Una **clase** en java, es una plantilla donde vamos a escribir el código.

Lo primero que vamos a escribir, es el nombre de nuestra clase,

Se lee de arriba hacia abajo y de DERECHA a izquierda.

El método main (en todo lenguaje de programación) es donde va a dar el inicio al programa, lo que permite que arranque tu programa

Indentado de código es lo que permite dar espacio o sangrías, para que quede prolijo

Siempre que termines una línea de instrucción debemos de poner un “;”

Para escrbir un texto en programación SIEMPRE debe ir entre comillas (“”)

Al momento de crear el archivo .java, debe de tener el mismo nombre que la clase, si no, no va a correr el programa

**Tipos de datos en java**

Primitivos: Enteros: (byte, short, int, long) nos permite alojar valores numéricos enteros, no puede haber decimales, la diferencia entre ellos es la cantidad que pueden alojar

Decimales: float nos va a permitir alojar hasta 8 valores después del punto a la derecha. Y double nos va a permitir alojar hasta 15 valores después del punto a la derecha

Otros: char nos permite alojar solamente un carácter, pero es indiferente si es numero o de texto. boolean nos va a permitir alojar dos tipos de elementos, los que son true o false, estos dos nos van a permitir tomar decisiones.

Tipos Objeto estos tipos de datos se caracterizan por hay que invocarlos. String (son clases y empiezan con mayúsculas) nos permite crear cadenas de textos, todo lo que tenga que ver con texto lo vamos a tratar con String

Variable es un espacio en memoria donde nosotros vamos a poder alojar información o datos ya sean de tipo numérico o texto, para declarar una variable en java, necesitamos de dos elementos primero necesitamos un tipo de dato y en segundo lugar necesitamos de un nombre, por el tipo de dato nosotros vamos a decir al espacio en memoria que clase de información o dato va a contener y con el nombre le vamos a decir a nuestro programa a qué tipo de variable o espacio en memoria estamos haciendo referencia. El nombre de la variable nunca puede ir separado (num1) si lo queres separar hay que utilizar un (“-“ num-1). Inicializar una variable o espacio en memoria, significa darle un valor de inicio o de arranque. A una variable no la puedes poner entre comillas, independientemente de que tu variable sea tipo numérico o de texto, no la puedes poner entre comillas porque es una cadena de texto, lo que tenemos que usar es un signo de “+” se le conoce como signo de concatenación, cuando tenemos dos elementos diferentes, debemos de agregar esta concatenación porque si no, no podremos mostrar lo que está dentro de la variable.

**Operaciones aritméticas en programación**

Siempre lo primero que se va a resolver es lo que está en paréntesis, potencias, y de ahí para abajo

**Estructuras condicionales con if-else**

Una estructura condicional es una instrucción que le ayuda a nuestro programa a saber qué proceso o camino debe de seguir. Una condición es la que rige el comportamiento de la estructura condicional. Tenemos dos tipos de estructuras condicionales, simples y compuestas, las simples únicamente consisten en que unos de los dos caminos contienen instrucciones, por otro lado, las compuestas van a tener instrucciones por ambos lados.

**Estructuras condicionales anidadas con if-else**

Una estructura condicional se convierte en anidada cuando está dentro de otra estructura condicional. Cuando hay un solo “=” le estamos diciendo al programa que ese valor tiene a la derecha debe guardarse dentro del lugar que tiene a la izquierda, en cambio cuando hay “==” le estaríamos diciendo que tiene que **COMPARAR**

