TALLER NUMERO 1 – POO

SEBASTIAN ZAPATA MARIN

CORPORACION DE ESTUDIOS TECNOLOGICOS DEL NORTE DEL VALLE

INFORMATICA EMPRESARIAL

CARTAGO VALLE

2017

1-**Resolución de problemas de programación**

La resolución de un problema mediante un ordenador consiste en el proceso que a partir de la descripción de un problema, expresado habitualmente en lenguaje natural y en términos propios del dominio del problema, permite desarrollar un programa que resuelva dicho problema.

Este proceso exige los siguientes pasos:

* Análisis del problema.
* Diseño o desarrollo de un algoritmo.
* Transformación del algoritmo en un programa (codificación).
* Ejecución y validación del programa.

Los dos primeros pasos son los más difíciles del proceso. Una vez analizado el problema y obtenido un algoritmo que lo resuelva, su transformación a un programa de ordenador es una tarea de mera traducción al lenguaje de programación deseado.

2- **COMPILADORES E INTÉRPRETES**

* Compilador, que analiza el programa fuente y "lo traduce" a otro equivalente escrito en otro lenguaje (por ejemplo, en el lenguaje de la máquina). Su acción equivale a la de un traductor humano, que toma un libro y produce otro equivalente escrito en otra lengua.
* Intérprete, que analiza el programa fuente y "lo ejecuta directamente", sin generar ningún código equivalente. Su acción equivale a la de un intérprete humano, que traduce las frases que oye sobre la marcha, sin producir ningún escrito permanente. Intérpretes y compiladores tienen diversas ventajas e inconvenientes que los hacen complementarios:
* Un intérprete facilita la búsqueda de errores, pues la ejecución de un programa puede interrumpirse en cualquier momento para estudiar el entorno (valores de las variables, etc.). Además, el programa puede modificarse sobre la marcha, sin necesidad de volver a comenzar la ejecución.
* Un compilador suele generar programas más rápidos y eficientes, ya que el análisis del lenguaje fuente se hace una sola vez, durante la generación del programa equivalente. En cambio, un intérprete se ve obligado generalmente a analizar cada instrucción tantas veces como se ejecute (incluso miles o millones de veces).
* Un intérprete permite utilizar funciones y operadores más potentes, como por ejemplo ejecutar código contenido en una variable en forma de cadenas de caracteres. Usualmente, este tipo de instrucciones es imposible de tratar por medio de compiladores. Los lenguajes que incluyen este tipo de operadores y que, por tanto, exigen un intérprete, se llaman interpretativos. Los lenguajes compilativos, que permiten el uso de un compilador, prescinden de este tipo de operadores. En pocas palabras "el compilador sólo traduce; el intérprete decodifica y ejecuta. "

3-La programación orientada a objetos (POO, u OOP según sus siglas en inglés) es un paradigma de programación que viene a innovar la forma de obtener resultados. Los objetos manipulan los datos de entrada para la obtención de datos de salida específicos, donde cada objeto ofrece una funcionalidad especial.

Muchos de los objetos pre-diseñados de los lenguajes de programación actuales permiten la agrupación en bibliotecas o librerías, sin embargo, muchos de estos lenguajes permiten al usuario la creación de sus propias bibliotecas.

Está basada en varias técnicas, incluyendo herencia, cohesión, abstracción, polimorfismo, acoplamiento y encapsulamiento.

Su uso se popularizó a principios de la década de 1990. En la actualidad, existe una gran variedad de lenguajes de programación que soportan la orientación a objetos.

4-En el paradigma de programación orientada a objetos (POO, o bien OOP en inglés), un objeto es una unidad dentro de un programa de computadora que consta de un estado y de un comportamiento, que a su vez constan respectivamente de datos almacenados y de tareas realizables durante el tiempo de ejecución. Un objeto puede ser creado instanciando una clase, como ocurre en la programación orientada a objetos, o mediante escritura directa de código y la replicación otros objetos, como ocurre en la programación basada en prototipos.

