& || ! & |	si ambos operandos son true. si alguno de los operandos es true. negación del operando si cada uno es true, es un and si al menos uno es true, es un or	true true ! (true) sera false true true

Operadores nivel de bits: >>' <<	Realizan desplazamiento a nivel de bit: hacia la derecha hacia la izquierda	

4.2.2.1 IDENTIFICADOR

Es importante el reconocimiento de cada uno de los elementos empleados en una aplicación por tanto la identificación de cada clase, miembro dato o procedimiento incluso paqueteo nombre de archivo está ligado a la estructura sintáctica del lenguaje y su aplicación en el espacio tiempo, ya se definieron los tipos de datos que se anteponen a cada identificador este preferiblemente debe iniciar con carácter letra, es independiente de mayúscula o minúscula hasta puede contener carácter digito no debe llevar caracteres especiales ni definir lo como una palabra reservada

Las palabras reservadas son identificadores predefinidos que tienen un significado para el compilador y por tanto no pueden usarse como identificadores creados por el usuario en los programas.

Algunas palabras reservadas en Java se presentan a continuación en el cuadro siguientes para tener presente al definir los identificadores:

abstract	continue	for	new	switch	assert	default	package	synchronized
boolean	o	if	private	this	break	double	implements	protected
throw	byte	else	import	public	throws	case	enum	instanceof
return	transient	requiera	extends	int	short	char	final	interface
static	void	class	finally	long	float	super	while	try/ catch

4.2.3 ESTRUCTURAS BÁSICAS

Una estructura de validación sea pregunta o selector múltiple presenta por medio de la sintaxis la forma correcta de realizar un condicional esta forma de preguntar puede ser simple compuesta o anidada con el fin de facilitar el control

4.2.3.1 ESTRUCTURAS CONDICIONALES

Pregunta con Bifurcación: se emplea el método if(condicional)-else, preguntas anidadas una dentro de otra y pregunta compuesta donde se combina los operadores relacionales y los lógicos en el condicional el manejo de los delimitadores {} y () son de gran utilidad para determinar el alcance y la prioridad de la pregunta.

Pregunta	Estructura	Ejemplo

bifurcacion	if(condicion) {
	instruccion1();
	} else {
	instruccion1();
	instruccion2();
	}
		if (A>=B) {
		System.out.println("mayor o igual) ;} else {
		System.out.println("No es mayor)}

anidada	if(condicion1) {
	instruccion11;
	if(condicion2) {
	instruccion22;
	instruccion22;
	} else {
	Instruccion22;
	}.
	}
	else {
	instruccion1;
	instruccion2;
		if (A>B) {
		System.out.println("mayor") ;
		if (A==B) {
		System.out.println("MayorIgual") ; } else
		{
		System.out.println("MayorIgual") ;}
		}
		else
		{
		System.out.println("No es mayor)}

	}.	

compuesta	if(condicion1 &&condición 2) {
	instruccion1();
	} else {
	instruccion1();
	}.
		if ((A>B)&&(A==B)) { System.out.println(“mayor o igual) ;} else { System.out.println(“No es mayor=)}

Para el selector múltiple el condicional se hace más práctico especialmente si en el condicional se evalúan secuencias preferiblemente numéricas para lo cual se emplea el método swich() dentro se establece el caso para seleccionar la condición elegida, cada opción del caso debe establecer su terminación para ello emplea la palabra reservada break; que cierra la secuencia de esta por ultimo cuenta con un control llamado default para especificar cuando no se ha seleccionado una alternativa correcta o en el rango establecido

Estructura del selector multiple	Ejemplo tipo numérico y cadena	

switch(expresión numérica) {
case 1: instrucción caso 1;
break;
case 2: instrucción caso 2;
break;
// si existen más se define cada caso y se cierra
default:
	switch(estadocivil) {
	case 1:
	System.out.println(“soltero”)};
	break;
	case 2:
	System.out.println(“casado”)}
	break;
	case 3:
	System.out.println(“separado”)}
		switch(estadocivil) {
		case “s”:
		System.out.println(“soltero”)};
		break;
		case “c”:
		System.out.println(“casado”)}
		break;
		case “S”:
		System.out.println(“separado”)}

instrucción de control de secuencia y mensaje ; }	break; case 4: System.out.println(“otro”)} break; **default:** System.out.println(“error”)}	break; case “o”: System.out.println(“otro”)} break; **default:** System.out.println(“error”)}

4.2.3.1 ESTRUCTURAS REPETITIVAS: PARA, MIENTRAS QUE Y HAGA MIENTRAS QUE

El proceso al tener más de una repetición se considera ciclo, la posibilidad de ser cualitativo o cuantitativo depende de si se tiene o no el número de veces a realizar el o las actividades en ese caso será ciclo para for() es un método con tres parámetros tipo numérico entero el primero inicia la variable controladora del ciclo, el segundo se encarga de validar el condicional hasta finalizar y el tercero actualiza o incrementa ascendente o descendentemente la variable controladora o cuantitativo , ciclo mientras : while () o cualitativo al igual al haga mientras que : do while (), entre este y el mientras la diferencia es la manera de ingresar y preguntar este mientras que pregunta antes de entrar al ciclo, el haga mientras que entra primero al ciclo luego para salir pregunta es de notar que se realizan quiebres o rompimientos dentro del ciclo por medio de break o continúe que evalúa la condicional y posibilita la continuidad se presenta un cuadro con los tipos de ciclo su sintaxis y ejemplo

ciclo	Sintaxis	ejemplo

para	**for**(iniciación; condición; incremento)
	{
	actividades;
	}
		for(int j = 1; j <= 5; j++)
		{
		System.out.println(j+",");
		}

Mientras que	Inicializar variable controlador while(condición) { actividades;	int j = 1 while(j <= 5) { System.out.println(j+",");

	actualizar variable controlador }	j=j+1; }
Haga mientras que	Inicializar variable controlador do { actividades; actualizar variable controlador **} while(condición);**	int j = 1 do { System.out.println(j+","); j=j+1; } **while(j < 5);**

4.2.3.2 ESTRUCTURA DE LA CLASE (CLASE, MÉTODO)

La implementación en un lenguaje de programación está ligado a la metodología puede ser estructurada u objetual las más comunes para nuestro caso es importante establecer la estructura correcta para describir sus elementos logrando no presentar errores de forma, primero se define un proyecto con su respectivo identificador y ubicación en un dispositivo (pc o usb) dentro del uno o más paquetes con su identificación respectiva con el objetivo de agrupar y clasificar la o las clase(s) con sus atributos y métodos se identifica con Primera mayúscula en el nombre para reconocimiento estándar, también entran otros archivos como las importaciones respectivas adicionales a conformaran la aplicación. Se tomara la clase otro elemento fundamental de la orientación a objeto de la cual se describen sus elementos tanto para una clase sin método principal como para una con método principal encargado de la ejecución de la aplicación en el siguiente cuadro:

Estructura del proyecto	Estructura de la clase sin método principal	Estructura de la clase con método principal

Paquete
 	
import java.util.*;
 	
public class nombredeclase
 	
{
 	
// lista de atributos con su tipo
// lista de métodos
}
	Paquete
	
	Importaciones
	
	public class nombredeclase
	
	{
	
	// lista de atributos con su tipo
	private String nombre;
	private int edad;
		Paquete
		
		import java.util.*;
		
		public class Principal {
		public static void main(String[] args) {
		
		// lista de variables locales
		//lista de objetos si existen
		desarrollo de actividades con un menu de actividades o con interfaz

	// métodos
	public nombredeclase(){}
	public void mostrarnombr(){}
	public String capturanombr(){}
	public String getNombre(){}
	public void setNombre(--){}
	}
		}
		public static int captura(String mensaje){}
		
		public static void mostrar(String mensaje){
		
		System.*out*.println(""+mensaje);
		}
		}

Hablar de métodos implica la modularidad otro elemento de los fundamentales de la metodología orientada a objetos donde se especifica la acción pero para su forma se compone de seis partes cada una cumple una función la primera el alcance puede ser público o privado método que solo se ra visto por los elementos de la clase , la segunda el tipo de clase describe si es estático sea método independiente de la clase en otro caso es un objeto perteneciente a la clase por ende norequiere ser especificado, tercero el tipo de dato de retorno involucra la tipificación especificando si es tipo de dato nativo o tipo clase y caracteriza el método como función por tanto debe retornar un dato de igual tipo y será procedimiento sin retorno si es tipo de dato void exclusivamente , cuarto el nombre del método identificador que para el estándar debe iniciar en minúscula, quinto la lista de parámetros cada uno de ellos si existe debe tener su respetivo tipo si tiene más de uno debe separarse cada uno com (,) y por último el delimitador de principio y final de bloque en donde se registran las actividades respectivas { }>

Estructura del método	Método tipo función o procedimiento l	Método tipo static

Ambiente objeto/static tipo de dato retornado nombredelmetodo
 	
(lista de parámetros) {
 	
// actividades del procedimiento
}
	// métodos
	public nombredeclase(){}
	public void mostrarnom(){}
	public String capturanombr(){}
	public String getNombre(){}
	public void setNombre(--){}
	}
		public static void main(String[] args) {
		
		// lista de variables locales u objetos si existen , actividades con un menu de actividades o con interfaz }
		public static int captura(String mensaje){}
		public static void mostrar(String mensaje){
		
		System.*out*.println(""+mensaje);
		}}

		package paqueteuno; **public class** Principal {**public static void** main(String[] args)

```
                                    { System.out.println
                                      ( "  BIENVENIDA");

                                  System.out.println( " Inicia
                                       programacion en java");
                                System.out.println( "realizado
                                                  por -----A");
                                                       }



                                                       }
```

4.2.4 FORMA DE ACCESO A DATOS

El acceso a datos en el lenguaje de programación registra el almacenamiento en memoria el cual se realiza de forma interna por asignación no se presenta intervención del usuario durante el proceso o de forma externa esta asignación captura de los datos requiere de intervención del usuario y su el almacenamiento en memoria independientemente de su temporalidad o sea que ingreso los datos sin especificar si se llevara a un archivo o una base de datos, el acceso a datos presenta otra manera de llevarlos a un espacio en memoria este es mediante la instancia pero debe tener presente que es para los miembros datos de una clase al definir el objeto para su uso.

4.2.4.1 ASIGNACIÓN INTERNA DE DATOS STRING

Este ingreso datos constante sean numérica, no numéricos (cadena) y lógicos a la memoria del procesador es temporal y como tal solo dura mientras este la aplicación pero permite realizar los procesos sin tener que utilizar estructuras de acceso y sus respectivas clases solo se declara cada identificador con sus respectivo tipo sea miembro clase el que instanciamos y posee procedimientos internos o variable que se debe inicializar en el siguiente cuadro se representa cada una de estas actividades

Inicialización de variable	Instancia los objetos de la clase
int numero = 122;	Integer num= new Integer (“122”);
double decimal=12.54;	Double decim= new Double (“12.54”);
boolean estado= true;	Boolean estad= new Boolean (“true”);

String cadena=”dato texto”;	String caden = new String (“textos”);

4.2.4.2 FORMA DE ACCESO A DATOS SCANNER

Estructura de una aplicación en lenguaje de programación java

Sin lectura	Con lectura de datos

public class nombredelaclase
 	
{
 	
//metodo principal para iniciar o ejecutar la aplicación
 	
public static void main(String [] arg)
 	
{
 	
int base= 5 ;//declaramos la variable tipo entero,se inicaliza con valor constante asignación interna se inicializa con
 	
System.out.println(“\n la base es \t”+base);
 	
// se imprime o muestra la informacion
 	
}
	import java.util.Scanner;// importamos el paquete para utilizar la clase Scanner y leer
	
	public class nombredelaclase
	
	{
	
	//metodo principal para iniciar o ejecutar la aplicación
	
	public static void main(String [] arg)
	
	{
	
	int base ;//declaramos la variable tipo entero
	
	Scanner captura= new Scanner(System.in);
	
	System.out.println(“digite el valor para la base”);
	
	base=captura.nextInt();// se captura el valor conel metodo convertidor entero y se asigna a la base forma externa
	
	System.out.println(“\n la base es \t”+base);
	
	// se imprime o muestra la informacion

```
}
```

Aquí el método main que debe estar dentro de una clase, se encarga de ejecutar la aplicación, tiene un modificador de alcance (llamado ámbito) que es público un tipo de objeto que es static el cual especifica independencia de la clase el tipo de dato de retorno es void que no regresa ningún valor a la aplicación el nombre del método es main con un parámetro entre los () tipo arreglo cadena por los [] :

```
public class Interno01
 {
    //metodo principal para iniciar o ejecutar la aplicación

    public static void main(String [] arg)
 {
int base= 5 ;//declaramos la variable  se inicializa con valor fijo
int altura= 10; se declara y se inicializa
int areatriangulo;
 areatriangulo = base *altura;
System.out.println("\n el área del triangulo  es \t"+ areatriangulo);

}
```

Sin captura de datos asignación interna

```
public class pregunta01 {
    //metodo principal para iniciar o ejecutar la aplicación
    public static void main(String [] arg)
 {
int base= 5 ;//declaramos la variable  se inicializa con valor fijo
```

```
int altura= 10; se declara y se inicializa

int areatriangulo;

 areatriangulo = base *altura;

if ( base< areatriangulo){

System.out.println("\n La base  es inferior al  área del triángulo  ");

}

else

{

System.out.println("\n La base  No es inferior al  área del triángulo  ");

}

System.out.println("\n el área del triángulo  es \t"+ areatriangulo);


}
```

Para captura de datos se debe emplear hasta el momento una clase llamada Scanner que pertenece al paquete útil, por ser clase tiene unos métodos que le permiten convertir la información tomada desde el teclado como texto y llevarla a la variable según su tipo por medio del método correspondiente el procedimiento será:

Importar el paquete ;import java.util.Scanner; andes de definir la clase

Segundo en el método principal (main) se declara un objeto tipo Scanner y luego se instancia con el operador new y el método constructor quien toma la información del sistema para nosotros el teclado

Scanner objcaptura = new Scanner(System.in);

Se presenta un mensaje con la clase System perteneciente al paquete lang con la sub clase out (salida o muestra en pantalla) y el método imprimir System.out.println(" por favor digite"); para orientar al usuario

La captura se realiza y se aplica el método convertidor next Int para entero, nextDouble para reales y nexto nextLine para las cadenas.

base= captura.nextInt(); la variable tipo entera recibe del objeto scanner la información tipo cadena Clase pero convertida en entero nativo.