**Librerías**

Es un conjunto de clases que contienen atributos y métodos. Es un conjunto de código escrito por alguien más, que nos va a facilitar el trabajo. Scanner es una librería que nos va permitir introducir desde el teclado en el programa. Antes de empezar a escribir nuestra línea de código, hay que importar la librería, en el caso de que vayamos a utilizarla. Cuando vayamos a utilizar una librería en java, no basta con solo importarla, sino que hay que especificar que la vamos a usar. Para hacer un objeto de tipo Scanner es como declarar una variable, hay que indicarle al programa que tipo de objeto va a ser, entonces vamos a indicar que vamos a usar “Scanner” in (“in” es el nombre que le asignamos a ese objeto, a este mismo objeto hay que inicializarlo, para esto vamos a realizar lo siguiente le vamos a decir que es un nuevo objeto de tipo scanner por ende queda así, “Scanner in = new Scanner(System.in);” system.in los que nosotros le estamos diciendo al programa es que nos va a poder permitir ingresar datos desde el teclado, así que siempre que se quiera realizar esta acción debemos de poner la palabra “in” . Cuando nosotros vamos a introducir datos debemos llamar a nuestro objeto, posteriormente vamos a poner este método “nextLine”, cuando vayamos a alojar lo que el usuario escriba dentro de una variable de tipo texto tenemos que poner nextLine, si nosotros fuéramos a guardar una variable de tipo numérico en caso de que fueran enteros, debemos de poner nextInt.

**Condiciones compuestas con operadores lógicos y relacionales**

Operadores relacionales van a trabajar únicamente cuando los coloquemos entre dos elementos, si los colocamos dentro de una condición van a convertirse en una pregunta en vez de una afirmación

Operadores lógicos, estos nos van a permitir una mejor toma de decisión a nuestro programa siempre y cuando tengamos estructuras condicionales dentro del mismo. Tenemos 3 tipos, vamos a ver 2 de momento, “&&” (and) a este se lo conoce como aperson lo que nos va a permitir es realizar comparaciones, y decirle que una condición debe de cumplir tanto la del lado derecho, como la del lado izquierdo. El otro es “||” a este se lo conoce como piper, también nos va a comparar dos condiciones, con la diferencia que se tiene que cumplir con una de las condiciones ya sea la de la derecha o izquierda.

**Estructuras de control con switch – Case**

Esto se utiliza para agilizar la toma de decisiones múltiples su funcionamiento es similar al de un menú de opciones. Para poder usar estas estructuras debemos decirle al programa que vamos a utilizarlo, para hacer esto debemos de poner “switch(){}”, debemos de poner las opciones dentro de nuestro switch y para eso vamos a poner “case” y especificar las opciones al terminar dichas opciones hay que cerrar con un “;” posteriormente a esto, debemos de poner un “break;” lo que hace esto es decirle al programa que ya se terminó de ejecutar el caso y la opción que se estaba usando. No tenemos un límite de “case” podemos poner todas las que sean necesarias, pero en cada una de ellas hay que cerrar con un “break”. Tenemos la opción “default” la que nos permite si el usuario cargo la opción que no está dentro del switch que envie o haga una opción determina

**Ciclos o bucles con estructuras repetitivas – for- while**

Es un sistema repetitivo hasta que se cumpla la condición, para poder establecer un ciclo nosotros debemos de darle una condición, después darle una instrucción. Al momento de programar se va a utilizar for, para esto debemos de indiciar que lo vamos a usar, luego de ello vamos a abrir paréntesis y dentro de ellos debemos de tener un inicio, una condición y un incremento. for(inicio; condicion; incremento).

**Ciclos o bucles con estructuras repetitivas – do- while**

Esta estructura se divide en dos secciones la parte en donde le vamos a decir que es lo que tiene que realizar y la parte de las condiciones, que es la parte en la que vamos a controlar en qué momento se tiene que detener este siclo. Primero debemos de escribir “**do**” en minúscula a continuación abrimos y cerramos una llave, vamos a colocar o escribir las instrucciones que vamos a querer que se ejecute, posteriormente afuera de esta llave, vamos a escribir la palabra “**while**” y colocar entre paréntesis la condición que vamos a querer que se cumpla finalizando con un “;”.

**Cadena de caracteres**

Una cadena de caracteres es una secuencia de los mismos, que puede contener “símbolos, letras y números”. “**equals**” esto lo que hace, es indicarle al programa que quieres comparar una cadena de caracteres en específico.

