JAVA  
 SE  
  
Un gran compañeros en tu camino querido por muchos odias por otros estas preparado para esto?.  
  
  
Me dirigo a ustedes sin importar donde este, como seas cual seas tu religion pero si estas aquí sin que estes siendo obligado compartimos un deseo de aprendizaje un deseo de amor propio o simplemente curiosidad desde aquí te digo hoy soy un joven con una laptop escribiendo al futuro pero te aseguro que estas en el punto de no retorno aprenderas tanto que cuando termines no queras que sea el final, en caso que estes aquí y este siendo obligado parate un momento y recuerdo esto o preguntate esto te apasiona? Si no es asi estas a tiempo de bajarte mientras mas te tardes lo hagas el precio a pagar sera mucho mas caro ya estaras mas lejos por lo tanto te recomiendo bajarte en esta parada muchas gracias.  
Y para quienes si se quedaron sean “BIENVENIDOS A LA INTRODUCCION JAVA ” te aseguro que este viaje sera unico enteras comprenderas y podras explicar que es esta es una de las mejores de saber si dominas un tema, no te mentire este camino no es para cualquiera es posible que te sientas frustado,triste o en soledad pero recuerda si estas seguro seras capaz de poder pagar este precio sin mas que decir empezemos desde las entrañas del queridisimo JAVA recalco sean bienvenido sin importar tu idioma, raza, religion, o edad lest’go.  
  
  
  
Donde nace JAVA?Como se llama su creador?  
  
Este nace de la mano de James gosling mientras trabaja en Microssystems el era un ingeniero de software de los mas nerd este lenguaje hace su debut en la decada de los 90 exactamente en el año 1995 pero el proyecto comenzo en 1991 en principio su nombre seria “oak” pero al darse cuenta que este ya estaba registrado fue por el cambiado por el que todos hoy conocemos, este nacio en base a la necesidad de crear un lenguaje que sea multilplaforma es decir que pueda ser usado desde un latop o un telefono es aquí donde entra el “JVM” pero mas adelante profundizaremos con este.  
  
Microssystems fue comprando por oracle en 2010 esto trabajo muchas expectativas ya que estas empresaba podria invertir mas en tecnlogias,versiones y soporte para muchos años mas un pequeño dato es que el creador de java trabajo para esta empresa durante algunos años pero decidio retirarse debido a que no le gusto hacia donde estaba llendo este.  
  
Cuando escribo esto es 2024 el ingeniero James Gosling tiene actualmente 69 años de edad no se a retirado del software pero sus aportes a java son casi nulos o muy pocos desde que retiro en 2010 se sabe que a participado en proyectos como News, empezo a trabajar para google el 28 de marzo del 2011 según lo que encontre en el red actualmente trabaja Arquicteto de software para liquid robotic.  
  
  
  
  
  
  
  
  
  
  
  
  
 VERSIONES DE JAVA Y JDK  
  
Tenemos dos versiones java se que es la Estándar edition uso en al web que pronto aprenderan y les explicare en este informe y su segunda version que es JAVA EE enterprise edition esta es utilizada para creacionde app empresariales muy grandes.  
  
  
Pero antes de sumerginos en JAVA SE debemos entender que java tiene dos formas de sacar actualizaciones primero regresemos un poco al pasado cuando java actualizaba o sacaba una nueva version cada 6 meses esto era un dolor de cabeza para quienes trabajan con proyectos grandes ya que tenian que estar migrando a la mas reciente con sus nuevas funciones, tecnologias en muchos casos habia problemas de compatibildad desde de esto oracle tomo la decision de crear versiones lts que tendrian soporte para 3 años. Y versiones que no estas seguirian con su soporte de solo 6 meses pero con menos novedades esto fue un respiro para la comunidad en lo que respeta a desarrollo se acuerdan que digimos que java nacio por la necesidad de multiplaformas es aquí cuando entra el JDK este es el que hace especial a java pero porque se preguntara lo vemos enseguida.  
  
Ques es el JDK o como su nombre en english lo dice el “JAVA VIRTUAL MACHINE”este es la maquina virtual de java que es capaz de hacerlo multiplataforma e interpetarlo este esta compuesto por tres partes, JAVA RUTIME ENVIRONMENT JRE -> COMPILADOR DE JAVA - >API DE DESARROLLO tenemos su maquina virtual que como ya lo mencione es lo que lo hace ser java, contamos con su compilar que es el encargado de traducir nuestras lineas de codigo a lenguaje que el computador entiende por utlimo tenemos el api este es un metodo class que nos ayudara. Crear nuestro programas esto a seguido evolucionando hasta yo la ultima version registrada de java es la 22 pero entre las mas usadas tenemos a las version 17 lts y las java 8 lts pero pongan atencion desde la version 10 de java se comenzo a cobrar licencias para poder usarlas en produccion tiene un modicos precio de 2.5 mesualmente se puede usar su jdk en version free pero solo testeo y desarrollo no para produccion.   
  