Captura de datos

```
import  java.util.Scanner;// importamos el paquete para utilizar la clase
scanner  y realizar la lectura


public class Captura01
 {
    //metodo principal para iniciar o ejecutar la aplicación

    public static void main(String [] arg)
 {
int base ;//declaramos la variable tipo entera
double altura; //se declara real
int areatriangulo;

//se prepara la clase scanner para leer
Scanner captura= new Scanner(System.in);
System.out.println("digite el valor del la base");
base=captura.nextInt();// se captura conel metodo convertidor entero

System.out.println("digite el valorpara  la altura");
altura=captura.nextDouble();// se captura conel metodo convertidor real double

//se calcula el área del triangulo
 areatriangulo = base *altura;
```

```
//pregunta simple
if ( base< areatriangulo)
{
System.out.println("\n La base  es inferior al  área del triangulo  ");
}
else
{
System.out.println("\n La base  No es inferior al  área del triangulo  ");

}
System.out.println("\n el área del triangulo  es \t"+ areatriangulo);

}
```

ciclo para

```
import  java.util.Scanner;// importamos el paquete para utilizar la clase
scanner  y realizar la lectura


public class Para01
 {
    //metodo principal para iniciar o ejecutar la aplicación

    public static void main(String [] arg)
 {
```

```
Int N;// variable numero de encuestados

int estadocivil ;//declaramos la variable

int contasolt = 0; se declara y se inicializa en cero los contadores

int contacas = 0;

int contasepa = 0;

int contaotro = 0;

int contador=0;

Scanner captura= new Scanner(System.in);// se instancia la clase scanner
para inicializar el objeto captura

System.out.println("\ndigite el número de personas a encuestar");


N=captura.nextInt();


 //se realiza la lectura del limite


for(contador=1; contador<= N)

{

System.out.println("\nSeleccione  un estado civil : 1 soltero/n 2 casado/n  3 separado/n  4  otro");

estadocivil=captura.nextInt(); //se realiza la lectura del estado civil


switch(estadocivil) {

 case 1:

System.out.println("\n estado civil soltero");

contasolt = contasolt +1;

break;

 case 2:

System.out.println("\n estado civil  casado/n");
```

```
contacas = contacas +1;


break;

case 3:

System.out.println("\n estado civil :  separado");

contasepa = contasepa +1;


break;

 case 4:

System.out.println("\n estado civil : otro");

contaotro = contaotro +1;


 break;

default:

System.out.println("\nSeleccione  solamente 1…4");

     break;

 }


}

System.out.println("\nel total de soltero es /t"+contasolt+" el total de casado/t"+ contacas+" el total de separados es /t"+ contasepa+" el total de otros es\t"+ contaotro);


 System.exit(0);

    }

}
```

Ciclo mientras

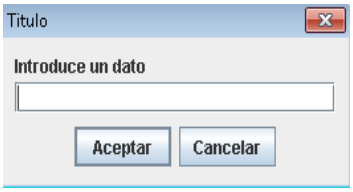
```
import  java.util.Scanner;// importamos el paquete para utilizar la clase
scanner  y realizar la lectura


public class mientras01
 {
    //metodo principal para iniciar o ejecutar la aplicación

    public static void main(String [] arg)
 {
Int N;// variable número de encuestados
int estadocivil ;//declaramos la variable
int contasolt = 0; se declara y se inicializa en cero los contadores
int contacas = 0;
int contasepa = 0;
int contaotro = 0;
int contador ;
Scanner captura= new Scanner(System.in);// se instancia la clase scanner
para inicializar el objeto captura
System.out.println("\ndigite el número de personas a encuestar");


N=captura.nextInt(); //se realiza la lectura del limite


System.out.println("\nSeleccione  un estado civil : 1 soltero/n 2 casado/n  3 separado/n  4  otro");
```

```
estadocivil=captura.nextInt(); //se realiza la lectura del estado civil


while(estadocivil  <= N)

{

contador = contador +1;


switch(estadocivil) {

 case 1: System.out.println("\n estado civil soltero");contasolt = contasolt +1;

break;

 case 2: System.out.println("\n estado civil  casado/n");contacas = contacas +1;

break;

case 3: System.out.println("\n estado civil :  separado");contasepa = contasepa +1;

break;

 case 4: System.out.println("\n estado civil : otro");contaotro = contaotro +1;

 break;

default: System.out.println("\nSeleccione  solamente 1…4");       break;

 }// cierre del selector

System.out.println("\nSeleccione  un estado civil : 1 soltero/n 2 casado/n  3 separado/n  4  otro");

estadocivil=captura.nextInt(); //se realiza la lectura del estado civil

 }

mostrar("\nel total de soltero es /t"+contasolt+" el total de casado/t"+ contacas+" el total de separados es \t"+ contasepa+" el total de otros es\t"+ contaotro);

 System.exit(0);

    }

 public    static    void    mostrar    (    String    mensaje    )    {
System.out.println(""mensaje);     }

}
```

4.2.4.3 FORMA DE ACCESO A DATOS JOPOTIONPANEL

JOptionPane es una clase contenedora que nos permite mostrar un dialogo en aspecto gráfico con el que podremos interactuar para introducir o mostrar la información en tipo texto se encuentra en el paquete **javax.swing.**, no sobra recordar el uso de métodos convertidores por ser clase para almacenar la información en el tipo de dato nativo correspondiente.

Métodos de JOptionPane	Descripción	grafico
showInputDialog	**showInputDialog:leer** permite introducir información , este método devuelve un String con lo que hayamos escrito	
showInputDialog(Object message	**showInputDialog(Object message):**permite mostrar mensaje al dialogo, tipo cadena.	
showInputDialog(parentComponent, message, title, messageType):	**showInputDialog(parentComponent, message, title, messageType)**: permite personalizar el dialogo, con un título y un icono (error, información, advertencia, pregunta o plano). **(null, "Introduce un dato", "Titulo", JOptionPane.INFORMATION_MESSAGE);**	
(JOptionPane.ERROR_MESSAGE)	cambiar el icono a error	

(JOptionPane.WARNIG_MESSAGE), (**JOptionPane.QUESTION_MESSAGE**) (**JOptionPane.PLAIN_MESSAGE**).	cambia a advertencia pregunta plano	
showMessage:	permite **mostrar** información	
showMessage(parentComponent, message):	Imprime o mostrar un mensaje. ejemplo,**JOptionPane.showMessageDialog(null, "Muestra de informacion");**	
showMessageDialog(parentComponent, message, title, messageType	personaliza la información, con título e iconoasi:, **JOptionPane.showMessageDialog(null, "Error", "Error", JOptionPane.ERROR_MESSAGE);**	
showConfirmDialog:	Devuelve un valor numérico tipo constantes predefinido	
showConfirmDialog(parentComponent, message):	muestra un mensaje con las opciones **Si, No y Cancelar,** cada una de las opciones están definidas con una constantes (nos ayuda a indicar la opción sin saber que numero devuelve	**YES_OPTION:** devuelve si. **NO_OPTION**: devuelve no. **CANCEL_OPTION**: cancelar. **OK_OPTION**: aceptar.

showConfirmDialog(parentComponent, message, title, optionType	personalizar el dialogo, coloca un título y un tipo de opción constantes que muestra : **OK_CANCEL_OPTION**: aceptar y cancelar. **YES_NO_CANCEL_OPTION**: si, no y cancelar. **YES_NO_OPTION**: si y no.	int codigo=JOptionPane.showConfirmDialog(null, "¿Quieres un euro para una buena causa?", "Donacion", JOptionPane.YES_NO_OPTION, JOptionPane.PLAIN_MESSAGE);

○ **showConfirmDialog(parentComponent, message, title, optionType, messageType**	es igual que el anterior pero indicándole el tipo de mensaje (advertencia, error o información), se definen igual que en el primer método que hemos visto.	if (codigo==JOptionPane.YES_OPTION){
		
		System.out.println("Has pulsado en SI");
		
		}else if(codigo==JOptionPane.NO_OPTION){
		
		System.out.println("Has pulsado en NO");
		
		}

Es importante nombrar algunos de los métodos convertidores de la clase cadena (String) a tipo de dato nativo ligados a sus respetivas clases como Integer.parseInt(cadena); Double.parseDouble(cadena) ; Float.parseFloat(cadena).

Tomado de: http://www.discoduroderoer.es/metodos-joptionpane-de-java/.>

4.3 APLICACIÓN DE LA ORIENTACIÓN A OBJETOS EN EL LENGUAJE JAVA Y EL MODELADO EN DIAGRAMA DE CLASE Y CASO DE USO.

Explorar herramientas que permiten una interpretación, diseño y modelado para la elaboración de aplicaciones en pro de una metodología de orientación a Objeto por medio de proyectos, aplicándolos adecuadamente el desarrollar software.

Elaboración de aplicaciones preliminares empleando la metodología orientación a objetos desde una plataforma como es la de java inicialmente incluyendo los diagramas de clase y casos de uso.

4.3.1 ORIENTACIÓN A OBJETOS

El paradigma Orientado a Objeto, es posterior al paradigma estructurado y es pre al paradigma orientado a aspectos. Siendo una metodología combinada por algoritmos y programación computacional o no que de manera detallada permiten la solución de un problema simple o complejo. Iniciado por la década de los años 60 -70 con lenguajes de programación como el Simula y Smalltalk, su mayor auge se logra con el lenguaje java que proviene de otro llamado OaK que es un proyecto de la SunMicroSystem originalmente llamado Green project y el otro lenguaje C# perteneciente a Microsoft ambos teniendo como base a C++ desde la década de los años 80. Para la década de los 90s Grady Booch (Booch, 1995)establece unos elementos referentes como fundamentales y secundarios para dicha metodología.

La orientación a objetos es una técnica de modelado de sistemas, que pueden ser o no computacionales. Mediante la orientación a objetos se obtiene una representación del problema en cuestión, representación cercana a como ocurre en el mundo real. Es decir, estamos rodeados de objetos, alumno, profesor, escuela, estos objetos a su vez interactúan entre ellos para obtener servicios unos de otros. En la orientación a objetos se tienen también objetos similares a los de la realidad que también reciben y solicitan servicios unos de otros.

En teoría, las principales ventajas de los modelos orientados a objetos son:

> "**El entendimiento del sistema es más fácil** dado que la diferencia semántica entre el sistema y la realidad es reducida."
>
> "**Las modificaciones al modelo tienden a ser locales** ya que frecuentemente afectan a una sola entidad, que está representada por un objeto."

public Estudiante() //Constructor de la clase Estudiante

CLASIFICACIÓN: es el ordenamiento categórico de los miembros dato o procedimiento de un determinado problema deben ser agrupados donde corresponda por el desarrollador para establecer la llamada clase, es tal la seriedad hasta establecer en lo posible jerarquías de orden superior como las superclase o inferior sub clase (especializaciones) como la herencia, proceso por medio del cual se le

transfieren los atributos y los métodos (variables o características en algoritmos y los subprogramas) propia para compartir y extender, se considera esta clase como un molde de la cual se definen uno o más copias (duplicados) que son en la aplicación los Objetos quienes se caracterizan por permitir el acceso a las propiedades (atributos) y al comportamiento(métodos).

Es notable la existencia de la herencia múltiple pero no aplica en java. Un ejemplo continuando con el automóvil descripto en la abstracción, sus componentes agrupados a nivel superior seria vehículo, la clase automóvil que heredaría sus características y procedimientos. Teniendo más elementos en la abstracción se toman los vehículos terrestre, los aéreos y los acuáticos agrupados pertenecen a la superclase vehículo, por tanto el automóvil sería una subclase herencia simple de los vehículos terrestres, luego la clase que agrupa los vehículos anfibios heredaría de dos clases la terrestre y la acuática en nuestra descripción.

4.3.1.1 ELEMENTOS FUNDAMENTALES DE LA ORIENTACIÓN A OBJETOS

Apoyados en (Booch, 1995) "una clase es un conjunto de objetos que comparten una estructura y un comportamiento común. Un objeto es una instancia de una clase". Jacobson describe a una clase como "una definición, una plantilla o molde para habilitar la creación de nuevos objetos y"…" describe la estructura interna de estos objetos.