Método “**length**”indica cuantos caracteres pose una cadena de caracteres. También tenemos el método “**substring**” el cual nos permite obtener una parte en específico de la cadena de caracteres hay que indicarle desde y hasta donde queremos obtener nuestra cadena de caracteres, **Substrin(desde,hasta);**

**Arreglos unidimensionales (Vectores)**

Es una estructura de datos que almacena bajo el mismo nombre a una colección de datos. Es decir que cuando nosotros declaramos un arreglo, estamos abriendo un espacio en memoria como hacemos con las variables, con la diferencia que un arreglo esta seccionado en una parte o en varias.

Los arreglos se caracterizan por almacenar elementos de posiciones contiguas en memoria. Es decir, los elementos están uno al lado del otro. Tienen un mismo nombre de variable que representan a todos los elementos. Los elementos son asignados a una posición dentro del vector y los arreglos trabajan en conjuntos con índices que especifican la posición de cada elemento dentro del arreglo. Tipos de arreglos, tenemos dos, primero tenemos los arreglos unidimensionales mejor conocidos como “Vectores” y en segundos lugar tenemos a los bidimensionales mejor conocidos como “Matrices”. Un Vector se le llama unidimensional, porque es una lista de una sola dimensión, es decir que solo podemos guardar datos de atrás hacia adelante o de adelante hacia atrás, su longitud depende del número de posiciones que nosotros asignamos al momento de declarar el vector, siempre la posición inicial de un vector es 0 indiferentemente del número de posiciones que tenga. Los vectores trabajan en conjunto con un índice que se representa gráficamente con una flecha, la misma tiene que ser declarada como una variable de tipo entero en consecuencia debe de tener un nombre y un valor de inicio o arranque, este índice nos permite recorrer todo el arreglo tanto de ida como de regreso, lo cual nos va a permitir consultar, modificar eliminar o insertar algún elemento en cualquier posición del vector. Sintaxis para programar un vector, al igual que en una variable a un vector hay que declararlo e indicarle el tipo de datos que vamos a manejar, ya que todos los elementos que ingresen deben de ser del mismo tipo de dato, posteriormente hay que asignarle un nombre, una vez asignado el nombre ponemos corchetes “[]” luego hay que inicializar el vector para lo cual hay que colocar un signo de igual “=” delante de los corchetes anteriormente colocados, escribimos la palabra “new” y nuevamente escribimos el tipo de datos que vamos a utilizar seguido de otro par de corchetes dentro el cual va a tener el número de posiciones que deseamos que contenga nuestro vector, finalizado de un “;”. Para ingresar valores en nuestro vector, vamos a necesitar de nuestro índice, el cual tiene comportamiento dinámico y estático, este último no es necesario declararlo, ya que basta con indicarle el nombre y la posición del vector que queremos que se dirija, para realizar esto, debemos de indicar el nombre de nuestro vector “**numeros** [**2**] = **8**;” donde el número “2” es la posición y el “8” el elemento a guardar. Para imprimir datos de un vector, necesitamos la siguiente línea de código “System.out.println(**numeros[2]);**”. Vectores dinámicos, a diferencia de los vectores estáticos, a estos no hace falta especificarle que posiciones debe de recorrer, ya que esto lo va a hacer de manera automática, de ida y de regreso, esto lo vamos a hacer con una estructura repetitiva “for”.

**Arreglos bidimensionales (Matrices)**

Un arreglo en java puede tener más de una dimensión, a esto se lo conoce como matriz, al igual que un vector, una matriz es un espacio en memoria seccionado o dividido en varias partes, con la diferencia que en una matriz podemos guardar datos en dos dimensiones, es decir, de lado a lado y de arriba hacia abajo. Una Matriz cuenta con varias filas y varias columnas. Para poder utilizar una matriz en java, debemos de tener en cuenta que necesitamos dos índices

Matrices dinámicas. Al igual que los vectores pueden tener un comportamiento dinámico. Para ellos vamos a recurrir a la estructura repetitiva “for” como en las matrices tenemos dos índices, para realizar esto de forma dinámica, vamos a utilizar un “for”, dentro de otro “for”