Para quienes ya tenemos experiencia en programacion o sabemos algo se que al leer API en el JDK pensaron si el api es como un puente que usa para conectar cosas mas adelante profundizaremos mas sobre esto.  
Al que acotar es que contamos con un version open source del JDK hecha por la comunidad donde todos podemos aportar justo ahora que escribo esto le tengo instalada si pueden y saben vayan instalando la 8lts y 17lts.  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
HERRAMIENTAS MAS USADAS EN JAVA  
  
  
No te voy a recomendar una ide para desarrollo ya que soy fiel creyente que nos importa la maquina sino el conductor por lo tanto dejare que elijas no escogas el primero ve en que se basan que te ofrecen que tan pesado puede ser que tants limitaciones puedas tener independientemente desde la maquina en que me leas por lo tanto eso queda a elecion tuya pero ojo ten encuenta que necesitaras extenciones para poder aprender de la mejor forma.  
  
Te puede decir que entre los ide mas usadas tenemos a esclipse si gusta le podes ir dando un vistaso, visual code studio , blueJ y Netbeans estos son los que mas destacan queda en usted elegir o buscar otro.  
  
  
Una vez instalado tu ide sin importar cual sea busque en internet como crear el archivo y lo guarda como deseo pero al final nombre.java listo si usted no tiene ningun conocimiento sobre git y github le pido que pare y busque en este mismo repositorio el documento de estos dos que le daran un plus en estos informes ya que estaremos usando.Una vez creado ponemos nuestro inico   
  
  
Public class nombre{  
 public static void main(String[] args){  
  
System.out.println(“hello world, how are you today yes you”);  
  
  
  
  
}  
}

Habras hecho tu primer hola mundo si tenes alguno error solucionalo todo esta en internet tengo paciencia y vamos tu puedes por que te dejo solo y no te ayudo por que esto te formara esta industria es a diario que hay errores y tienes que saber como resolverlos y sobrellevarlos .  
  
 ETAPAS DE LA PROGRAMACION

Bueno las etapas de la programacion es todo lo que ocurre por detrás que nosotros no vemos son 4 etapas que debemos conocer.  
  
.- Primera etapa   
en esta etapa tenemos nuestro codigo .java en este se encasula nuestras lineas de codigo  
  
.- Segunda etapa  
en este se compila e interpreta el codigo en caso que tenga un error de sintaxis sera mostrado.  
  
Que es un codigo interprestado recuerdan a JVM este es aquel que se encargada de esto interpretarlo para poder comprelo de una mejor forma veamolo como un vigilante que revisa linea por linea de en busca de errores pero entonces quien se encarga de compilar bytecode es aquel que se encarga de esto este lo encontraremos en todo los lenguajes de programacion y en este caso trabaja de la mano con el java virtual machine en esta etapa nuestro archivo es .java.c, el code es traducido a lenguaje que el   
  
.- Tercera etapa  
en esta se recibe el codigo ya traducido listo para ser ejecutado y interpretado  
  
.-Cuarta etapa  
esta es la salida que ve el usuario ya sea un programa un resultado etc.  
  
  
  
  
  
  
  
  
  
 JSHELL  
  
  
Esta shell nos ayudara depurar codigo, testear pequeñas partes , explicar conceptos de formas rapidas pero tengamos en cuenta que no podemos programar en esta shell pero que son estas son tecnologias que funcionan con nuestra terminal donde podemos descargar muchas informacion en pocos minutos, hay de muchos tipos con distintas sintaxis y funcionalidades para todo tipo de usuario por lo tanto tenemos uno para java recuerda usarlo cuando sea necesario.  
  
  
  
VARIABLES EN JAVA  
  
bueno que hace a una variable esta deben tener dos puntos importantes para que puedan ser considerados una variable deben tener un nombre y almacenar algo ya sea un dato primitivo o una cadena string

-🡪 String valorUno = “valor”;  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
CONVENCIOn DE NOMBRES   
  
Que son las convenciones estas entran en buenas practicas que seran un regalo tanto para ti del futuro para comprender y entender tus codigos o para otros programadores que traten de hacer lo mismo tenemos varias convencios algo que debemos saber es que java es sensible a mayusculas y minusculas por lo tanto tengan mucho cuidado algo declarar variables o metodos, estas convenciones sone estandares ya establecidos por la comunidad.  
  