La clase es un ente real o abstracto posible de ser programado.

Entonces, una clase es una descripción de un grupo de objetos que comparten propiedades comunes (atributos), comportamiento común (operaciones) y asociaciones con otros objeto tomemos como ejemplo, la computadora en la cual esté usted trabajando es una instancia de la clase que describe a todas las computadoras personales, que podría llamarse "Computadora Personal", tiene una estructura de información que incluye su marca, velocidad, espacio en disco, memoria, entre otros Tiene comportamientos como encenderse, permanecer en espera, apagarse, bloquearse, aumentar la memoria, segmentar el disco duro, etc. Tiene asociaciones de agregación con otros objetos como el ratón o la impresora.

Un objeto es una entidad lógica que contiene datos (características) y un código especial (métodos) que indica como manipular los datos.

Objetos son entidades o duplicados de las clases que combinan estado (atributo), comportamiento (método) e identidad bien estructurada, se conforman con el fin de solicitar o enviar mensajes. Se conforman con un identificador de clase (tipo de clase nombre de objeto tipo clase) luego se instancia.

"un objeto tiene límites definidos tales como estado, comportamiento e identidad"

El estado de un objeto es una de las posibles condiciones en las que un objeto puede existir, normalmente cambia en el tiempo, es implementado por un conjunto de propiedades (atributos), los valores de éstos, y las relaciones que el objeto puede tener con otros objetos.

Cada objeto tiene una identidad única, incluso cuando se encuentra en el mismo estado que otro objeto.

Una clase es una definición abstracta de un objeto. La clase define la estructura y el comportamiento de cada objeto de la clase. Sirve como una plantilla para crear objetos. Los objetos pueden agruparse en clases.

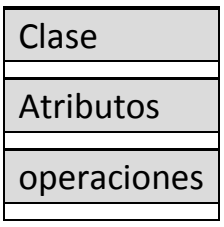
Por ejemplo, en la siguiente imagen tenemos 4 objetos. ¿Cuántas clases puede distinguir?

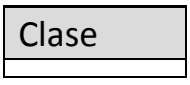
Lineamientos para encontrar clases

Identificar adecuadamente las clases de un sistema y asignarles las responsabilidades de forma correcta es lo más importante de un proyecto de software y de él dependerá su éxito, su reusabilidad, claridad y mantenibilidad.

Las clases se encuentran en los documentos del proyecto, requerimientos, casos de uso, entrevistas con el cliente, experiencia en proyectos similares, etc.

En UML las clases se representan en alguna de las dos siguientes notaciones:





LA ABSTRACCIÓN permite extraer (comprender) del problema o del mundo real las variables o características que solo son útiles al momento de solucionarlo, determinando inclusive su tipología, su delimitación, su estado y comportamiento sin especificar como se ha de realizar la implementación en relación al algoritmo.

> Proceso mental que permite al individuo comprender un concepto de un objeto, sin tener al objeto de forma tangible como puede ser Estudiante, inventario, nómina.
> Abstracción se puede definir como la capacidad de examinar algo sin preocuparse de los detalles internos.

Un ejemplo para comprender la abstracción de un Automóvil seria por ejemplo algunas de sus características: marca, placa, color, fecha de fabricación, modelo igual obtenemos sus métodos o funciones típicas de esta entidad como por ejemplo: frenar, encender, acelerar, parar; de esta forma se obtendría la definición de variables para la algoritmia luego del análisis.

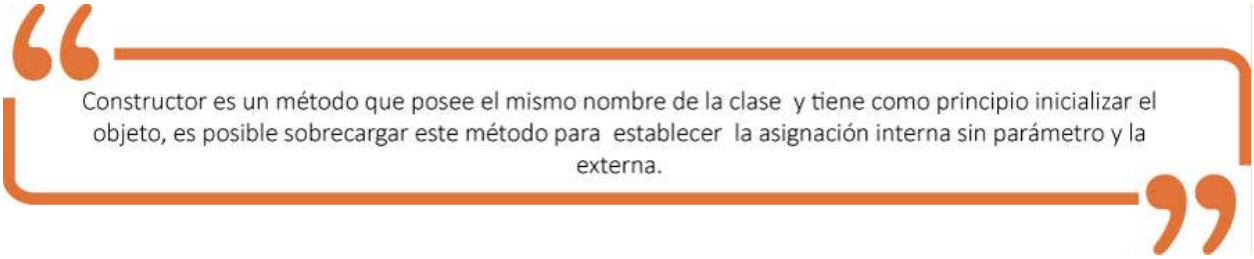
ENCAPSULAMIENTO: posibilidad de ocultar del exterior mínimamente para no hacer fácil su acceso los miembros (dato y/o procedimiento) clasificados (agrupados en la clase o en el proyecto), permitiendo una alta cohesión. Con ello solo se permite a los privilegiados quienes están autorizados a tener entrada a las componentes ocultas y realizar actualizaciones según corresponda; en relación al algoritmo se protege un método siendo independiente sin saber que contiene el uno o el otro, según el alcance local o global o encarpetando el archivo en directorios.

> "Toda la información de un objeto está almacenada dentro del mismo objeto y sólo puede ser manipulada cuando al objeto se le ordena que lleve a cabo alguna operación. El comportamiento y la información están encapsulados en el objeto.
> La única forma de realizar operaciones en el objeto es realizar operaciones en él.
> Los objetos, entonces soportan el concepto de ocultamiento de la información, esto es, ocultan su estructura interna de su alrededor. Cada una de las operaciones del objeto tiene como propósito algún comportamiento del mismo. No se necesita saber cómo está implementada alguna operación o cómo se representa la información, sólo necesitamos conocer las operaciones que tiene como interfaz para comunicarnos con él."
> [Jacobson, 1992]

MODULARIDAD es la subdivisión de los procesos en actividades específicas e independientes estas se encargan de cambiar el estado de una o más componentes al ser requerido por un mensaje, el acoplamiento del método es bajo pero con una **alta relación de sus componentes** por que los cambios ocurridos en unos métodos afectan a otros. Desde la algoritmia se referencia a los subprogramas, como programas específicos pequeños o no, tipo función o procedimiento que cumplen su tarea y retorna a donde **se les hace el llamado al terminar.**

Es la propiedad que permite subdividir una aplicación en partes más pequeñas (llamadas *módulos*), cada una de las cuales debe ser tan independiente como sea posible de la aplicación en sí y de las restantes partes (bajo acoplamiento).

El Módulo A *depende* del Módulo B si cualquier cambio en el Módulo B implica que el Módulo A también tenga que ser modificado. A veces se dice que el Módulo A es *un cliente* del Módulo B, o que el Módulo B actúa como *servidor* del Módulo A.



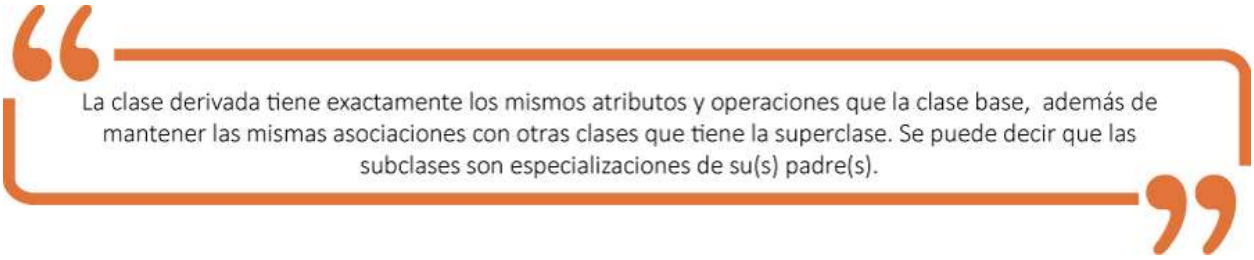
Destructor método iniciado cuando se elimina un objeto de la memoria.

Automovil objetoauto = new Automovil ();

Automovil objetoauto1 = new Automovil ("placa1", "azul");

Automovil objetoauto2 = new Automovil ("placa2", "rojo");

Generalización y Herencia



Herencia es la propiedad que tienen las agrupaciones de proporcionar sus componentes propiedades o características y sus procedimientos o métodos de manera jerárquica. Para especificar la herencia se emplea extends y el nombre de la clase padre ejemplo:

public class Autoterrestre extends Vehículo{ }

La clase Autoterrestre heredara los componentes de la clase Vehículo en el acto posibilitando definir los miembros propios que lo harán diferente de otros.

Instancia es la asignación de espacio en memoria por el operador new y el método constructor para que un objeto pueda ser utilizado.

Es el proceso de crear objetos pertenecientes a una clase. El objeto es la instancia de la clase a la que pertenece. La clase se debe instanciar ya que cada vez que se ejecute el programa siempre se trabajará con respecto a la instancia y no a clase original. Requiere del operador new y el método constructor.

***Estado* está compuesto de datos, atributos a los que se habrán asignado unos valores concretos.**

Para el nombre de atributo placa se le asigna un valor "placa1" tipo cadena por medio del objeto entidad objetoauto1 que le envió un mensaje a través del método constructor de la clase Automóvil.

Comportamiento definido por procedimientos o métodos con que puede operar dicho objeto, es decir, qué operaciones se pueden realizar con él.

Identidad propiedad del objeto o duplicado que lo diferencia de los otros.

4.3.1.1 ELEMENTOS SECUNDARIOS DE LA ORIENTACIÓN A OBJETOS

Estas características en la metodología orientada a objetos se establecen como fundamentales es necesaria su presencia en una aplicación de este tipo, **las siguientes se consideran secundarias y pueden o no estar en nuestra comprensión**:

P**OLIMORFISMO:** cuando dos métodos diferentes tienen igual el nombre en distintos objetos, esto es posible, pero solo se activara según el accionar o el llamado del mensaje o sea depende del momento de ejecución que es cuando se le asigna al método correspondiente.

Otro:

> El Polimorfismo es la respuesta distinta frente a una llamada a un método dependiendo de la naturaleza del objeto.
> Consiste en definir métodos distintos, que comparten el mismo nombre, pero que se aplican a clases diferentes.
> Por ejemplo, un método llamado breathe puede responder de manera distinta dependiendo de quién lo invoque:

Por ejemplo, un método llamado mostrar () puede responder de manera distinta dependiendo de quién lo invoque.

SOBRECARGA cuando **los métodos poseen la misma identificación pero sus parámetros varían en sus tipos respectivos**, su ubicación y en número.

Es el uso de dos o más métodos dentro de una misma clase para diferentes objetivos, estos pueden llevar los mismos parámetros o diferentes

REUSABILIDAD: hace que un proceso o procesos de un programas de un determinado programador se pueda aplicar cuantas veces sea requerido por el o por otro dado el caso.

PERSISTENCIA: cuando se define la duración del almacenamiento en memoria de los diferentes identificadores como temporal (solo mientras esta la aplicación), dinámica (una duración moderada por asignación interna) o permanente (dura por fuera de la aplicación como en archivos o en base de datos).

RECOLECCION DE BASURA: consiste en recuperar los espacios de memoria que han sido liberados o no.

4.3.2 INTRODUCCIÓN AL UML DIAGRAMA DE CLASE Y CASO DE USO.

Se traduce Lenguaje Unificado de Modelo , se describe como una técnica todavía no metodología encargada de agrupar notaciones y diagramas estandarizados para facilitar el diseño y modelado de estructuras relacionadas con los sistemas de información y utilizados en el desarrollo del mismo de gran apoyo para la metodología orientada a objetos se clasifica en estructural donde se agrupa a clases, objetos, componentes e instalaciones y los comportamiento como los casos de uso, secuencia, colaboración estado y actividades, para nuestro caso se tomara el diagramas de clases y el diagrama de caso de uso este describe al actor su operación con el sistema la tipificación y la jerarquía de los datos está conformado por el actor, el caso , la relación y su comunicación

> **UML, es un lenguaje de Modelos Unificados por sus siglas en inglés,** es el lenguaje de modelos de sistemas de software más popular en la actualidad, **es un lenguaje para construir, especificar, visualizar y documentar sistemas de aplicativos.**

4.3.2.1 COMPRENSIÓN DE LOS MODELOS

> **Un modelo es una colección de imágenes y texto que representa algo, para nuestro software.**

Los modelos son valiosos por muchas razones específicas, en gran parte, constan de imágenes e incluso, las imágenes simples pueden transmitir más información que una gran cantidad de texto. Los modelos son valiosos porque es más fácil dibujar algunas imágenes sencillas que escribir código o incluso texto que describa lo mismo, además **es más económico, rápido y fácil de cambiar modelos que cambiar código o texto.**

El UML **es una definición de un lenguaje de símbolos y relaciones comunes** que tiene un **significado común**, si todos los participantes hablan UML, entonces las imágenes tienen el mismo significado para todo aquello que las observe, por lo tanto, **aprender UML es esencial para ser capaz de usar imágenes para experimentar económico, flexible y dar rápidas soluciones.**

4.3.2.2 USO DE LOS MODELOS

> **Los modelos consisten en diagramas o imágenes, lo que intenta con los modelos es que sea más fácil de producir y experimentar que con solo código.**

