Unidad 1 y2 Programacion1

Introducción a la programación

Conceptos generales:

**Software:**

Sistema formal de un sistema informático, que comprende el conjunto de componentes lógicos que hacen posible la realización de tareas específicas.

El software es la parte lógica e intangible que poseen los sistemas informáticos.

El software utiliza el hardware para poder operar.

**Algoritmo:** Conjunto ordenado de operaciones sistemáticas que permite hacer un cálculo y hallar la solución de un tipo de problema.

Los algoritmos describen la solución a un problema en términos de los datos requeridos para representar el caso del problema y el conjunto de pasos necesarios para producir el resultado pretendido.

**Lenguaje de programación:** Un lenguaje de programación es un lenguaje formal que le proporciona a una persona, en este caso el programador, la capacidad de escribir una serie de instrucciones o secuencias de órdenes en forma de algoritmos. Los lenguajes de programación deben ser precisos.

**Pseudocódigo:** Mezcla de lenguaje de programación y léxico habitual en el que estamos trabajando, sin conocer lenguajes específicos.

**¿Qué es la programación?**

Programación es el proceso de tomar un algoritmo y codificarlo en un lenguaje de programación, de modo que pueda ser ejecutado por una computadora. Se conoce como programación al proceso de diseñar, codificar, depurar y mantener el código fuente de un programa (software).

**Fases de la resolución de un problema**

La teoría de la resolución de un problema de programación nos indica que existen 6 fases fundamentales en éste proceso, y éstas son:

1. **Análisis del problema:** Estudiamos y analizamos el problema hasta comprenderlo por completo y seguimos las especificaciones otorgadas por el cliente, empresa o persona que encarga el programa.

2. **Diseño del algoritmo**.

3. Codificación.

4. **Ejecución**, **verificación y depuración**: El programa se ejecuta y se revisa rigurosamente en busca de algún error.

5. **Mantenimiento:** Una vez que nuestro programa funciona perfectamente, éste se actualiza y se modifica según sea necesario.

6. **Documentación**: Se refiere a que el programa se documenta por completo desde el momento del análisis hasta la parte de mantenimiento

**Tipos de lenguajes de programación**

**Por el nivel:**

Existen tres tipos de lenguajes claramente diferenciados por nivel; el lenguaje máquina y los lenguajes de bajo nivel y los de alto nivel.

Lenguaje de Máquina: es el lenguaje de programación que entiende directamente la máquina (computadora).

Lenguaje de bajo nivel: Son mucho más fáciles de utilizar que el lenguaje máquina, pero dependen mucho de la máquina o computadora como sucede con el lenguaje máquina.

Lenguaje de alto nivel: Los lenguajes de programación de alto nivel son más fáciles de aprender porque se usan palabras o comandos del lenguaje natural, generalmente del inglés.

**Por la forma como se ejecutan:**

El programa escrito en un lenguaje de programación de alto nivel **(programa fuente)** no se puede ejecutar directamente en una computadora.

La opción más común es compilar el programa obteniendo un **código objeto**, aunque también, si el lenguaje lo soporta, puede ejecutarse en forma directa pero solo a través de un intérprete. A partir de esto entonces se diferencian:

 **Lenguajes compilados:** necesitan de un programa especial que lea el código fuente y cree un archivo binario ejecutable para una plataforma específica. Ejm: C, Pascal.

 **Lenguajes interpretados:** necesitan de un programa que traduzca en directo el código fuente escrito a instrucciones de la plataforma en la que se ejecutan.

Los primeros son más rápidos debido a que al compilar un programa las ordenes son más entendibles para la computadora, mientras que al interpretarlo la máquina primero debe leer el código y convertir al paso las instrucciones a instrucciones de máquina entendibles para ella.

**IDE**

Un entorno de desarrollo integrado (IDE), es un software que proporciona servicios integrales para facilitarle al programador el desarrollo de software.

Normalmente, un IDE consiste de un editor de código fuente, herramientas de construcción automáticas y un depurador (para ayudar a buscar errores en el software). La mayoría de los IDE tienen auto-completado inteligente de código (IntelliSense). Algunos IDE contienen un compilador, un intérprete, o ambos.