.- No podemos comenzar el nombre de nuestra variable con un numero podemos empezarla con “$” o “\_”  
  
.- las contantes son inmutables estas son escritas en mayusculas en caso que tengan dos palabras o mas son separadas por un “\_” y deben ser escritas de esta forma para que sean inmutables:  
 🡪 final VARIABLE\_UNO = 12;   
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
 CAMEL CASE  
  
  
Estas es una de las nomeclaturas mas usadas por las comunidad traducida al español dice camello pero aunque no lo crean este nombre tiene mucho que ver, cuando profundizamos nos damos cuenta que tenemos dos tipos upper y lower.  
  
  
.- UPPER CAMEL  
Esta convecion no dice que toma clase debe empezar la primera letra en mayuscula si son mas de uno debe separase con mayusculas:  
  
🡪 public class CafecteriaDeLuz{  
  
.- LOWER CAMEL  
  
En cuanto a variables o metodos debe contener dos palabras que describan lo que requiere hacer la primera palabra debe empezar en minuscula y la primera letra de la segunda en mayuscula:  
  
🡪 int numberUno = 12;

TIPOS DE DATOS   
  
  
  
HEY QUE PASADO YO te informo que nos he equivocamos eliminadnos el work donde llevamabamos nuestras anotaciones por tanto aqui andamos   
  
  
  
bueno en los distintos tipos de datos numéricos tenemos datos primitivos que encontraremos en todos los lenguaje de programacion en este momento nos enfocaremos en los datos numéricos debemos tener un compresión solida para poder usar el correcto ya que como sabemos las variables ocupan despacio en la ram bueno sin mas que apórtate   
  
  
tenemos byte, short,long,int,float y double   
  
el mas comun y mas usado sera el int pero es muy seguro de muchos de mis lectores ya lo conocen y usaron en sus proyectos pero no conocemos cuantos bytes usan de espacio en la ram  
  
primero los byte son los mas pequeños y van desde los -128 a 127 y ocupan 1 byte algo que casi se me olvida menciona el que alcance de estos datos van desde numéricos a positivos   
  
los short le siguen estos cuentan con un rango de -32,768.a 32,767 y ocupan 2 bytes  
  
  
los int un viejo conocido este tiene un rango de 2,147,483.467 algo igual que negativo este ocupa cuatro bytes  
  
uno de los grandes es long que ocupa 8 bytes recuerda poner el L al final para decir que es un long

en lo que respeta lo decimales estos entienden valga la redundancia con numeros decimales   
  
  
float es pequeño a comparación de double pero en caso que usted necesite ser exacto no cerca use double  
  
  
float ocupa 4 bytes no olvides poner la.f al final para declararlo con float  
  
  
y duble 8 bytes

TIPO DE DATO CHAR Y BOOLEAN  
  
Primero empecemos con char que es un este es un metodo que solo trabaja con characteres este fragmenta palabras y las entrega como caracteres aunque como metodo curioso recordemos la tabla aski de unicon donde toda letra tiene un valor numerico como este tiene un valor numerico podemos sumar este char con un int o cualquier dato primitivo.  
  
 char characterPrueba = ‘c’;  
int numberUno = 11;  
  
numberUno = numberUno + characterPrueba;  
  
System.out.println(“hey funciona”+numberUno);  
  
  
  
En lo que respeta al boolean como sabemos las computadoras tambien toman decisiones que son 0 ,1 o true o false este operador de decisiones no sera muy util cuando necesitemos tomar decisiones o saber si algo es verdadero o falso:  
  
  
boolean verdaderoFalso = 8 < 5  
  
System.out.println(verdaderosFalso);  
  
  
  
  
  
  
  
OPERADORES DE ASIGNACION INCREMENTO Y DECREMENTO Y OPERACIONES MATEMATICAS  
  
  
Bueno los operadores de asigancion tanto de incremento como decremento son ayudaran muchisimo de ayuda en operaciones artimeticas para poder ser mas rapidos todos concemos “>” , “<” ,”=” con el igual tenemos algo especial cunado usamos dos quiere decir que estamos comparando “==”, “-” , “+”, la multiplicacion “\*” y la division “/”.  
   
