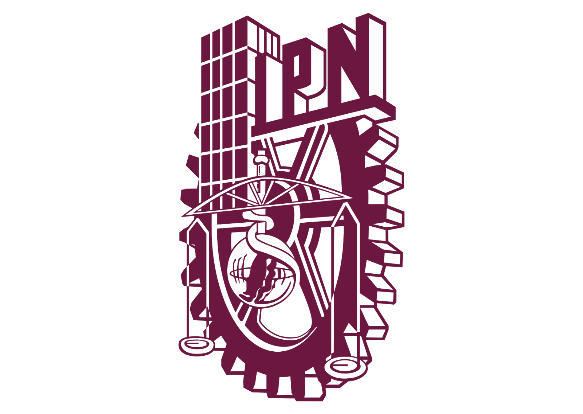
****

**INSTITUTO POLITECNICO NACIONAL**

**ESCUELA SUPERIOR DE INGENIERIA MECANICA Y ELECTRICA**

**INGENIERIA EN COMUNICACIONES Y ELECTRONICA**

**FUNDAMENTOS DE PROGRAMACION**

**PROF. OSCAR CRUZ**

**Actividad Inicial**

**Alumno:**

**DIAZ ANAYA EDUARDO**

**Boleta:**

**2020300206**

Desarrollar los temas de investigación para:

1. Que es un sistema operativo Historia Tipos de sistemas

2. Que es un compilador tipos de compiladores compilador C compilador C++

3. Que es un algoritmo Que son los algoritmos comunicacionales

4. Etapas de desarrollo de Software

1.- **¿Qué es un sistema operativo?**

Un sistema operativo es un conjunto de programas que permite manejar la memoria, disco, medios de almacenamiento de información y los diferentes periféricos o recursos de nuestra computadora, como son el teclado, el mouse, la impresora, la placa de red, entre otros.

Los periféricos utilizan un driver o controlador y son desarrollados por los fabricantes de cada equipo. Encontramos diferentes sistemas operativos como Windows, Linux, MAS OS, en sus diferentes versiones. También los teléfonos y tablets poseen un sistema operativo.

Dentro de las tareas que realiza el sistema operativo, en particular, se ocupa de gestionar la memoria de nuestro sistema y la carga de los diferentes programas, para ello cada programa tiene una prioridad o jerarquía y en función de la misma contará con los recursos de nuestro sistema por más tiempo que un programa de menor prioridad.  
El sistema operativo se ocupa también de correr procesos. Llamamos proceso a la carga en memoria de nuestro programa, si no está cargado en memoria nuestro programa simplemente “no corre”.

**Historia de los sistemas operativos**

La informática tal y como se le conoce hoy día, surgió a raíz de la II Guerra Mundial, en la década de los 40. En esos años no existía siquiera el concepto de "Sistema Operativo" y los programadores interactuaban directamente con el hardware de las computadoras trabajando en lenguaje máquina (esto es, en binario, programando únicamente con 0s y 1s).  
  
El concepto de Sistema Operativo surge en la década de los 50. El primer Sistema Operativo de la historia fue creado en 1956 para un ordenador IBM 704, y básicamente lo único que hacía era comenzar la ejecución de un programa cuando el anterior terminaba.  
  
En los años 60 se produce una revolución en el campo de los Sistemas Operativos. Aparecen conceptos como sistema multitarea, sistema multiusuario, sistema multiprocesadores y sistema en tiempo real.  
  
Es en esta década cuando aparece UNIX, la base de la gran mayoría de los Sistemas Operativos que existen hoy en día.

En los años 70 se produce un boom en cuestión de ordenadores personales, acercando estos al público general de manera impensable hasta entonces. Esto hace que se multiplique el desarrollo, creándose el lenguaje de programación C (diseñado específicamente para reescribir por completo el código UNIX).  
  