Estos objetos interactúan unos con otros, en contraposición a la visión tradicional en la cual un programa es una colección de subrutinas (funciones o procedimientos), o simplemente una lista de instrucciones para el computador. Cada objeto es capaz de recibir mensajes, procesar datos y enviar mensajes a otros objetos de manera similar a un servicio.

En el mundo de la programación orientada a objetos (POO), un objeto es el resultado de la instanciación de una clase.1 Una clase es el anteproyecto que ofrece la funcionalidad en ella definida, pero ésta queda implementada sólo al crear una instancia de la clase, en la forma de un objeto. Por ejemplo: dado un plano para construir sillas (una clase de nombre clase\_silla), entonces una silla concreta, en la que podemos sentarnos, construida a partir de este plano, sería un objeto de clase\_silla. Es posible crear (construir) múltiples objetos (sillas) utilizando la definición de la clase (plano) anterior. Los conceptos de clase y objetos son análogos a los de tipo de datos y variable; es decir, definida una clase podemos crear objetos de esa clase, igual que disponiendo de un determinado tipo de dato (por ejemplo el tipo entero), podemos definir variables de dicho tipo:

int a,b;

( 'int' es un tipo de dato y 'a' y 'b' son variables de tipo entero con las que podemos operar)

Para utilizar la funcionalidad definida en una clase en particular (salvo en las clases abstractas), primeramente es necesario crear un objeto de esa clase. De la misma manera, para una persona que desea sentarse, las especificaciones para construir una silla serán de poca utilidad; lo que se necesita es una silla real construida a partir de esas especificaciones. Siguiendo con la analogía anterior, también se puede decir que para hacer operaciones aritméticas, de nada sirve por sí solo el tipo entero (int); para ello necesitamos variables (o constantes) con las que operar.

5-Transcripción de Solución de problemas con enfoque orientado a objetos

**Etapas**

• Definición del problema o dominio del problema.

• Análisis orientado a objetos y diseño orientado a objetos.

• Programación.

• Documentación.

**Definición del problema o dominio del problema**

Describir en forma narrativa o esquemática, de modo claro y concreto el problema a resolverse.

Ejemplo:

Definición o dominio del problema:

Leer dos lados de un triángulo, calcular la hipotenusa H = √a2 + b2 y presentar el resultado.

**Análisis orientado a objetos y diseño orientado a objetos**

-identificar datos y métodos. AOO

-definir objetos lógicos (clases) DOO

Para esta etapa utilizar UML, el DOO implica reexaminar los objetos del dominio del problema para refinarlos, extenderlos y reorganizarlos.

**Diagramas de clases**

Describir entidad, nombre de la clase atributos, métodos

-Atributos

Atributos. Describen las características de los objetos: tipo de acceso

(Privado, protegido, publico) y tipo de dato (entero, real, booleano, etcétera).

**Métodos**.

¿Qué puede hacer la clase?

Realizar un proceso o tarea específicos

Tipo de acceso, Tipo de retorno, nombre del método, parámetros, y el cuerpo del método

**Tipos de acceso:**

Publico representado con el signo + accesibles para cualquier parte

Privado representado con el signo - accesibles en métodos que pertenecen en una clase

Protegido representado con el signo # para herencias

6-En ciencias de la computación un tipo de dato informático o simplemente tipo es un atributo de los datos que indica al ordenador (y/o al programador) sobre la clase de datos que se va a trabajar. Esto incluye imponer restricciones en los datos, como qué valores pueden tomar y qué operaciones se pueden realizar.

Los tipos de datos comunes son: números enteros, números con signo (negativos), números de coma flotante (decimales), cadenas alfanuméricas (y unicodes), estados, etc.

**7-Constantes**

Una constante es un dato numérico o alfanumérico que no cambia durante la ejecución del programa.

Ejemplo:

pi = 3.1416

**Variable**

Es un espacio en la memoria de la computadora que permite almacenar temporalmente un dato durante la ejecución de un proceso, su contenido puede cambiar durante la ejecución del programa.

Para poder reconocer una variable en la memoria de la computadora, es necesario darle un nombre con el cual podamos identificarla dentro de un algoritmo.