Usos y ejemplos Operaciones asignacion:  
  
a+= b a = a + b;  
 int a = 12;   
 int b = 13;  
   
a+=b  
  
System.out.println(a);  
a-= b a= a – b;  
 int a = 12;   
 int b = 13;  
  
a-=b;  
System.out.println(a);  
  
  
a\*=b a = a \* b;  
a/=b a = a / b;  
a%=b. a = a % b;  
  
.- Este ultimo nos dara el residuo de la division.  
  
  
  
   
Operaciones de incremento y decremento las encontraremos en muchas operaciones y ciclos pero estas tiene dos formas:  
  
  
Preincremento:  
++i, --i  
este da un valor de resultado es decir si ingresamos 5 como nos aumenta 1 es decir retorna un nuevo valor pero a que se debe esto primero aguanta el uno y despues recibe el valor de i por lo tanto al llarmalo tenemos un nuevo valor.  
  
  
Postincremento:  
i++ , i--  
en este caso no nos regresa un nuevo valor este nos retorna el mismo valor de “i” pero a que se debe esto, aquellos es por la forma de declarar ya que primero ponemos “i” y despues el incremento por lo tanto este primero coje el valor y despues le agrega el 1 pero este no se refleja directamente este solo nos refleja el valor de i el nuevo valor se guarda en la ram.  
  
  
  
En lo que nos respeta a operaciones matematicas tenemos las operaciones basicas con sus respetivas sintaxis en java.  
  
  
Suma:  
  
int a = 12;   
 int b = 13;  
  
a+=b;  
System.out.println(a);  
  
  
Resta  
  
a-=b  
System.out.println(a);  
  
  
Multiplicacion

A\*=b  
  
System.out.println(a);  
  
  
  
Division  
  
  
a/=b;  
  
System.out.println(a);  
  
  
  
  
  
  
  
  
  
  
  
  
Operaciones con Math

Estas es un clase estatica muy invaluable por todo nos que nos ofrece para poder realizar operaciones matematicas con una amplia gama de metodos  que nos permitiran trabajar de forma precisa y rapida.

Que es una clase estatica?

Es bueno como ya sabemos este cuenta con muchos metodos lo que hace sea estatica es que no necesitas crear  un clase para poder llamarle simplemente usas math y uno de sus metodo esta tambien la hace inmutable para metodos que traen por defecto un valor.

Trigonometria

pero que es no esta demas dar un repaso

\* sin(double a) = Calcula el seno de un angulo en radianes

\*cos(Double a) = calcula el coseno de un angulo en radianes

\* tan(Double a)= calcula el tangente de un angulo en radianes

\*asin (double a)= calcula  el arcoseni de un valor devolviendi n angulo en radianes \*acos(Double a)= calcula el arcocoseno de un valor devolviendo un angulo en radianes

\*atan(Double a)= calcula el arcotangente de un valot devolviendo un angulo radianes

\* exp(double a) = calcula. El valot de e elvado a la potencia a.

\* log(double a) = calcula el logaritmo natural base (base e) de a.

\*Raices .- sqrt(double a) = nos sirve para obtener la raiz cuadrada de un numero sintaxis Math.sqrt(14);

\* Redondeo

.- ceil(double a) = este redondea a hacia arriba al entero mas cercano

.-floor(double a)= este redondea hacia abajo al entero mas cercano

.-round(double a)= redonde al entero mas cercano

\*Maximos y Minimos

.- max(int a a b int)= este nos entregara el numero mayor de lo dos enteros que has comparado

.- min(a b)= devuelve el valor menor de los dos numeros a comparar

\*Potencia

.-pow(a b)= este nos ayudara a sacar un numero elevado a una potencia

\*Numeros aletorios

random()= nos devolvera a nuemros random.

CAST EN VARIABLES  
  
Que son los cast en las variables se que mucho mas adelante necesitaremos ser muy exactos o simplemente un estimado de un recuerdo.   
Pero que son estos dos terminos que tocamos primero exactos cuando necesitamos ser muy pero muy exactos con un resultado o un objecto por lo tanto necesitaremos usar aquello por otro lado tenemos un estimado un resultado que se acerque al exacto por lo tanto tengamos en cuenta estos dos conceptos para poder tener resultados que nos ayuden en nuestros programas pero como aplicamos estos conceptos a codigo.  
  
  
EXACTOS:  
  
int a = 9 ;  
int b = 100;  
  
  
  
System.out.println((double)a/b)  
  
esta es un de las tantas formas de ser exactos.  
  