4.3.2.2.1 Creación de diagramas

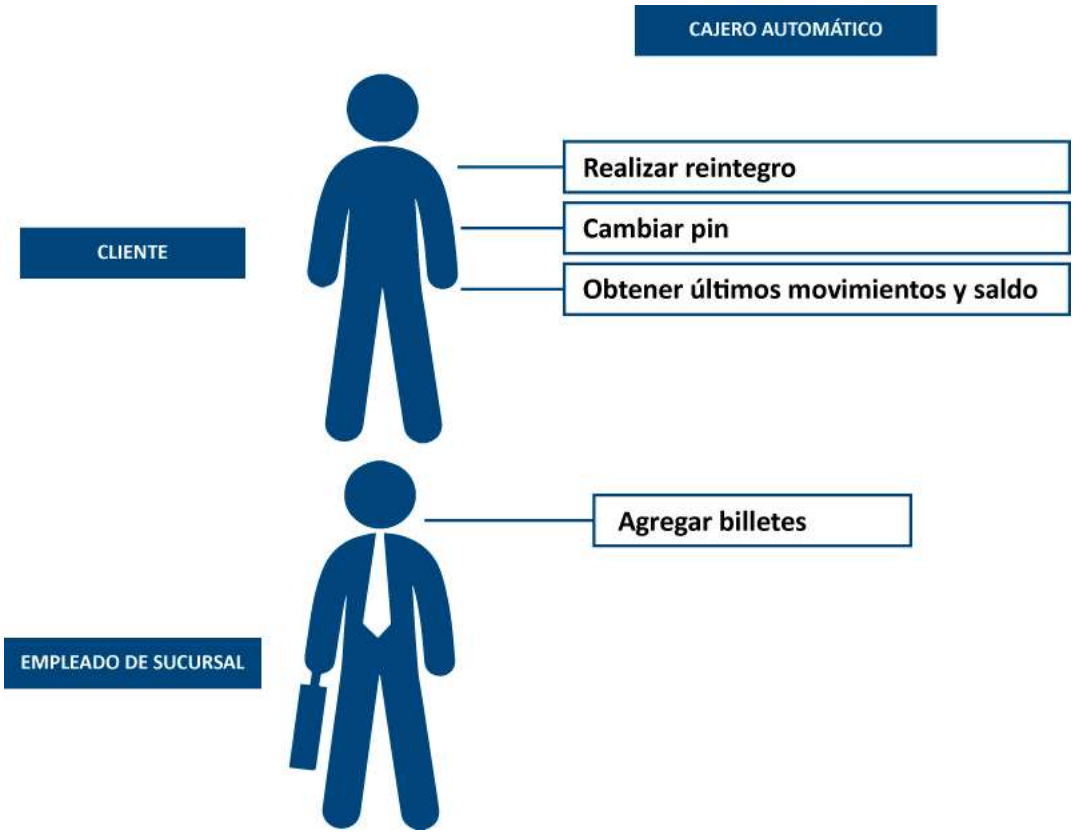
La primera regla de la creación de diagramas de modelos es que el código y el texto consumen tiempo, y no queremos pasar una gran cantidad de tiempo creando documentos de texto que nadie leerá. Lo que si queremos hacer es captar con exactitud las partes importantes del problema y una solución. Lamentablemente, esta no es una prescripción para el número o la diversidad de diagramas que necesitamos crear y no indica cuanto detalle necesitamos agregar a esos diagramas.

4.3.2.2.2 Tipos de Diagramas

> **Existen varios tipos de diagramas que el individuo puede crear, miremos los tipos que se pueden usar y los tipos de información que se pretende transmitir con cada uno de los diagramas.**

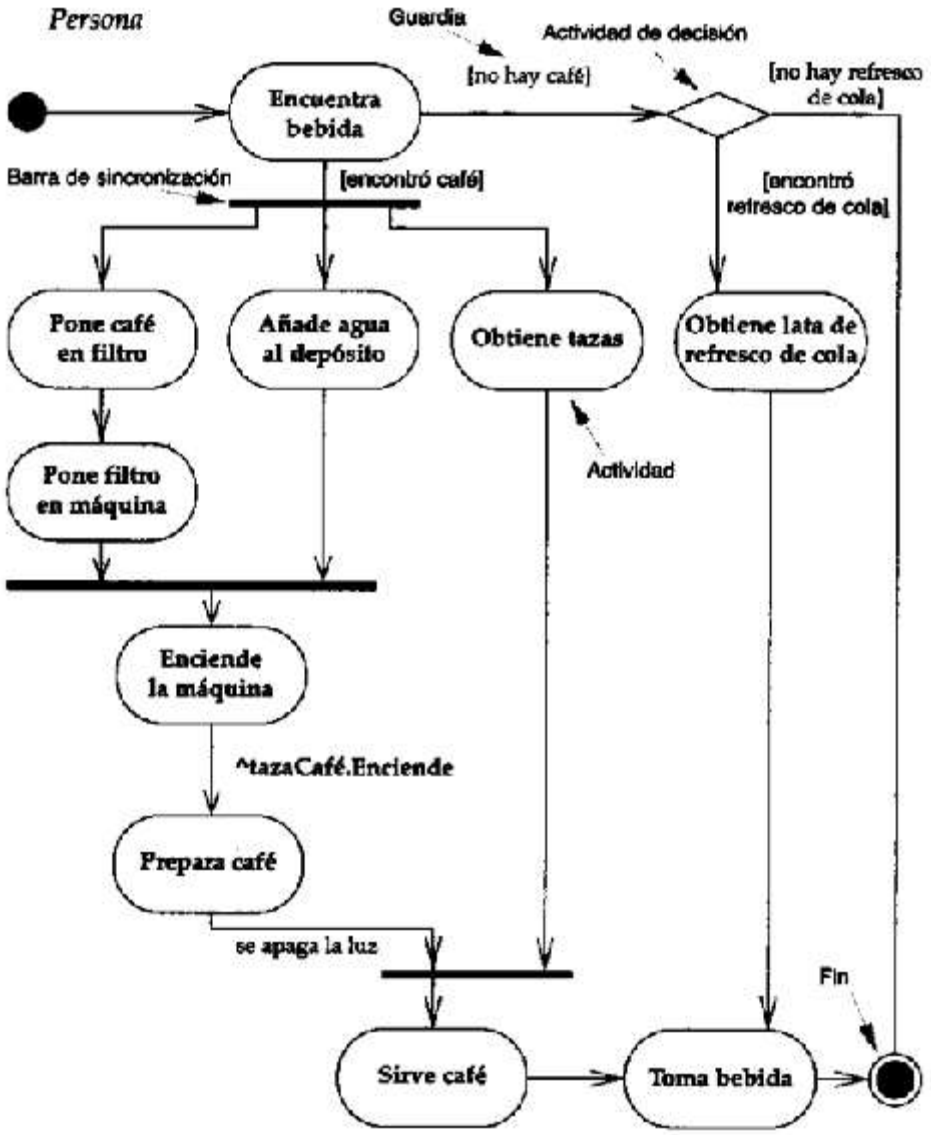
4.3.2.2.2.1 DIAGRAMA DE CAJAS DE USO (CASOS DE USO)

Los **símbolos principales** de las cajas de uso son **el actor**, y **el ovalo de la caja de uso**, los diagramas son responsables principalmente de **documentar los macro requisitos** del sistema, pensando en la lista de capacidades que debe tener o proporcionar el sistema.



4.3.2.2.2.2 DIAGRAMA DE ACTIVIDADES

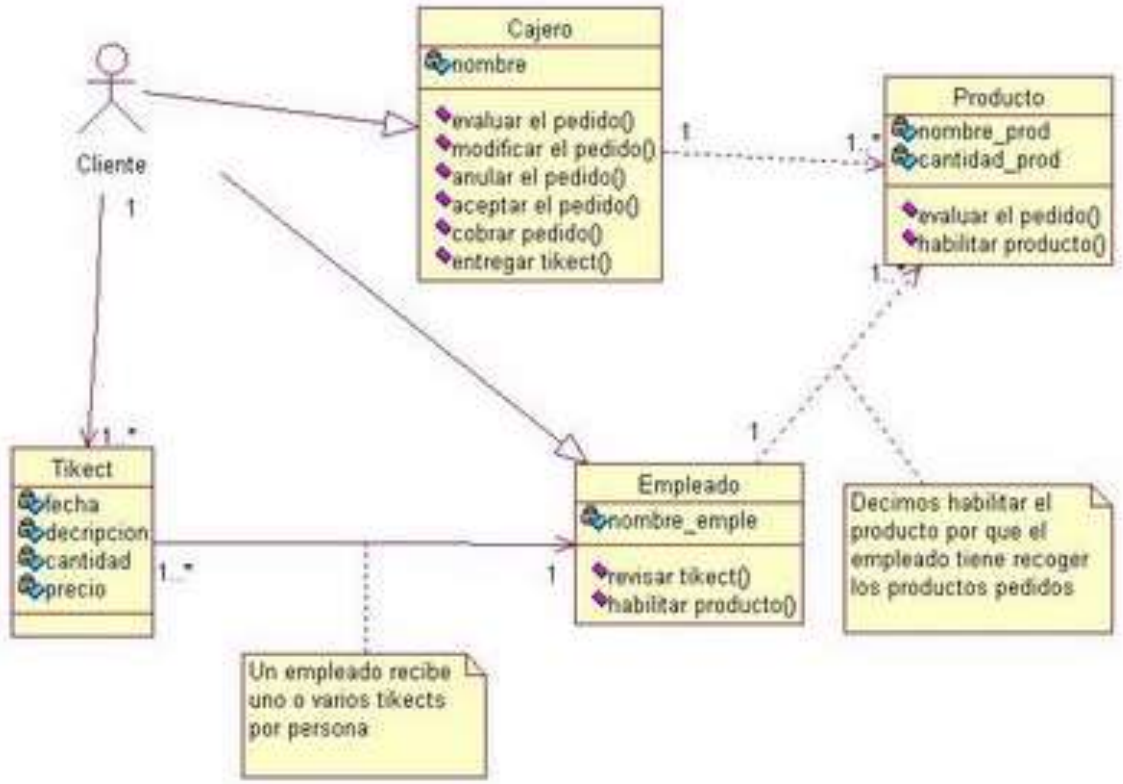
> **Un diagrama de actividades es la versión UML de un diagrama de flujo. Los diagramas de actividades se usan para analizar los proyectos y, si es necesario, volver a realizar la ingeniería de los procesos.**



Un diagrama de actividades es una herramienta excelente para analizar problemas que al final, el sistema debe resolver. Como una herramienta de **análisis**, no queremos empezar resolviendo el problema a un nivel técnico mediante la asignación de clases, pero **podemos usar diagramas de actividades para entender el problema e incluso refinar los procesos que comprenden el problema**.

4.3.2.2.2.3 DIAGRAMA DE CLASES

> **Los diagramas de clases se usan para mostrar las clases de un sistema y la relación entre ellas. Una sola clase puede mostrarse en más de un diagrama de clases y no es necesario mostrar todas las clases en un solo diagrama monolítico de clases. El mayor valor de mostrar las clases y sus relaciones desde varias perspectivas, de una manera que ayudara a transmitir la comprensión más útil.**



Los **diagramas de clases muestran una vista estática del sistema**, no describe los comportamientos o cómo interactúan los ejemplos de la clase. Para describir los comportamientos y las interacciones entre los objetos de un sistema, podemos revisar los diagramas de interacción.

4.3.2.2.2.4 DIAGRAMAS DE INTERACCIÓN

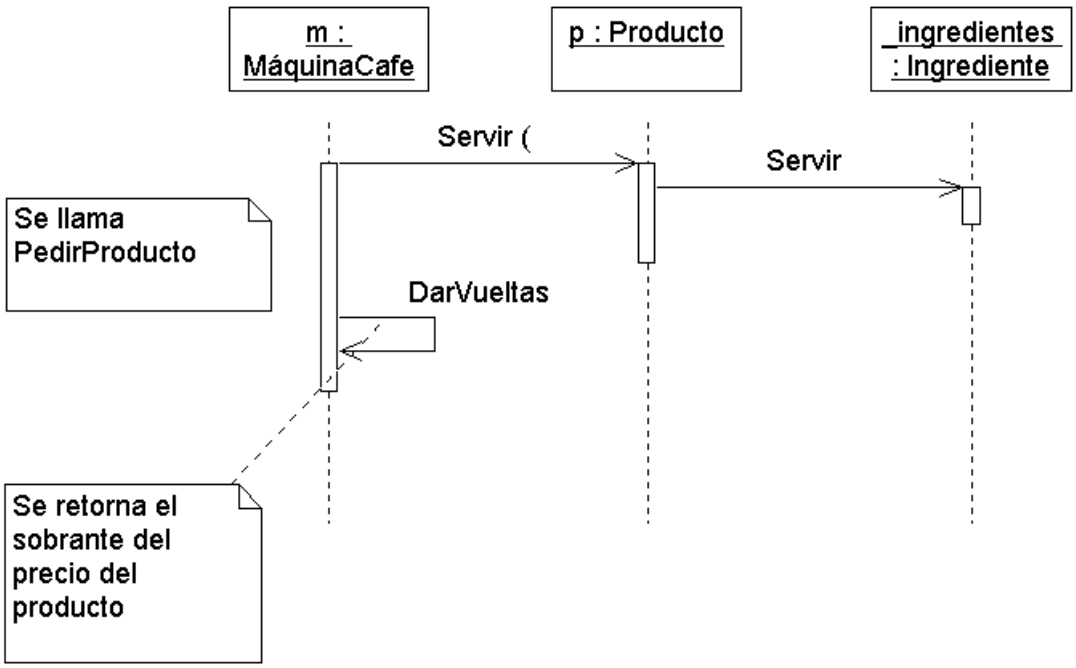
Existen 2 tipos de diagramas de interacción, la secuencia y la colaboración. Ambos **transmiten la misma información, empleando una perspectiva un poco diferente**.