Ejemplo:

area = pi \* radio ^ 2

Las variables son : el radio, el area y la constate es pi

**Clasificación de las Variables**

**Por su contenido**

Variables Numéricas: Son aquellas en las cuales se almacenan valores numéricos, positivos o negativos, es decir almacenan números del 0 al 9, signos (+ y -) y el punto decimal.

Ejemplo:

iva = 0.15 pi = 3.1416 costo = 2500

Variables Lógicas: Son aquellas que solo pueden tener dos valores (cierto o falso) estos representan el resultado de una comparación entre otros datos.

Variables Alfanuméricas: Esta formada por caracteres alfanuméricos (letras, números y caracteres especiales).

Ejemplo:

letra = a apellido = lopez direccion = Av. Libertad #190

**Por su uso**

Variables de Trabajo: Variables que reciben el resultado de una operación matemática completa y que se usan normalmente dentro de un programa.

Ejemplo:

Suma = a + b /c

Contadores: Se utilizan para llevar el control del numero de ocasiones en que se realiza una operación o se cumple una condición. Con los incrementos generalmente de uno en uno.

Acumuladores: Forma que toma una variable y que sirve para llevar la suma acumulativa de una serie de valores que se van leyendo o calculando progresivamente.

**Expresiones**

Las expresiones son combinaciones de constantes, variables, símbolos de operación, paréntesis y nombres de funciones especiales.

Por ejemplo:

a + (b + 3) / c

Cada expresión toma un valor que se determina tomando los valores de las variables y constantes implicadas y la ejecución de las operaciones indicadas.

Una expresión consta de operadores y operandos. Según sea el tipo de datos que manipulan, se clasifican las expresiones en:

* Aritméticas
* Relacionales
* Lógicas

8-

9-El lenguaje unificado de modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el Object Management Group (OMG).

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.

Es importante remarcar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo.

Se puede aplicar en el desarrollo de software gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional, Rational Unified Process o RUP), pero no especifica en sí mismo qué metodología o proceso usar.

UML no puede compararse con la programación estructurada, pues UML significa Lenguaje Unificado de Modelado, no es programación, solo se diagrama la realidad de una utilización en un requerimiento. Mientras que programación estructurada es una forma de programar como lo es la orientación a objetos, la programación orientada a objetos viene siendo un complemento perfecto de UML, pero no por eso se toma UML solo para lenguajes orientados a objetos.

UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas.

10-Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como WORA, o "write once, run anywhere"), lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Java es, a partir de 2012, uno de los lenguajes de programación más populares en uso, particularmente para aplicaciones de cliente-servidor de web, con unos 10 millones de usuarios reportados.

El lenguaje de programación Java fue originalmente desarrollado por James Gosling de Sun Microsystems (la cual fue adquirida por la compañía Oracle) y publicado en 1995 como un componente fundamental de la plataforma Java de Sun Microsystems. Su sintaxis deriva en gran medida de C y C++, pero tiene menos utilidades de bajo nivel que cualquiera de ellos. Las aplicaciones de Java son generalmente compiladas a bytecode (clase Java) que puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente.

La compañía Sun desarrolló la implementación de referencia original para los compiladores de Java, máquinas virtuales, y librerías de clases en 1991 y las publicó por primera vez en 1995. A partir de mayo de 2007, en cumplimiento con las especificaciones del Proceso de la Comunidad Java, Sun volvió a licenciar la mayoría de sus tecnologías de Java bajo la Licencia Pública General de GNU. Otros también han desarrollado implementaciones alternas a estas tecnologías de Sun, tales como el Compilador de Java de GNU y el GNU Classpath.

<https://es.wikipedia.org/wiki/Resoluci%C3%B3n_de_problemas_de_programaci%C3%B3n>

<https://funprogramacion.wikispaces.com/Compiladores+e+Int%C3%A9rpretes>

<https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos>

<https://es.wikipedia.org/wiki/Objeto_(programaci%C3%B3n)>

<https://prezi.com/37q-cj1fvqdp/solucion-de-problemas-con-enfoque-orientado-a-objetos/>

<https://es.wikipedia.org/wiki/Tipo_de_dato>

<https://desarrolloweb.com/articulos/2164.php>

<https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado>

<https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)>