  
   
ESTIMADO:  
  
int a = 49;  
int b = 42;  
  
System.out.println(a/b);  
  
este sera un resultado estimado dejando de lado muchos numeros pero como tal se dice .  
  
  
  
  
CASTEO ENTRE TIPO DE DATOS  
  
Como ya conocemos tenemos distintos tipos de datos pero la conversion y el casteo entre esto puede ser de forma automatica pero no con todo ya que un int si puede entrar en long pero un long no puede entrar en int por que es mucho mas pequeño veamos a continuacion con el casteo entre datos y sus curiosidades

BYTE  
1BYTE

Listo como nos daremos cuenta tenemos lineas de colores en nuestro grafico las lineas azules nos dan a entender que la conversion sera de forma automatica y de daras cuentas de que esto se da solo para datos que entran en los mas grande y las lineas a naranjadas nos dan entender que para este necesitamos usar el cast, un pequeño dato es que los char pueden ser convertidos de forma automatica ya que este en el fondo son numeros como son represtandos por la tabla asci e unicode, algo que le puede agregar es que un char vale mas que un short a pesar de valor lo mismo 2 bytes esto se debe a que java actualizo a char para que pueda con la unicode y asci y asi paso de 16 bits.  
  
  
  
  
ARCHIVOS .JAR  
  
Estos son archivos que nos ayudaran para poder llevar nuestros archivos .java a otro ordenadores para compilarlos y ejecutarlos esta es una de las mejores para compartir nuestro proyectos de forma rapida.  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
SENTENCIAS IF ELSE Y ALCANCE DE VARIABLES

CHAR  
2 bytes

DOUBLE  
8 BYTES

FLOAT  
4 BYTES

LONG  
8 BYTES

INT  
4 BYTES

SHORT  
2 BYTE

La sentencia if else es para tomas de decisiones las usaremos mucho mas adelante dentro de esta en calsularemos un codigo que sera respuesta de la condicion que le apliquemos a nuestro codigo if que este al español no dice si y la parte else no dice sino bueno vemos su sintaxis.  
  
  
  
if(condicion del codigo){  
 ///codigo que se ejecuta si la condicion se cumple  
}else{  
// codigo que se ejecuta si la condicion se cumple  
}  
  
esta es las sentencia if-else recuerda que podemos agregarle mas cosas  
  
  
if(¡(codigo a ejecurtarse)){  
codigo que se ejecuta si la condicion se cumple  
}else{

codigo que se ejecuta si la condicion se cumple  
}  
  
  
  
el alcance de las variable dentro de estos ciclos no es la mismo a los que estan fueras de estos por lo tanto tengan mucho atencion donde las declara y piensa ejecutar.  
  
  
  
  
  
OPERADORES LOGICOS Y EXPRESIONES BOOLEANAS  
  
  
Operadores logicos para es lo mismo que la tabla de la verdad por lo tanto se lo enseñare de la misma como yo lo aprendi hace ya año:  
  
  
  
Tenemos nuestro primero operador la negacion (not) que se representa con “!” este nos dice que si enviamos un elementos en verdadero este nos dira que es falso es lo contrario si le envias en falso es verdadero.  
   
! 🡪 si entra verdadero entrega falso es facil y sencillo se que cuesta en entenderle durante un poco pero lo podras  
  
Nuestro segundo operador es el operador AND representado por “&&” que nos indica evaluando de izquierda a derecha si el primero es verdadero continua sino para de golpe este necesita que sus condiciones sean veraderas para poder tener como resultado verdadero  
  
&& 🡪   
 literal 1. Literal 2. Resultado  
 v f f  
 v v v  
  
  
por ultimo tenemos al operador “OR” este es represntado con “||” este nos dice que al menos uno de los literal debe ser verdadero para que el resultado sea verdad  
  
|| 🡪   
literal 1. Literal 2. Resultado  
 v f v  
 v v v  
   
expresiones booleanas esta son auqellas que no ayudara en tomas decisiones respondiendo si algo es verdadero o falso o true o false.  
Las podemos complementar con operadores de comparacion como   
 ¡= que no dira si los dos numeros son iguales, == comparada los dos valores, > o < , <= menor igual que y => mayor igual que .

* STWICH  
    
  Esta forma parte de las estructuras de control donde podremos comparar la condicion y este escogera el case que sea el mismo debemos usar este cuando tenemos mas de 3 condiciones veamos su sintaxis.  
    