Los diagramas de secuencia muestran las clases a lo largo de la parte superior y los mensajes enviados entre esas clases, modelando un solo flujo a través de los objetos del sistema. Los diagramas de colaboración usan las mismas clases y mensajes, pero organizados en una disposición espacial.

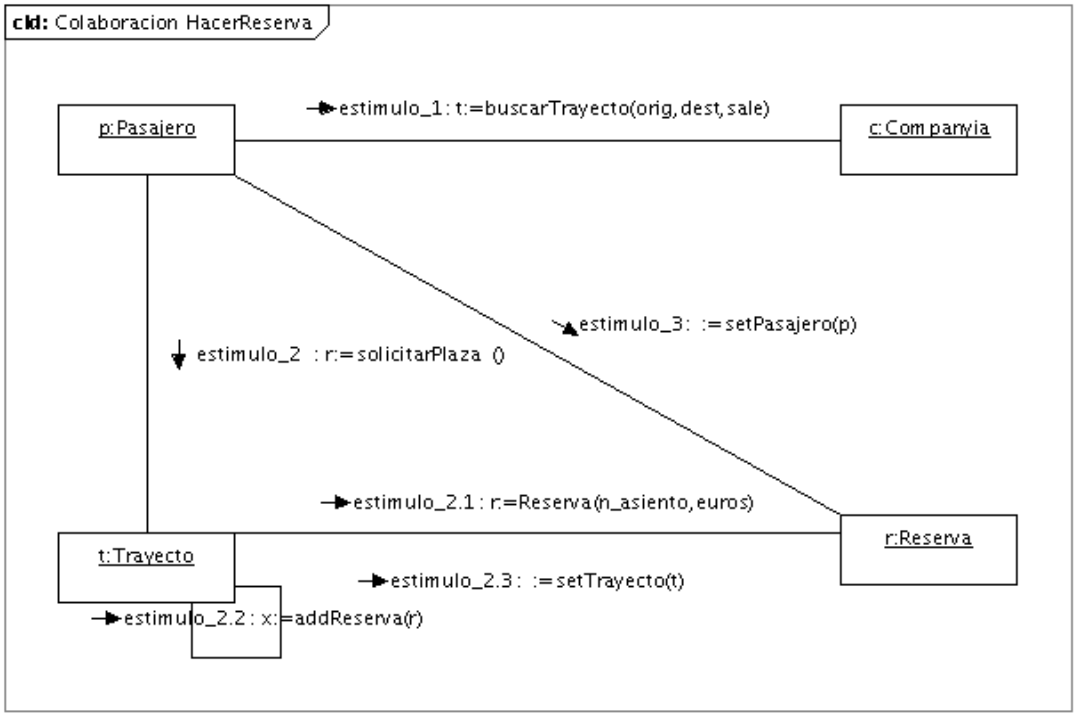
Un diagrama de secuencia implica un ordenamiento en el tiempo al seguir la secuencia de mensajes desde arriba a la izquierda hasta abajo a la derecha. Debido a que en el diagrama colaborativo no se indica en forma visual un ordenamiento en el tiempo, enumeramos los mensajes para indicar el orden en el cual se presenta.

> **Algunas herramientas convertirán de manera automática los diagramas de interacción entre secuencia y colaboración.** Pero no es necesario crear los dos tipos de diagrama. En general, se percibe que un diagrama de secuencia es más fácil y más común.

Ejemplo (Diagrama de secuencia)

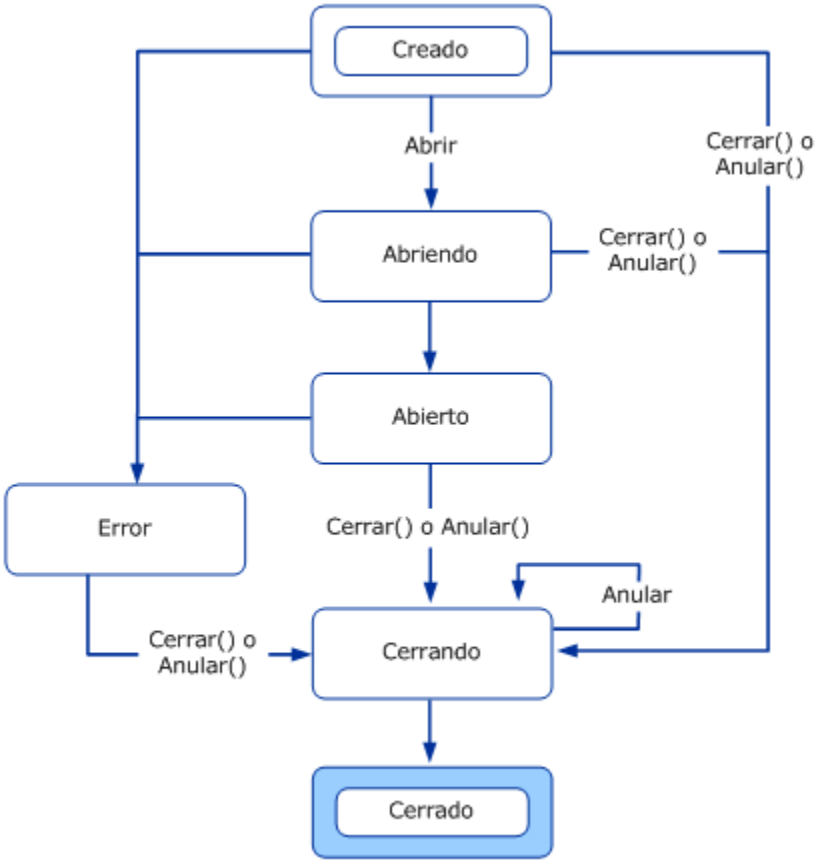


Ejemplo (Diagrama de colaboración)



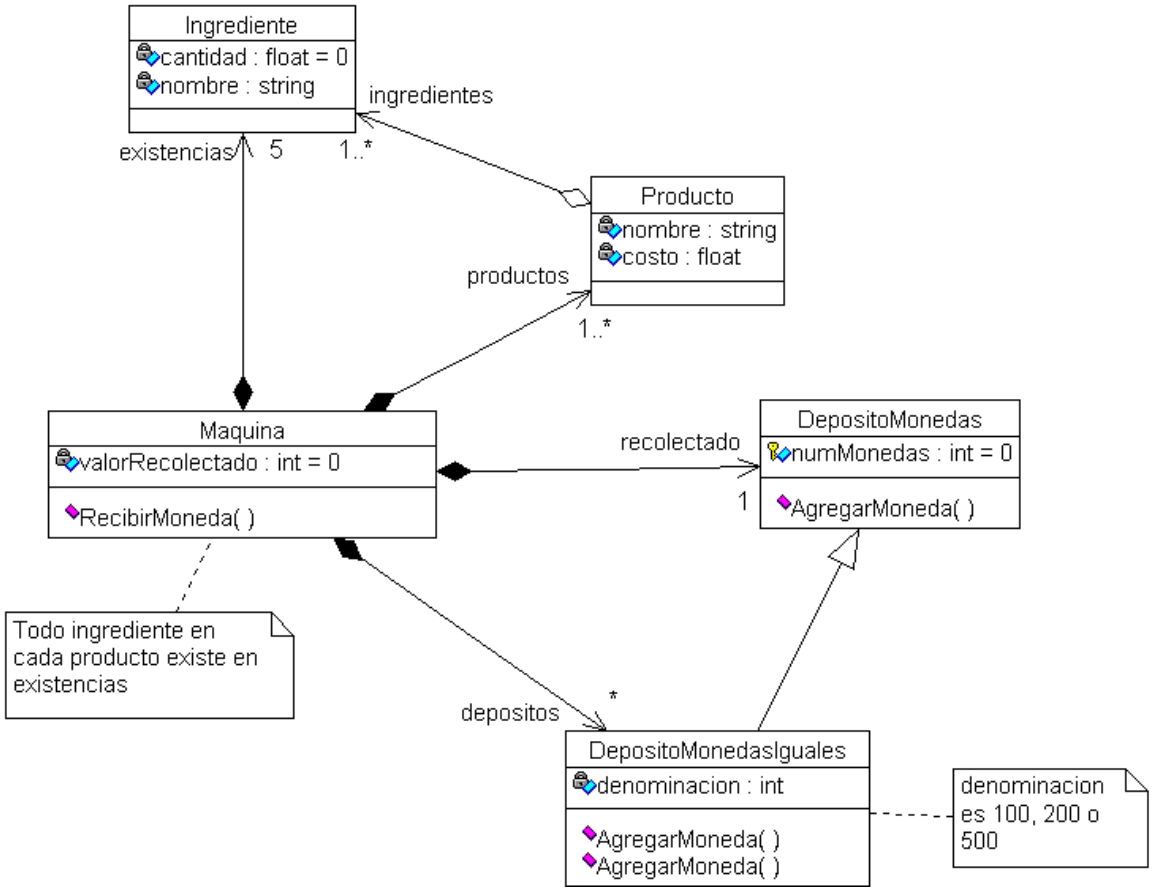
4.3.2.2.2.5 DIAGRAMA DE ESTADO

> **Mientras que los diagramas de interacción muestran los objetos y los mensajes que se pasan entre ellos, un diagrama de estado muestra el estado cambiante de un solo objeto, conforme este pasa por un sistema.**



4.3.2.2.2.6 DIAGRAMAS DE COMPONENTES

El UML define varios tipos de modelos, incluyendo modelos de análisis, para el diseño y para implementación, sin embargo, nada hay que le fuerce a crear o mantener tres modelos para una aplicación. Un ejemplo de un diagrama que podría encontrar en un modelo de implementación es de componentes. En un diagrama de componentes, estos se muestran en el producto final.

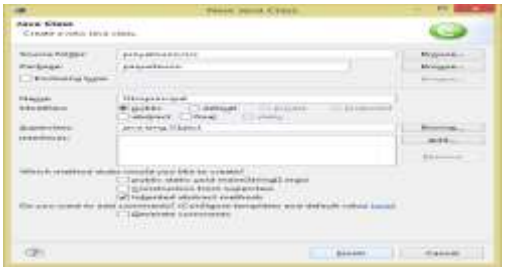
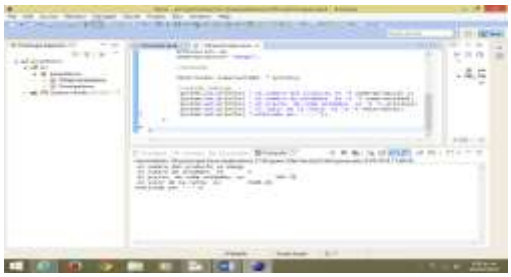


4.3.2.2.2.7 OTROS DIAGRAMAS

Existen **otros tipos o variaciones de diagramas que podemos crear**. Por ejemplo, un **diagrama de topología del despliegue** le mostrara como se verá desplegado su sistema. Lo común es que un diagrama de este tipo contenga símbolos que representan cosas, como servidores web, servidores de bases de datos y varios dispositivos diversos, así como software que construye la solución que usted requiere. **Este tipo de diagrama es más común cuando usted está estructurando sistemas distribuidos en n hileras**.

4.3.3 APLICACIÓN DE LA ORIENTACIÓN A OBJETOS EN EL LENGUAJE JAVA

La ventana de código presenta varias estructuras las cuales pueden se asignada por el editor en otro caso si es requerido se procede a codificarlo. Como lo muestra la siguiente tabla

<table>
<tr>
<td> 
Nombre de clase sin opción de método</td>
<td> 
Sin método main por lo tanto debe digitarlo</td>
</tr>
<tr>
<td> 
Se declara, inicializa , calcula y muestra</td>
<td> 
No se captura (lee) el resultado</td>
</tr>
<tr>
<td>

```
package paqueteuno;


public class Otroprincipal {


    public static void
main(String[] args) {


    // definir  variable


    int  numerounidad;

    double preciou;

            String
nombreproducto;

    double Valorventa;
```

</td>
<td></td>
</tr>
</table>

```
                //inicializar las
      variables asignacion interna


                  numerounidad= 7;

                   preciou=345.78;

                   nombreproducto=
                          "mango";



                        //proceso


                       Valorventa=
            numerounidad  * preciou;



                    //salida informe

                 System.out.println(
     " el nombre del producto es "+
                   nombreproducto );

                 System.out.println(
  " el numero de unidades  es \t "+
                       numerounidad);

                 System.out.println(
  " el precio  de cada unidades  es
                      \t "+ preciou);

                 System.out.println(
   " el valor de la venta  es \t "+
                        Valorventa);

                 System.out.println(
            "realizado por -----");

                                }



                                }
```

4.3.3.1 APLICACIÓN DE LA ORIENTACIÓN A OBJETOS EN EL LENGUAJE JAVA

Una nueva aplicación tipo procedimental (sin clase) pero realizando uno de los tantos formatos con captura la clase Scanner en una nueva aplicación clase llamada Principal capturando esto requiere de una carpeta adicional que contiene las clases para la captura llamada útil, se requiere importar desde java ubicándolo luego del paquete pero antes de la clase respectiva como lo muestra la siguiente tabla.