**Paradigmas de programación**

Paradigma imperativo

Los programas consisten en una sucesión de instrucciones o conjunto de sentencias, como si el programador diera órdenes concretas. El desarrollador describe en el código paso por paso todo lo que hará su programa.

Dentro del paradigma imperativo encontramos a

 Programación estructurada: es un tipo de programación imperativa donde el flujo de control se define mediante bucles anidados, condicionales y subrutinas, en lugar de a través de GOTO.

 Programación procedimental: Este paradigma de programación consiste en basarse en un número muy bajo de expresiones repetidas, englobarlas todas en un procedimiento o función y llamarlo cada vez que tenga que ejecutarse.

 Programación modular: consiste en dividir un programa en módulos o subprogramas con el fin de hacerlo más manejable y legible. Se trata de una evolución de la programación estructurada para resolver problemas de programación más complejos.

Paradigma declarativo

Este paradigma no necesita definir algoritmos puesto que describe el problema en lugar de encontrar una solución al mismo. Este paradigma utiliza el principio del razonamiento lógico para responder a las preguntas o cuestiones consultadas.

Este paradigma a su vez se divide en dos:

 Programación Lógica: El problema se modela con enunciados de lógica de primer orden. Ejemplo: Prolog

 Programación funcional: Los programas se componen de funciones, es decir, implementaciones de comportamiento que reciben un

Programación orientada a objetos

En este modelo de paradigma se construyen modelos de objetos que representan elementos (objetos) del problema a resolver, que tienen características y funciones. Permite separar los diferentes componentes de un programa, simplificando así su creación, depuración y posteriores mejoras. La programación orientada a objetos disminuye los errores y promociona la reutilización del código.