    
    
  stwich(condicion a evaluar){  
    
  case ‘condicion’:  
    
  System.out.println(“…..”);  
   break;  
    
  case ‘condicion’:  
    
  System.out.println(“…..”);  
   break;  
    
    
  case ‘condicion’:  
    
  System.out.println(“…..”);  
   break;  
    
    
    
    
    
    
  }  
    
    
  en esta sentecia de bloque solo se adminten datos como. Byte, Short, Int, long o string esto cambio en 17 pero pronto veremos mas de ella, en pocas palabras este busca el case con coincida con la condicion y ejecuta el codigo dentro de este.  
  FUNCIONES Y PARA QUE SIRVEN  
    
    
  Estos son usados cuando estamos repitiendo mucho codigo y para poder delegar tareas a otras partes del codigo esto va de la mano con la moduralidad y todos sus beneficios a largo plazo para un codigo o proyecto escalable pero como hacemos uso de las funciones ya que sabemos para que sirven.  
    
  Primero debemos arguntamentar nuestras variables para despues poder llamarlas pero como hacemos esto:  
    
    
    
  \*Digamos que tenemos tres variables que entran son int , double, float debemos poner una descripcion de lo que queremos hacer en nuestro argumento.  
  int valorUno = 12;  
  double valoTwo = 23;  
     
  SumaTotalDeLaEntradaDelUsuario =(valorUno, valoTwo);  
  //aquí los estamos argumentando comenzamos con la funcion  
  // cabe recalcar que no importa donde la pongamos si es al comienzo dentro de la class o futura del metodo main pero no puede ir nunca dentro de este   
    
  System.out.println(SumaTotalDeLaEntradaDelUsuario)  
    
    
    
    
    
  //la palabra static por el momento ira dentro de este metodo tanto podemos usar un returno o simplemente hacer los System.out.println(“dentro”); ya veremos como en este caso usaremos un return   
  //recuerda decile que tipo sera int o double  
  public static int SumaTotalDeLaEntradaDelUsuario(int valorUno, Double valoTwo);  
    
  valorUno = valorUno / valoTwo;  
    
  return “su valor es” +valorUno;  
  este regresara al ultimo System.out.println y asi habras hecho tu primera funcion ☺

JAVA DOC  
Este es importate cuando queremos escribir nuestra propia documentacion esto es un acto tanto para tu yo del futuro pueda comprender lo que hace x variable o para otros programadores del futuro por lo tanto vamos a conocer como crear esto.  
  
  
🡪 todo hasta este punto sabemos como realizar comentarios en nuestro codigo // comentario uno pero esto aquí viene al caso es esta es una de las primeras formas que puedes usar pero teneme mas.  
  
🡪 en segundo lugar tenemos este tipo //\*  
\*hey que tal   
\*  
  
  
  
\*//

🡪 por ultimo tenemos a esta que es con cual podremos guardar y generar nuestra documentacion en html y podemos usar este lenguaje de etiquetas para darle.Una mejor presentaacion

/\*\*  
\*@author: este nos ayudara a declarar quien es el autor (solo para clases)  
\*@version: version de la clase (solo para clases)  
\*@see referencia a otra clase  
\*@param: description de parametro  
\*@return descripcion de lo que devuelve un metodo(solo si no es void (recuerda funciones))  
\*@throws descripcion de las expeciones que pueda dar un metodo  
\*@deprecated marca un metodo como obsoleto  
\*@since indica la version desde que exite el metodo  
\*/  
recuerda que podemos usar html dentro del javadoc de esta forma:  
  
  
/\*\*  
\*<tittle>metodos y sus funciones</tittle>  
\* <p> tenes nuestro @param nombredelavariable</p>  
\*@expection se indica que metodo es vulnerable a tirar expeciones  
\*@inheritdoc indica la herencia o implementacion procedentora  
\*/  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
BUCLE WHILE AND DO-WHILE  
  
While este es un ciclo que muchos llegan a confundir con el for pero a diferencia de este busca algo en especifico y solo se ejecuta si la condicion es verdadera , en caso que la condicion se cumpla o sea falsa este saldra del bucle.  
  
Do while este a diferencia de su hermano menor primero ejecuta y despues verifica la condicion si la condicion es falsa o en muchos se cumple este saldra del codigo si no es asi este se seguira repitiendo hasta que se cumpla o sea falsa esto depende de tu codigo y como este enfocado.