 Crear una nueva clase	 Seleccionar la métodos scanner
 La declaración y captura de la información	 La puesta en ejecución Ojo al entrar los decimales emplee la , según el teclado .

La clase scanner presenta unos métodos internos que se pueden referenciar o aplicar en un caso específico por ejemplo una lectura convirtiendo la captura a numérico doublé como el método next.Double() obtenido del objeto scanner .

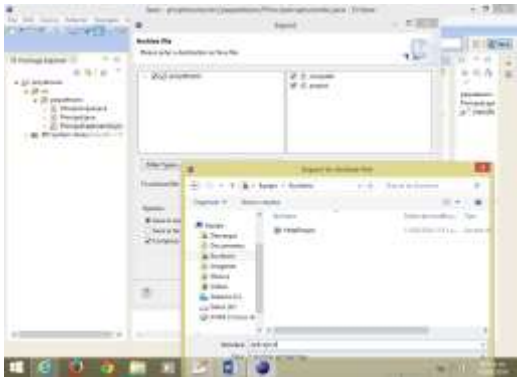
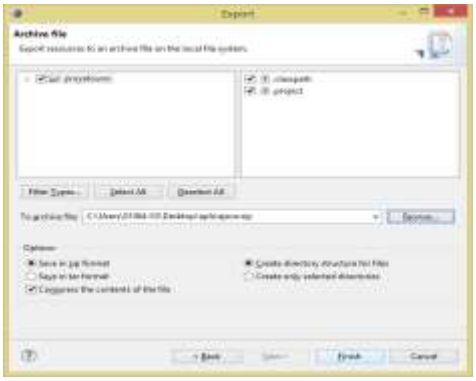
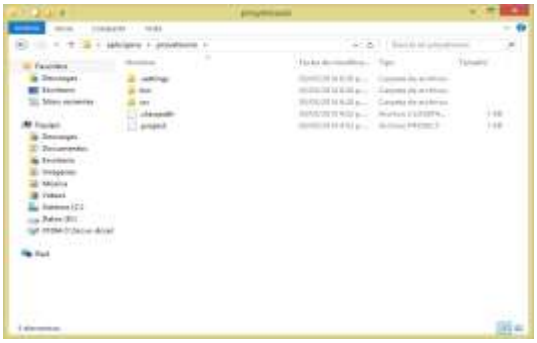
Cuadro de ejemplo de una aplicación

```
                                                    package paqueteuno;


        // se debe importar el paquete  util para usar  la clase Scanner
```

```
import java.util.*;


public class Principalcapturando {


public static void main(String[] args) {

// definir  variable


int  numerounidad;

double preciou;

String nombreproducto;

double Valorventa;


// asignacion Externa

//requiere un paquete llamado util , contiene
la clase Scanner

Scanner captura= new Scanner (System.in);

// para la lectura  se acompaña con un
mensaje  previamente


System.out.println( " Digite el nombre del
producto " );


//nombreproducto= "mango";

nombreproducto= captura.next();//tipo cadena


System.out.println( " Digite el numero de
unidades  " );

numerounidad= captura.nextInt(); //metodo
next tipo entero
```

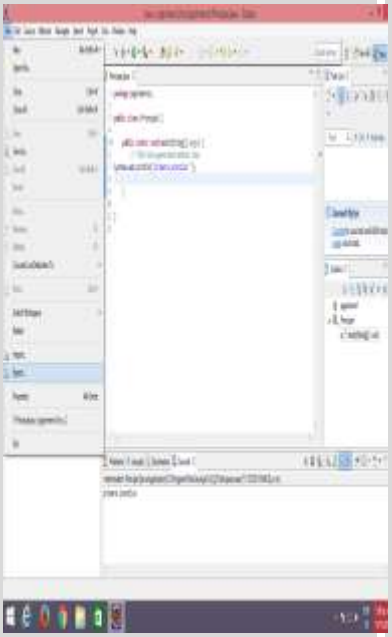
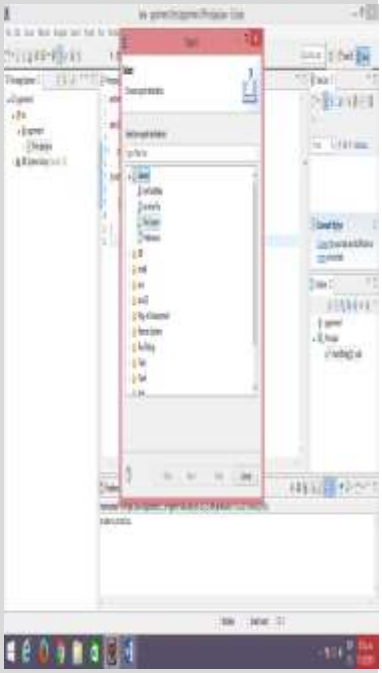
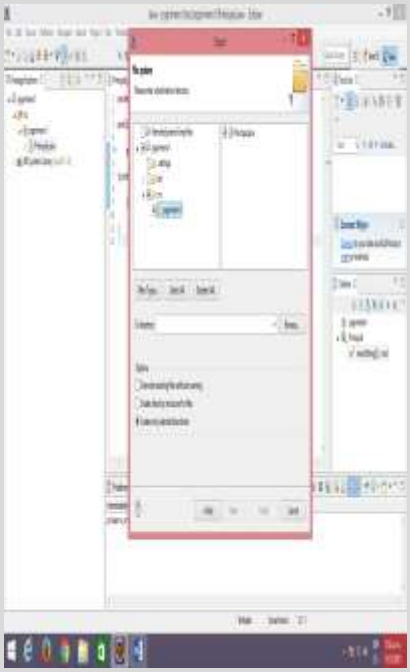
```
                    System.out.println( " Digite el precio
                                        unitario  " );


                    preciou=captura.nextDouble();


                                        //proceso


                    Valorventa= numerounidad  * preciou;


                                        //salida informe

                System.out.println( " el nombre del producto
                                    es "+ nombreproducto );

                 System.out.println( " el numero de unidades
                                    es \t "+ numerounidad);

                    System.out.println( " el precio  de cada
                                unidades  es \t "+ preciou);

                 System.out.println( " el valor de la venta
                                    es \t "+ Valorventa);

                System.out.println( "realizado por -----");


                                                        }


                                                        }
```

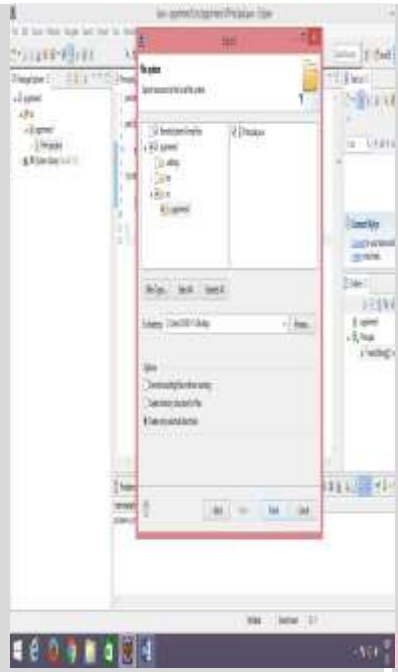
Para guardar se realizan los siguientes pasos que se presentan en el siguiente cuadro

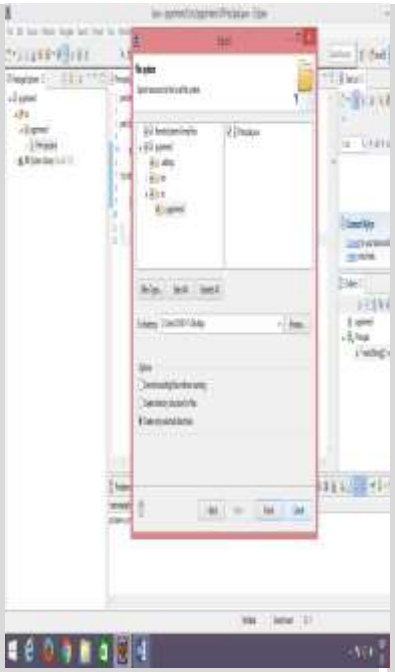
	
Procedemos desde la barra de menú a seleccionar el menú archivo(flie) desde allí se toma el comando exportar	luego desde la ventana selecciono general y archivos de sistemas Se especifica en cual unidad se realizara el almacenamiento generalmente escritorio
	

4.3.3.2 GUARDAR EXPORTAR E IMPORTAR EN EL LENGUAJE JAVA

Para almacenar una aplicación desde su proyecto se realiza el proceso de exportar para almacenar la información desde el editor o importar traer el proyecto codificado y realizar actualizaciones como se muestra en el cuadro siguiente para realizar la exportación por pasos

Iniciar la exportación del proyecto 	General Archivos de sistema 	Selecciona todos los archivos Y ubicar el directorio 
Selccionar	finalizar	Comprobar si esta almacenado en el escritorio





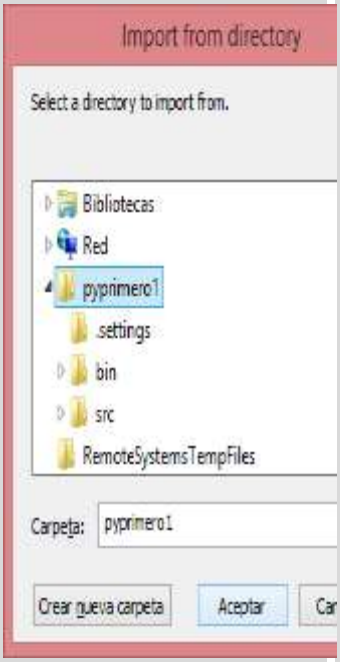
Se comprime el archivo en caso de enviarlo por correo

Queda

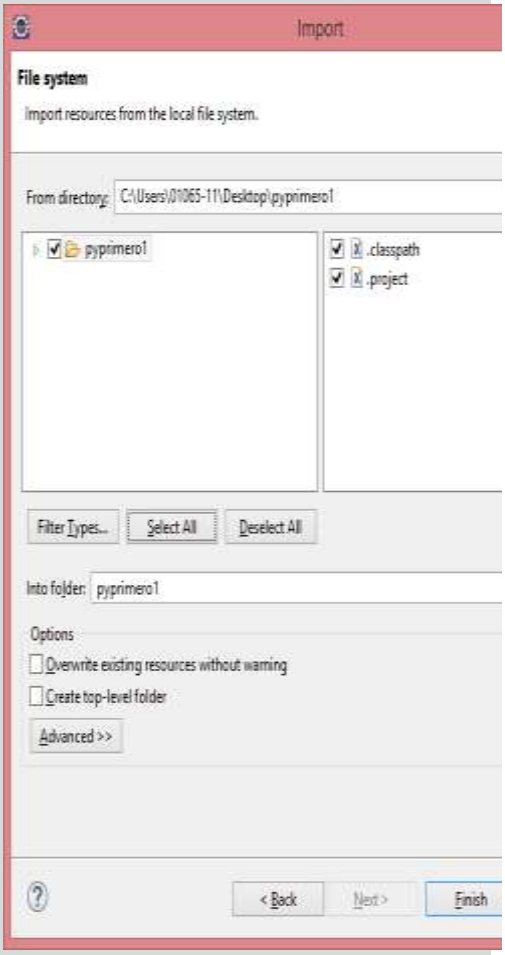
Para importar

Para traear el proyecto almacenado al editor se importa

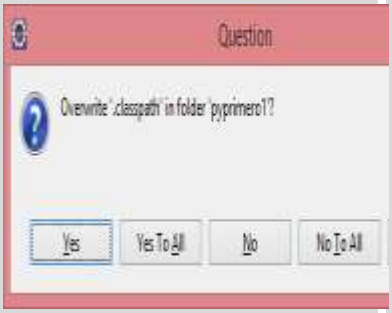
Selccionar la carpeta



Con lotodos los archivos



Se comprueba y se sobre escribe si es nescsario



Se visualiza el proyecto en la ventana del explorador del editor listo para editar

4.3.3.3 CON (ESTILO, JAR)

Los jar Es crear un archivo (ejecutable) con extensión Jar se construye a partir de la línea de comandos, sin depender de los IDEs (entorno Integrado de desarrollo) más conocidos como Netbeans o Eclipse. Es de aclarar que algunos editores tienen su generador de jar.