Como consecuencia de este crecimiento exponencial de usuarios, la gran mayoría de ellos sin ningún conocimiento sobre lenguajes de bajo o alto nivel, hizo que en los años 80, la prioridad a la hora de diseñar un sistema operativo fuese la facilidad de uso, surgiendo así las primeras interfaces de usuario.  
  
En los 80 nacieron sistemas como MacOS, MS-DOS, Windows.

En la década de los 90 hace su aparición Linux, publicándose la primera versión del núcleo en septiembre de 1991, que posteriormente se uniría al proyecto GNU, un sistema operativo completamente libre, similar a UNIX, al que le faltaba para funcionar un núcleo funcional. Hoy en día la mayoría de la gente conoce por Linux al Sistema Operativo que realmente se llama GNU/Linux

**Tipos de sistemas Operativos**

Los sistemas operativos se dividen en cuatro categorías.

1. **Sistema operativo de red o diálogo**. Este va de acuerdo con el modo operativo.
2. **Simple o multitarea**. Tiene que ver con el número de programas que se ejecutan al mismo tiempo.
3. **Usuarios únicos o múltiples**. Está relacionado con el número de usuarios.
4. **Procesador único o multiprocesador**. Toma en cuenta el número de procesadores

2.- ¿Qué es un compilador?

Un compilador es un pequeño programa informático, que se encarga de traducir (compilar) el código fuente de cualquier aplicación que se esté desarrollando. En pocas palabras, es un [software](http://culturacion.com/el-software-y-su-arquitectura-de-trabajo/) que se encarga de traducir el programa hecho en lenguaje de programación, a un lenguaje de máquina que pueda ser comprendido por el equipo y pueda ser procesado o ejecutado por este.

Un concepto un poco más elaborado es el siguiente: Un compilador es un programa que convierte o traduce el código fuente de un programa hecho en lenguaje de alto nivel, a un lenguaje de bajo nivel (lenguaje de máquina).

Tipo de compiladores

Existen varios tipos de compiladores, los cuales se adecuan a los requerimientos de los [programadores](http://culturacion.com/batch-una-forma-de-crear-scripts-basicos-para-windows/). Podemos destacar los compiladores cruzados, compiladores optimizadores, compiladores de una sola pasada y de varias pasadas, compiladores JIT y los conocidos [intérpretes](http://culturacion.com/que-es-un-lenguaje-de-programacion-interpretado/), que traducen el código del programa en tiempo real.

Cada uno de estos compiladores, se adecua a un lenguaje en específico o a varios. Algunos son de código abierto, otros son de licencia propietaria. La versatilidad de unos los hace mucho más populares que otros, pero a grandes rasgos, el escoger un determinado compilador dependerá de los requerimientos de cada usuario.

Compiladores en C / C++

En el caso del lenguaje C y su evolución orientada a objetos el C++, **los compiladores más conocidos** son GCC (GNU Compiler Collection), MinGW (implementación de GCC para [Windows](https://www.headsem.com/category/windows/)), los antiguos Turbo C y Turbo C++ que eran tanto compiladores como IDE para el sistema operativo MS-DOS ya fueron descontinuados, siendo sustituidos por C++Builder, y Visual C++ (que forma parte de Visual Studio).

Algunos de los compiladores mencionados anteriormente, pueden emplearse en combinación con algunos de los IDE a continuación, otros ya poseen su propio compilador incorporado. En cualquier caso, dentro de **los mejores IDE para programar con C/C++**, destacamos los siguientes:

**Dev-C++:** Este emplea el compilador MinGW. Se trata de un software libre, sencillo, ligero y eficiente, para la plataforma Windows.

**Code::Blocks:**Este es un software libre, multiplataforma. Code Blocks es una alternativa a Dev-C++ y desarrollada mediante el propio lenguaje C++. Sus capacidades son bastante buenas y es muy popular entre los nuevos programadores. Se puede encontrar separado del compilado o la versión “mingw” que incluye g++ (GCC para C++).

**Visual C++:**Posee editor de interfaces gráficas y una serie de asistentes que hacen muy cómodo el proceso de desarrollo.