*public* *class* NumberGuess{

*public* *static* *void* main(*String*[] *args*){

*// Genera un número aleatorio entre 1 y 100. El usuario debe intentar adivinarlo. El programa le dirá si su*

*// número es mayor o menor que el número secreto. Continuará hasta que el usuario acierte.*

*// System.out.println(numberUsuario);*

*Random* numberUsuario = new Random();

*Scanner* usuarioListen = new Scanner(System.in);

*int* randomUsuario;

*int* soloNumeros;

do {

System.out.println("escriba un numero quiere ganarle a la maquina :0");

soloNumeros = usuarioListen.nextInt();

*// int intValue = Integer.parseInt(soloNumeros);*

randomUsuario = numberUsuario.nextInt(100);

*String* process = (soloNumeros>randomUsuario) ? "gano usuario" : "gano maquina";

System.out.println(process+" numeros mayor");

} while (soloNumeros == randomUsuario);

System.out.println("ganaste tu numero acetto ya que el numero de la maquina fue " +randomUsuario);

en lo que respeta a while mi pequeño viajero aquí tu ejemplo ponele atencion por favor:  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
Operador ternario   
  
  
Este nos ayudara para poder simplificar el uso de if-else pero ojo este nos es un remplazo de esta estructura de control ya que el operador ternario es simple pero no tan estructura y libre este trabaja con booleanos. Y operaciones artimeticas simples como hacemos esto de la siguiente su sintaxis es:  
  
  
  
  
boolean operacionInter = (a < b)? true: false;  
  
String frasesRetorno = (maria == userHabla)? “no coinciden” : “very good”;  
  
  
  
cabe reclacar que debemos evitar el uso de ternario para logica compleja, legibilidad y asignaciones o llamadas de funciones  
  
  
Mas ejemplos para poder comprender:  
  
  
boolean preguntaOne =(comprobandoUser.length >= 2)? True:False;  
  
  
  
int regresandoNumber = (1+2) ? 2 : 3;

System.out.println(regresandoNumber); // Imprimirá 2  
  
  
  
  
  
  
  
Bucle for   
  
  
  
Este es una ciclo que nos dejara recorrer listas, diccionarios e arrays tambien podremos ver o modificar cuantas veces iterara sobre si mismo o sobre alguna lista lo podemo usar con mucho fines el mas comun es recorrer cadenas sin importar cuales sean su sintasis es simple en este lenguaje:  
  
  
for(int i =0; i < variable; i++){  
  
}  
 Lo podremos usar tanto de forma negativa y positivamente para recorrer cadenas de atrás para adelante o vicerversa este es fundamental en nuestro camino nos permitira procesar gran cantidad de datos en segundos automatizar tareas entre otras cositas dejo una pista anidados.  
  
  
  
  
  
  
  
  
  
  
  
  
  
Este ultimo ejemplo es conocido como for mejorado o for-each por que este nombre este es dado porque itera sobre un elemento sin un indice dado o especificado.Este toma el valor de num y saca indices y valores. Es decir no tendras que preocuparte por el limite del array que recorras o la lista, ayuda a tener una lectura del codigo mas simple y deja menos brecha abierta para errores, no pomdemos iterar en negativo o modificar elementos  
  
  
 Break , continue y return  
  
  
Primero hablare en general todos estos son usados dentro de ciclos y cuales son los ciclos son aquellos que iteran sobre si mismo ya sea buscando algo en especifico, recorriendo un almacen (array) o con mucha informacion de por medio por lo tanto recuerda estas palabras claves son usadas en ciclos.  
  
  
  
Break es un usado para darle un alto a un ciclo cuando se cumple x condicion pero tenemos que tener en cuenta que este caso tambien le usamos en los stiwch con el mismo proposito pero este no es un ciclo sino una estructura de control.  
  
  
Continue este es espetacular ya que al cumplirse x condicones este si lo usamos no parara el flujo de trabajo sino que lo regresa al incio para volver a empezar  
  
  
Por ultimo el viejo conocido return no nos confundamos con su nombre mi buen amigo este aquí no regresara el flujo de trabaja al inicio con la anterior palabra clave este parara este flujo pero retornada algo de retorno si es aque la condicion lo tiene en las funciones esta presente mas adelante le veremos.  
  
  
  
  
  
 ARRAYS  
  
Veamos a este como un almacen donde podemos agregar datos de todo tipo un lugar con datos int otros con char otros byte etc.  
Debemos recordar que estos y sus indices comienzan de 0 hasta el valor que usted ponga con esto podemos agregar valores eliminar remplazar y eliminar.  
  
caracteristicas generalmente estos solo pueden llevar un tipo de dato por arreglo su limite se establece a lo que son creados.  
  