Primero se debe crear un Archivo de Manifiesto en el cual se deberá poner el nombre de la clase que contenga el método main, este archivo puede ser creado a partir del Bloc de Notas, entonces se abre el Notepad (Bloc de Notas) y escribimos lo siguiente:

Main-Class: HolaMundo

Sealed: true

***Nótese que no se agrega el .class y que la palabra MAIN-CLASS: es obligatoria.**

Donde HolaMundo es el Nombre de la Clase que tiene mi método main, una vez escrito eso lo guardamos con el siguiente nombre en el mismo lugar donde tenemos nuestros archivos .java y .class:

temp.mf

Ahora que tenemos todo listo para construir el .jar vamos a la línea de comandos (CMD) y ejecutamos la siguiente sentencia:

jar -cf HolaMundo.jar HolaMundo.class

Después de haber ejecutado la sentencia anterior ya tendremos el archivo HolaMundo.jar en nuestro directorio, y contendrá el fichero de manifiesto, la clase o clases que tenga nuestro proyecto, ahora al archivo .jar que tenemos, vamos a modificarlo para indicarle que archivo de manifiesto debe usar, esto con la siguiente sentencia:

jar cmf temp.mf HolaMundo.jar HolaMundo.class

Se permiten comodines (por ejemplo, para incluir todos los archivos class de la carpeta o que cumplan con alguna condición dada):

- *jar cmf temp.mf HolaMundo.jar fichero$.class*
- *jar cmf temp.mf HolaMundo.jar *.class*

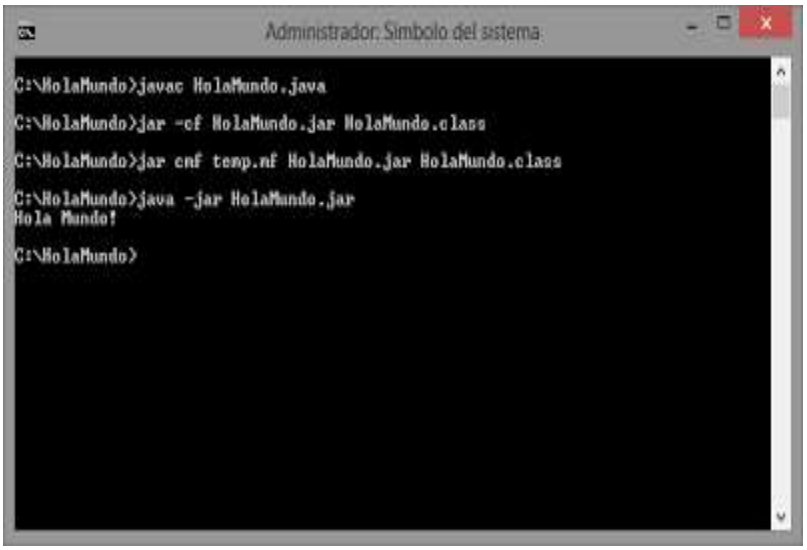
Y finalmente ejecutamos nuestro archivo en la línea de comandos con la siguiente llamada: ***java -jar HolaMundo.jar.***
Se adjunta una captura de todos los pasos antes mencionados al final del post, para que tengan una mejor idea del Proceso llevado a cabo, les recuerdo que solo empezamos con los archivos:

- HolaMundo.java
- temp.mf

Y a partir de la ejecución de los comandos terminamos con los siguientes archivos:

- HolaMundo.java
- temp.mf
- HolaMundo.class
- HolaMundo.jar

Cualquier duda la pueden hacer en los comentarios.



Tomado de

http://gl-epn-programacion-ii.blogspot.com/2013/02/como-crear-un-jar-con-java-desde-la.html.>

5 PISTAS DE APRENDIZAJE

Tenga en cuenta que al manejar medidas de almacenamiento debe saber cuáles son estas, que nombres reciben y cuáles son sus equivalencias en las otras medidas.

Tenga cuidado cuando convertimos de una medida mayor a una menor se multiplica y cuando convertimos de una medida menor a una mayor se divide.

No olvide que la mínima medida de medida en informática es el byte y todas las demás medidas se construyen teniendo esta como base.

Tenga presente que siempre que vaya a realizar una conversión debe saber:

1. Que nos dieron y a que lo vamos a convertir
2. Qué medida es mayor y cual es menor
3. Que operación debe realizar si una multiplicación o una división.
4. la regla tres simple o una operación directa

Tenga en cuenta que los juegos lógicos deben ser vistos como algo divertido.

Tenga cuidado al realizar o desarrollar cada juego seguir los pasos indicados.

No olvide leer y entender en consiste cada juego lúdico antes de sentarse a realizarlos o jugarlos.

Tenga presente que con los juegos lúdicos se pretende desarrollar habilidades de pensamiento para utilizarla en el desarrollo de software.

Tenga en cuenta que el lenguaje Java "Ayudan a resolver" y sistematizar "problemas de la vida real"

Tenga cuidado con el hecho de que la articulación entre los diferentes conceptos de programación y el lenguaje Java tiene implicaciones de alto impacto en la investigación y desarrollo de software.

No olvide que las teorías de la programación el lenguaje Java constituyen un campo del conocimiento con límites concretos que comprenden diversas formas de solucionar problemas en el desarrollo de software.

Tenga presente que al manejar el lenguaje Java deben ser manejados de una manera cauta cada una de sus estructuras para obtener una buena solución.

Para recordar

- **La orientación a objetos** es una técnica de modelado de sistemas, que pueden ser o no computacionales.

- **La clase** es un ente real o abstracto posible de ser programado.
- **Instancia** es la asignación de espacio en memoria por el operador new y el método constructor para que un objeto pueda ser utilizado.
- **Constructor** es un método que posee el mismo nombre de la clase y tiene como principio inicializar el objeto, es posible sobrecargar este método para establecer la asignación interna sin parámetro y la externa.
- **Destructor** método iniciado cuando se elimina un objeto de la memoria.

A tener en cuenta

Origen del lenguaje de programación orientado a Objeto se fundamenta en el paradigma Orientado a Objeto, es posterior al paradigma estructurado y es pre al paradigma orientado a aspectos.

- Lpoo
- http://es.slideshare.net/Karlytoz_36/programacin-orientada-a-objetos-15003678 youtube
- http://es.slideshare.net/NesMey/paradigma-orientado-a-objetos-4954115
- http://es.slideshare.net/NesMey/paradigma-orientado-a-objetos-4954115?next_slideshow=1

Estas características en la metodología orientada a objetos se establecen como fundamentales es necesaria su presencia en una aplicación de este tipo, las siguientes se consideran secundarias y pueden o no estar en nuestra comprensión:

POLIMORFISMO cuando dos métodos diferentes tienen igual el nombre en distintos objetos, esto es posible, pero solo se activara según el accionar o el llamado del mensaje o sea depende del momento de ejecución que es cuando se le asigna al método correspondiente.

Por ejemplo, un método llamado mostrar () puede responder de manera distinta dependiendo de quién lo invoque:

SOBRECARGA cuando los métodos poseen la misma identificación pero sus parámetros varían en sus tipos respectivos, su ubicación y en número.

REUSABILIDAD hace que un proceso o procesos de un programas de un determinado programador se pueda aplicar cuantas veces sea requerido por el o por otro dado el caso.

PERSISTENCIA cuando se define la duración del almacenamiento en memoria de los diferentes identificadores como temporal (solo mientras esta la aplicación), dinámica (una duración moderada por asignación interna) o permanente (dura por fuera de la aplicación como en archivos o en base de datos).

RECOLECCION DE BASURA consiste en recuperar los espacios de memoria que han sido liberados o no.

ELEMENTOS BÁSICOS DEL LENGUAJE DE PROGRAMACIÓN JAVA

http://es.slideshare.net/whaleejaa/elementos-bsicos-de-la-programacin-orientada-a-objetos

La máquina virtual de java garantiza la independencia de la plataforma lo cual permite utilizar el programa con solo una compilación y una interpretación, con una disminución de rendimiento pero es mejorado con la tecnología JIT(just in time compilaction) luego generara un código intermedio llamado bytecode, para ser utilizado en cualquier sistema operativo

Compilación proceso de verificar(detectar errores de sintaxis al transcribir el algoritmo a código del lenguaje de programación) todo el archivo y si no presenta errores llevarlo a lenguaje de maquina (binario)

Interprete proceso de verificar(detectar errores de sintaxis al transcribir el algoritmo a código del lenguaje de programación) línea por línea y si no presenta errores llevarlo a lenguaje de maquina (binario)

El JRE javaRuntime Environment Java runtime su función es incluir bibliotecas de código.

El JDK (Java Develpment Kit), kit de desarrollo de java facilita a los programadores de java la elaboración de aplicaciones (básicas y graficas API) y applets(para el explorador de internet).

Los directorios son las carpetas que se encuentran en el jdk con su respectiva versión y que contienen diferentes archivos binarios, documentos librerías, fuentes y demás por ejemplo la carpeta bin que contiene los archivos ejecutables entre ellos el javac y el java.

Los paquetes se encargan de agrupar las clases con características similares requieren ser importados para tener disponible dichas clases la orden debe estar al iniciar la aplicación antes de la clase su forma de escribir es la sentencia (orden) import java. paquete. * ; si coloca el carácter asterisco llamado comodín se dispondrá de todas las clases que contenga el paquete, si en lugar de * se especifica el nombre de la clase se dispone únicamente de esta asi import java. paquete. Clase especifica; una lista de paquetes es la siguiente:

La estructura básica de una aplicación en java

Identificadores son los nombres (formados preferiblemente por caracteres letras no se recomienda empezar por número) que se le asignan a cada uno de los elementos a utilizar en la aplicación, están clasificados así: de clase, de método, de variable y de constante

Los símbolos son caracteres regidos por la asociación Unicode

Los caracteres llamados bloques y separadores cumplen una función específica entre otros tenemos:

Abstract, boolean, byte , switch ,class , const, do , double ,else , enumer , extends, final, float, for,

If , import ,int , long, new ,return, short, static, super, this, while

Estructura de una aplicación en lenguaje de programación java

http://www.youtube.com/watch?v=l15tM7l_vs8

Para captura de datos se debe emplear hasta el momento una clase llamada Scanner que pertenece al paquete útil, por ser clase tiene unos métodos que le permiten convertir la información tomada desde el teclado y llevarla a la variable según su tipo el procedimiento será:

Importar el paquete

```
   public static void main(String [] arg)
{
Int N;// variable numero de encuestados
int estadocivil ;//declaramos la variable
int contasolt = 0; se declara y se inicializa en cero los contadores
int contacas = 0;
```

Ejecutar es poner a funcionar la aplicación

Esto está en el documento de la página webnode que se puede ajuntar

Ejercicio de Entrenamiento:

Programar en java lo siguiente:

Una empresa está dividida en N sedes, cada sede en M secciones y cada sección tiene un número no determinado de empleado. De cada empleado se conoce nombre, sexo, edad, salario básico y deducciones.

Elabore un algoritmo que permita calcular e imprimir.

a). El promedio de edad de los empleados por sección y por sede

b). El porcentaje de mujeres de 18 años por sección y por sede

C). El número de mujeres que ganan más de 4 salarios mínimos por sección

d). Un mensaje “si existe al menos un empleado hombre que gane más de 20 salarios mínimos” de toda la empresa

e). El nombre y la edad de la mujer más joven por sección

f). El nombre y la edad del hombre más viejo por sede

g). El valor de las deducciones más altas de toda la empresa

h). El promedio de salarios netos de las mujeres mayores de 18 años por sede y sección.

i). El porcentaje de empleados hombres que ganan menos de dos salarios mínimos por sección.

j). El valor del salario neto más bajo de toda la empresa.

6 GLOSARIO

Tuvo al vacío: es un componente electrónico utilizado para amplificar, conmutar, o modificar una señal eléctrica mediante el control del movimiento de los electrones en un espacio "vacío" a muy baja presión, o en presencia de gases especialmente seleccionados.

Transistor: es un dispositivo electrónico semiconductor utilizado para entregar una señal de salida en respuesta a una señal de entrada.

Circuito integrado: es una combinación de elementos de un circuito que están miniaturizados y que forman parte de un mismo chip o soporte. La noción, por lo tanto, también se utiliza como sinónimo de chip o microchip.

Procesador: también conocido como CPU o micro, es el cerebro del PC. Sus funciones principales son, entre otras, la ejecución de las aplicaciones y la coordinación de los diferentes dispositivos que componen el equipo.

Realidad virtual: es por lo general un mundo virtual generado por ordenador (o sistemas informáticos) en el que el usuario tiene la sensación de estar en el interior de este mundo, y dependiendo del nivel de inmersión este puede interactuar con este mundo y los objetos del mismo en un grado u otro.

Abaco: Se trata de un cuadro construido con madera que dispone de 10 alambres o cuerdas dispuestos de manera paralela. Cada uno de estos alambres o cuerdas, a su vez, cuenta con 10 bolas que pueden moverse. El ábaco, por lo tanto, es un instrumento que ayuda a realizar cuentas y cálculos simples.

Binario: es aquel que numera empleando sólo ceros (0) y unos (1). Esto quiere decir que, en el marco de estos sistemas, cualquier cifra puede expresarse a partir de estos números. Este sistema es utilizado por las computadoras u ordenadores, que funcionan con un par de voltajes diferentes y que atribuyen el 0 al apagado y el 1al encendido.

Decimal: el que utiliza las potencias de 10 como base para escribir números.

Octal: El sistema numérico en base 8 se llama octal y utiliza los dígitos 0 a 7.

Hexadecimal: El sistema hexadecimal (a veces abreviado como Hex, no confundir con sistema) es el sistema de numeración posicional que tiene como base el 16.

Bit: En informática y otras disciplinas, unidad mínima de información, que puede tener solo dos valores (cero o uno).