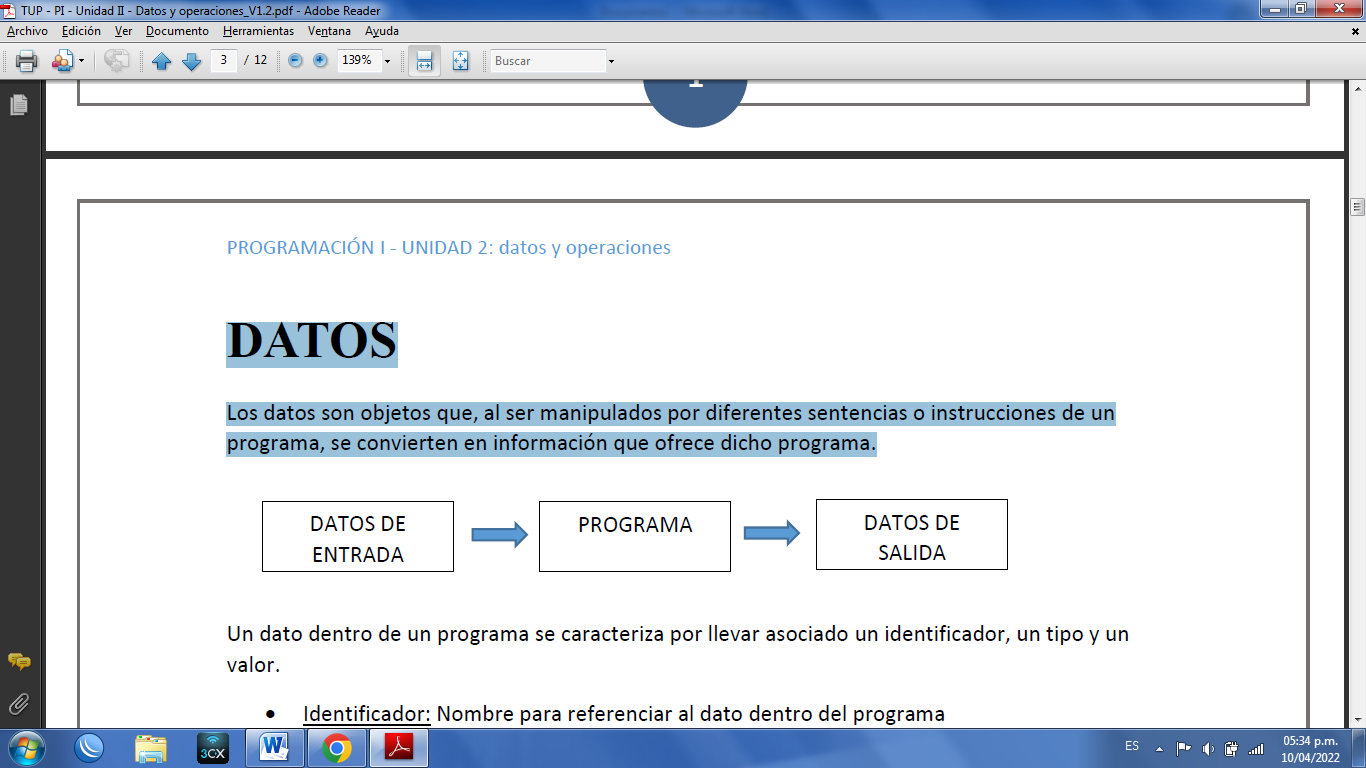
Programación reactiva

Este paradigma se basa en **escuchar lo que emite un evento o cambios en el flujo de datos**, en donde los objetos reaccionan a los valores que reciben de dicho cambio.

Datos y operaciones

**DATOS**

Los datos son objetos que, al ser manipulados por diferentes sentencias o instrucciones de un programa, se convierten en información que ofrece dicho programa.



Un dato dentro de un programa se caracteriza por llevar asociado un identificador, un tipo y un valor.

 Identificador: Nombre para referenciar al dato dentro del programa

 Tipo: el tipo de un dato determina el rango de valores que puede tomar el dato y su ocupación en memoria durante la ejecución del programa

 Valor: Sera un elemento determinado dentro del rango de valores permitidos por el tipo de dato definido.

**IDENTIFICADORES**

Si se parte del hecho de que programar es solucionar problemas a través de variables, entonces un programa tiene algunas variables, pero también existen constantes.

Cada variable, constante e incluso algún proceso requieren un nombre que los identifique, y de ahí la palabra "identificadores", por lo que puede decirse que un identificador es el nombre de cualquier variable, constante, procedimiento o programa.

**Variable:** Una variable puede tomar distintos valores a lo largo de la ejecución de un programa.

**Constante:** Una constante posee un solo valor fijo en toda la ejecución del programa. Supongamos que tenemos que desarrollar un método que recibe como entrada el radio de un círculo y debe calcular el área del mismo.

**NOMENCLATURAS DE NOMBRES DE IDENTIFICADORES**

La escritura de un identificador tiene reglas, las cuales deben aplicarse siempre en el momento de crear alguno de los elementos descritos.

"Un identificador debe iniciar con una letra, puede estar seguido de letras y/o números, no debe tener espacios en blanco, ni caracteres especiales, excepto el guion bajo (\_) o el guion medio (-)"

Actualmente existen muchos convenios para asignar un nombre apropiado al identificador.

Buenas prácticas para la asignación de un identificador:

**1. El nombre debe ser representativo**

**2. No utilizar caracteres especiales (excepto “-” y “\_”)**

**3. Empezar siempre con una letra**

**4. Si el identificador consta de dos palabras, entonces utilizar mayúsculas para separar las palabras, o bien “\_” o bien “–“**

**TIPOS DE DATOS**

**Estándares**

 **Real (Float o Double en C):** Ejemplo, el saldo de una cuenta bancaria

 **Entero (Int en C):** Ejemplo, la edad de una persona

 **Caracter (Char en C):** Representa un único caracter. Ejemplo, podemos usarlo para identificar el estado civil de una persona, si el dato guarda solo los caracteres ’c’,’d’,’s’,’v’

 **Cadena de texto (Char[] en C):** Ejemplo el nombre de una persona

 **Lógico (Bool en C):** Los valores siempre indican o verdadero o falso. Ejemplo se puede usar para identificar si una persona es estudiante o no