Sintaxis:

for(*int* iv = 0; iv < numberT12; iv ++){

NumberWoo.add(iv);

}

for(*int* ii = 0; ii < procesoLin.size(); ii++){

NumberWoo.add(procesoLin.get(ii));

}

for(*int* num : numeros){

System.out.println("tu edad es "+intValue+" los años que has tenido son "+numeros);

break;

}

}

*ArrayList*<*String*> palabras = new *ArrayList*<>();

palabras.add("lenguaje");*//00*

palabras.add("matematicas");*//11*

palabras.add("fisica");

palabras.add("historia");

tenemos muchos tipo este es un array list a este no le decimos cual es el limite que puede recibir de datos este es infinito.

*String*[] grupoA = {"a","b","c","d","e","f","g","h","i","j","k","l","m"};

*String*[] grupob = {"ñ","n","o","p","q","r","s","t","u","v","w","x","y","z"};

*String* f = "feminino";

*String* m = "masculino";

Contamos con array de distintas dimensiones de una , de dos , tres o hasta incluso cuatro dimensiones. Que podremos recorrer con ciclos pero por el momento aprenderemos a como hacerlas, agregar a estos arreglos elmentos y sus posiciones:  
  
  
Bueno tenemos que tener en cuenta las dimensiones de nuestro arrays pero antes tenemos saber cuales seran sus dimensiones cuantas filas y cuantas colummas tenes un listados de nombre de estudiantes con sus respetivas edades.  
  
lucas 15 años  
Maria 14 años  
Mariano 16 años  
Luis 15 años  
Matea 17 años  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
es el mismo proceso para cualquier otro ejercicio o para mas dimensiones mi querido viajero.  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
 Ciclo for anidados  
  
Que son y para que nos sirven estos nos ayudaran a recorrer arrays en varias dimensiones tanto de filas y colummas para poder obtener sus valores veamos ejemplos:  
  
recalco usted no aprendera viendo ejemplo usted tiene que practicar:

*public* *class* ArrayDimensiones{

*public* *static* *void* main(*String*[] *args*){

*String* [][] edadesStudent = new *String*[5][2];

edadesStudent [0][0]= "lucas"; edadesStudent [0][1] = "15 años";

edadesStudent[1][0] = "Maria"; edadesStudent [1][1] = "14 años";

edadesStudent[2][0] = "Mariano "; edadesStudent[2][1] = "15 años";

edadesStudent [3][0] = "Luis"; edadesStudent[3][1] = "17 años";

System.out.println(edadesStudent[0][0]+" "+edadesStudent[0][1]);

*public* *class* MatrizIntermedio{

*public* *static* *void* main(*String*[] *args*){

*// Diagonal principal: Imprime los elementos de la diagonal principal de una matriz.*

*int* matrizTres[] [] = new *int*[3][3];

*// {{1,2,3}}*

*// {{1,2,3}}*

*// {{2,3,2}}*

System.out.println(matrizTres[0][0]);

System.out.println(matrizTres[1][1]);

System.out.println(matrizTres[2][2]);

}

}

*public* *class* MatrizIntermedio{

*public* *static* *void* main(*String*[] *args*){

*// Diagonal principal: Imprime los elementos de la diagonal principal de una matriz.*

*// Matriz traspuesta: Calcula la matriz traspuesta de una matriz dada.*

*int* matrizTres[] [] = new *int*[3][3];

matrizTres[0][0] = 1; matrizTres[0][1] = -2; matrizTres[0][2] = 5;

matrizTres[1][0] = 10; matrizTres[1][1] = 11; matrizTres[1][2] = 133;

matrizTres[2][0] = 0; matrizTres[2][1] = -7; matrizTres[2][2] = 67;

System.out.println(matrizTres[0][0]+" "+matrizTres[1][0]+" "+matrizTres[2][0]);

System.out.println(matrizTres[0][1]+" "+matrizTres[1][1]+" "+matrizTres[2][1]);

System.out.println(matrizTres[0][2]+" "+matrizTres[1][2]+" "+matrizTres[2][2]);

}

}

Como ultimo para terminar esta unidad. Te recomiendo practicar y luchar creeme solo practica hace al maestro en todo ambito por lo tanto te recomiendo aquello mientras escibro esto sigue practicando por tal hagalo mi viajero este no es el final es un nuevo comienzo te dejare unos pequeños datos para seguir y un ejemplo de un matriz.