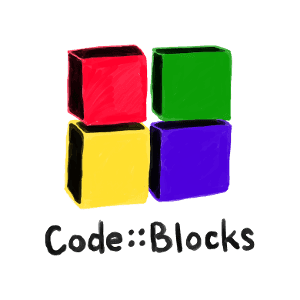
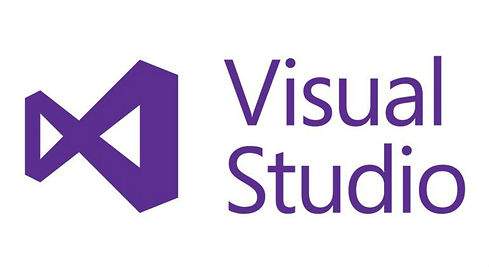
**C++Builder:** Esta es la opción desarrollada por Borland, es un software propietario y de pago destinado a la plataforma Windows.

**Eclipse:** Su principal propósito es programar mediante Java, también es libre y multiplataforma.

**MonoDevelop:** Esta es una alternativa a Visual Studio, pero este IDE es multiplataforma y de software libre. Posee un editor de interfaces gráficas que implementa la biblioteca GTK y es compatible con el .Net Framework de Microsoft.

**Xcode:** Este es uno de los mejores IDE para programar en Mac con el compilador GCC e Interface Builder.(MAC)

**KDevelop:** Este software de desarrollo está destinado exclusivamente para la plataforma GNU/Linux y otros sistemas Unix. No cuenta con un compilador propio por lo que requiere de su integración con GCC. (Linux)

**Anjuta:** Este software tiene como propósito principal utilizar herramientas proporcionadas por GTK+ para desarrollar [aplicaciones](https://www.headsem.com/category/aplicaciones/) para el escritorio GNOME. Esta opción es propia de los sistemas GNU/Linux y BSD. (Linux)

3.-¿Qué es un algoritmos?

La palabra **algoritmo**proviene del sobrenombre de un matemático árabe del siglo IX, **Al-Khwarizmi**, que fue reconocido por enunciar paso a paso las reglas para las operaciones matemáticas básicas con decimales (suma, resta, multiplicación y división).

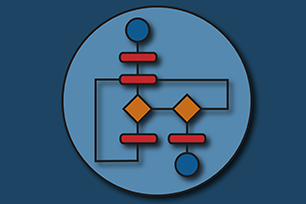
Vemos definición de algoritmo como un grupo de órdenes consecutivas que presentan una solución a un problema o tarea. Algunos ejemplos de algoritmos los podemos encontrar en las matemáticas (como el algoritmo para resolver una multiplicación) y en los manuales de usuario de un aparato (como una lavadora o una impresora).

Sin embargo, hoy en día se relaciona la palabra algoritmo con el mundo de la informática, más concretamente en la programación; los conocidos como **algoritmos informáticos**.

Un programa informático no es otra cosa que una serie compleja de algoritmos escritos en un lenguaje de programación que pueden ser ejecutados en un ordenador. Por lo tanto, el conocimiento de algoritmos en programación es esencial, por lo que es muy útil realizar un [**curso de introducción a la programación**](https://openwebinars.net/cursos/introduccion-programacion/) que incluya formación en algoritmos. Esta formación incluye conocimientos de pseudocódigo, que será un lenguaje escrito para representar los algoritmos antes de programarlos y de diagrama de flujos, que representa de manera simbólica los distintos pasos del algoritmo.

Un algoritmo informático es una secuencia de instrucciones finitas que llevan a cabo una serie de procesos para dar respuesta a determinados problemas. Es decir, un algoritmo informático resuelve cualquier problema a través de unas instrucciones y reglas concisas, mostrando el resultado obtenido.

Los algoritmos son muy importantes en el mundo de la informática ya que permiten al programador resolver el problema antes de escribirlo en un lenguaje de programación que entienda la máquina u ordenador. Antes de escribir el código de un programa hay que resolver con un algoritmo el problema que se nos plantea.