Byte: Conjunto de 8 bits que recibe el tratamiento de una unidad y que constituye el mínimo elemento de memoria direccionable de una computadora.

Kilobyte: Medida de la capacidad de memoria de una computadora que es igual a 1024 bytes.

Megabyte: Medida de la capacidad de memoria de una computadora que es igual a 1 millón de bytes.

Terabyte: Un terabyte es una unidad de almacenamiento de información cuyo símbolo es el TB, y equivale a 10^{12} bytes.

Sudoku: es un juego que tomo su poderío en Japón a principio de los años 80's, es una matriz de 9 x 9, (el formato más clásico de este juego), y consiste en una colección de 9 filas, 9 columnas, y 9 áreas más pequeñas, en las que no se deben de repetir número, símbolos o letras, la forma más tradicional se trabajó con números del 1 al 9,

Kakuro: es otro juego de origen Japonés, que nos permite desarrollar habilidades numéricas, en un proceso similar a un crucigrama, este tiene diferentes tamaños y formas y debe de arrojar un valor que nos pueda ser útil tanto en filas y columnas, sus principales condiciones es que solo acepta valores entre el 1 y el 9 sin repetir por bloque de números.

Tangram: es un juego de origen chino, que costa de 7 fichas 5 de ellas triángulos con las que se pretenden realizar una serie de figuras con la condicional de que no deben sobrar fichas, siempre se deben de utilizar las 7.

La torre de Hanoi: es una de las más representativas dentro de los aspectos lógicos por su apoyo a la recursividad, es un juego matemático que a medida que se ubican más discos, estos duplican la cantidad de movimientos de la opción anterior, si esta se inicia con 3 discos, los movimientos mínimos para solucionarla es de 7, pero si se colocan 4 discos su solución mínima es de 15 movimientos.

UML: es un lenguaje de Modelos Unificados por sus siglas en inglés, es el lenguaje de modelos de sistemas de software más popular en la actualidad, es un lenguaje para construir, especificar, visualizar y documentar sistemas de aplicativos.

Lenguaje de Programación: un sistema estructurado y diseñado principalmente para que las máquinas y computadoras se entiendan entre sí y con nosotros, los humanos. Contiene un conjunto de acciones consecutivas que el ordenador debe ejecutar.

Clase: se encarga de definir las propiedades y comportamiento de un tipo de objeto concreto. La instancia de lectura de estas definiciones y la creación de unos objetos nuevos a partir de ellas.

Herencia: Es la facilidad mediante la cual la clase A comparte en ella cada uno de los atributos y operaciones de B, como si esos atributos, operaciones y procesos se hubieran definidos en la primera clase A. en este caso se pueden usar los mismos métodos y variables publicas declaradas en la clase B.

Objeto: entidad compuesta por n conjunto de propiedades o atributos y de métodos, los mismos que a eventos. Es equivalente a los objetos reales del mundo que nos rodea.

Método: es la parte lógica que se aplica a la programación en cuestión, en el desarrollamos todos los procesos que el aplicativo requiere, está asociado a un objeto, pero la ejecución de este solo se mediante un "mensaje". Un método puede producir un cambio en las propiedades del objeto, o la generación de un "evento" con un nuevo mensaje para otro objeto del sistema.

Evento: Es un suceso que ocurre en el programa, tal como puede ser un clic, un doble clic etc. El sistema maneja el evento enviando el "mensaje" adecuado al objeto seleccionado.

Mensaje: es la forma de comunicarse con un objeto, mediante este se ordena que ejecute uno de sus métodos con parámetros asociados al evento que lo generó.

Propiedad o atributo: contiene un tipo de datos relacionado con un objeto, cuyo valor puede ser alterado por la ejecución de algún método.

Abstracción: Especifica las características fundamentales de un objeto, donde se captura su comportamiento. Cada objeto que posee el programa sirve como modelo abstracto, además de poder informar y cambiar su estado, y "comunicarse" con otros objetos en el sistema sin revelar *cómo* se implementan estas características.

Encapsulamiento: Es la característica de reunir a todos los elementos que pueden pertenecer a una misma entidad, al mismo nivel de abstracción. Se puede definir también como el principio de ocultación, principalmente porque se suelen emplear conjuntamente.

Polimorfismo: Se cataloga como un comportamiento diferente, asociado a objetos distintos, pueden tener el mismo nombre, al llamarlos por ese nombre se utilizará el comportamiento correspondiente al objeto que se esté usando. Esto indica que las referencias y las colecciones de objetos pueden tener objetos de diferentes tipos.

Recolección de basura: la Recolección de basura o GarbageCollector es la técnica por la cual el ambiente de Objetos se encarga de destruir automáticamente sus clases, objetos o métodos, este tipo de proceso anteriormente se conocía como constructor. Esto significa que el desarrollador no debe preocuparse por la asignación o liberación de memoria.

7 BIBLIOGRAFÍA

Fuentes bibliográficas

Brassard, G.; Bratley, P. (2012). *Fundamentos de algoritmia.* Bogota: Prentice Hall.
Anton, J; Ribas , L. (2004). *Introducción al Desarrollo de Software.*Barcelona: Eureca Media.
Evitts, P. (2000). *UML Pattern Language, A.* United States: New Riders Publishing.
Page-Jones, M; Wesley, A;. (2000). *Fundamentals of Object-Oriented Design in UML.*Massachusetts.
Peláez, J. (2007). *Análisis Y Diseño De Algoritmos: Un Enfoque Teórico Y Práctico.* Malaga España: Intercambio Científico.
Schneider, G; Winter, J; . (2001). *Applying Use Cases.*Boston: Pearson Education, Inc.
Tremblay, J; Karl, W;. (2010). *Fundamentos de Programación.* Pearson Education.

Fuentes digitales o electrónicas

ALGORITMIA, N. (2008). *Algortimos*. Obtenido de Estructura de Datos: http://www.algoritmia.net/
Caceres, D. (04 de 04 de 2008). *NTRODUCCIÓN PRÁCTICA AL DESARROLLO DE SOFTWARE DIRIGIDO POR MODELOS*. Obtenido de http://www.upct.es/contenido/estudios_postgrado/mostrar_curso.php?id_rec=170
Castiblanco, J. ((2010). *Historia De La Pc*. Obtenido de [imagen]: http://sistemaskmilo.blogspot.com/2010/08/historia-de-la-pc-mapa-conceptual.html
DumitruAlcantara. (2011). *Unidades de almacenamiento en informática*. Obtenido de [Imagen]: http://dumitrualcantara.blogspot.com/2011/10/unidades-de-almacenamiento-en.html
Geraldine. (2012). *Historia y evolución del Pc*. Obtenido de [imagen]: http://geraldine75.blogspot.com/
Gonzalez, R. (2010). *Mapa Conceptual Programación*. Obtenido de http://ramonesteban.blogspot.com/2010/08/semana-1-mapa-conceptual.html
Ihmc, T. (2014). *Mapa Conceptual Programación Orientada a Objetos*. Obtenido de http://cmapspublic.ihmc.us/rid=1191521763968_1644716976_9540/Programaci%C3%B3n%20orientada%20a%20objetos.cmap
Jabry. (2014). *Mapa mental conceptual que explique la materia de fundamentos de programación*. Obtenido de http://users6.nofeehost.com/kaos07/examen1.html
Latina, O. (2011). *Guia de UML*. Obtenido de http://www.osmosislatina.com/lenguajes/uml/
Matemáticas, F. (2014). *Sistema de numeración - un concepto / opinión personal*. Obtenido de [Imagen]: http://matematicasfps.wikispaces.com/Sistemas+de+Numeraci%C3%B3n
Modeller, T. c. (s.f.). *Elementos de UML*. Obtenido de Diagrama de casos de uso: https://docs.kde.org/stable/es/kdesdk/umbrello/uml-elements.html
nafsther0784. (2011). *Mapa Conceptual UML*. Obtenido de http://nafsther0784.wikispaces.com/UML+(Modelo+de+Lenguaje+Unificado)
Rodríguez, C. (28 de 07 de 2003). *Ejemplo de desarrollo software Utilzando la Metodología RUP*. Obtenido de Desarrollo de un Sistema para la Gestión de Articulos Deportivos: http://users.dsic.upv.es/asignaturas/facultad/lsi/ejemplorup
Torres, G. (2011). *Mapa Conceptual Programación*. Obtenido de http://cgerardotorres.blogspot.com/
Unad. (2014). *Mapa Conceptual Del Curso Académico De Lógica Formal Y Simbólica*. Obtenido de [imagen]: http://datateca.unad.edu.co/contenidos/103300/103300exe/mapa_conceptual.html

Brassard, G.; Bratley, P. (2012). Fundamentos de algoritmia. Bogota: Prentice Hall.
ALGORITMIA, N. (2008). Algortimos. Obtenido de Estructura de Datos: http://www.algoritmia.net/
Anton, J; Ribas , L. (2004). Introducción al Desarrollo de Software. Barcelona: Eureca Media.
Booch, G. (1995). Análisis y diseño orientado a objetos con aplicaciones. madrid: Dias de Santos/Addison Wesley.
Booch, G. (2001). Analisis y diseño orientado a Objetos con aplicaciones. mexico: s.a Alhambra mx.
Caceres, D. (04 de 04 de 2008). NTRODUCCIÓN PRÁCTICA AL DESARROLLO DE SOFTWARE DIRIGIDO POR MODELOS. Obtenido de http://www.upct.es/contenido/estudios_postgrado/mostrar_curso.php?id_rec=170
Castiblanco, J. ((2010). Historia De La Pc. Obtenido de [imagen]: http://sistemaskmilo.blogspot.com/2010/08/historia-de-la-pc-mapa-conceptual.html
Deitel, P. J., & Deitel, H. M. (2008). Java como programar. mex: pearson prentice hall 139789702611905.
DumitruAlcantara. (2011). Unidades de almacenamiento en informática. Obtenido de [Imagen]: http://dumitrualcantara.blogspot.com/2011/10/unidades-de-almacenamiento-en.html
Evitts, P. (2000). UML Pattern Language, A. United States: New Riders Publishing.
Geraldine. (2012). Historia y evolución del Pc. Obtenido de [imagen]: http://geraldine75.blogspot.com/
Gonzalez, R. (2010). Mapa Conceptual Programación. Obtenido de http://ramonesteban.blogspot.com/2010/08/semana-1-mapa-conceptual.html
Ihmc, T. (2014). Mapa Conceptual Programación Orientada a Objetos. Obtenido de http://cmapspublic.ihmc.us/rid=1191521763968_1644716976_9540/Programaci%C3%B3n%20orientada%20a%20objetos.cmap
Jabry. (2014). Mapa mental conceptual que explique la materia de fundamentos de programación. Obtenido de http://users6.nofeehost.com/kaos07/examen1.html
Johaspot. (7 de 10 de 2010). LA INGENIERIA DE SISTEMAS EN LA ACTUALIDAD. Obtenido de http://johaspot.blogspot.com.co/2010/10/la-ingenieria-de-sistemas-en-la.html
Joyannes, L. (2003). Fundamentos de Programación. Algoritmos y Estructuras de Datos y Objetos. Ed. McGraw-Hill.
Latina, O. (2011). Guia de UML. Obtenido de http://www.osmosislatina.com/lenguajes/uml/
Matemáticas, F. (2014). Sistema de numeración - un concepto / opinión personal. Obtenido de [Imagen]: http://matematicasfps.wikispaces.com/Sistemas+de+Numeraci%C3%B3n
Modeller, T. c. (s.f.). Elementos de UML. Obtenido de Diagrama de casos de uso: https://docs.kde.org/stable/es/kdesdk/umbrello/uml-elements.html
nafsther0784. (2011). Mapa Conceptual UML. Obtenido de http://nafsther0784.wikispaces.com/UML+(Modelo+de+Lenguaje+Unificado)
Page-Jones, M; Wesley, A;. (2000). Fundamentals of Object-Oriented Design in UML. Massachusetts.
Peláez, J. (2007). Análisis Y Diseño De Algoritmos: Un Enfoque Teórico Y Práctico. Malaga España: Intercambio Científico.
Pico, P. (02 de 2013). Definición de ingeniería de sistemas. Obtenido de http://ingenierodesistemas.co/editorial/definicion-de-ingenieria-de-sistemas/
Rodríguez, C. (28 de 07 de 2003). Ejemplo de desarrollo software Utilzando la Metodología RUP. Obtenido de Desarrollo de un Sistema para la Gestión de Articulos Deportivos: http://users.dsic.upv.es/asignaturas/facultad/lsi/ejemplorup
Schneider, G; Winter, J; . (2001). Applying Use Cases. Boston: Pearson Education, Inc.
Torres, G. (2011). Mapa Conceptual Programación. Obtenido de http://cgerardotorres.blogspot.com/
Tremblay, J; Karl, W;. (2010). Fundamentos de Programación. Pearson Education.

Unad. (2014). Mapa Conceptual Del Curso Académico De Lógica Formal Y Simbólica. Obtenido de [imagen]: http://datateca.unad.edu.co/contenidos/103300/103300exe/mapa_conceptual.html

Unad. (s.f.). Actualidad de la Ingeniería en Colombia. Obtenido de http://datateca.unad.edu.co/contenidos/90022/Modulo_2013_II/actualidad_de_la_ingeniera_en_colombia_2.html