4.- Etapa de desarrollo de software

Etapa de análisis: Es el proceso de investigar un problema que se quiere resolver. Definir claramente el Problema que se desea resolver o el sistema que se desea crear. Identificar los componentes principales que integrarán el producto.

Etapa de Diseño: Es el proceso de utilizar la información recolectada en la etapa de análisis al diseño del producto. La principal tarea de la etapa de diseño es desarrollar un modelo o las especificaciones para el producto o Componentes del Sistema.

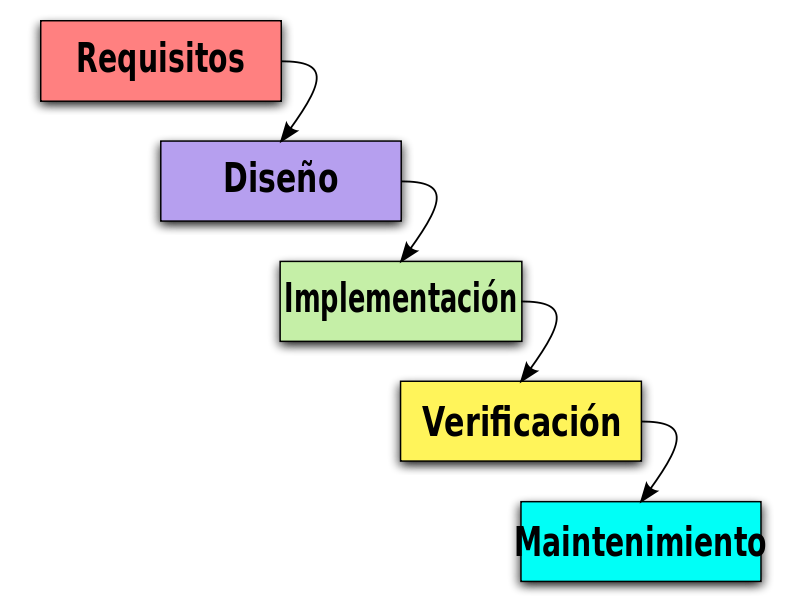
Etapa de Desarrollo: Consiste en utilizar los modelos creados durante la etapa de diseño para crear los componentes del sistema.

Etapa de Pruebas o Verificación Prueba : Consiste en asegurar que los componentes individuales que integran al sistema o producto, cumplen con los requerimientos de la especificación creada durante la etapa de diseño. Se recomienda aplicar las etapas: • Análisis • Diseño • Desarrollo • Prueba A cada uno de los ejercicios de este curso.

Etapa de Implementación o Entrega Implantación: Consiste en poner a disposición del cliente el producto.

Etapa de Mantenimiento: Consiste en corregir problemas del producto y re- liberar el producto como una nueva versión o revisión (producto mejorado).

Etapa final EOL (End-of-Life) El fin del ciclo del producto consiste en realizar todas las tareas necesarias para asegurar que los clientes y los empleados están conscientes de que el producto ya no será vendido ni soportado.



FUENTE.

<https://www.ceupe.com/blog/que-es-un-sistema-operativo.html>

<https://desarrollarinclusion.cilsa.org/tecnologia-inclusiva/que-es-un-sistema-operativo/>

<http://www.monografias.com/trabajos12/hisis/hisis.shtml>

<http://es.wikipedia.org/wiki/Historia_de_los_sistemas_operativos>

<http://culturacion.com/que-es-un-compilador/>

<http://www.headsem.com/mejores-ide-para-programar-en-c/#:~:text=En%20el%20caso%20del%20lenguaje,DOS%20ya%20fueron%20descontinuados%2C%20siendo>

<https://openwebinars.net/blog/que-es-un-algoritmo-informatico/>

<https://sites.google.com/site/fundamentosdeingendoftware/u1-fundamentos-ingenieria-de-software/1-3-etapas-del-desarrollo-software>