Java

Java es un Lenguaje de programación orientado a objetos diseñado para se portable en diversas plataformas y sistemas operativos. Desarrollado por Sun Microsystems, se diseñó con base en el lenguaje de programación C++, e incluye características especiales que lo hacen ideal para crear programas en internet.

En principio, Java permite incluir gráficas interactivas y otros efectos especiales en la páginas del World Wide Web. Como cualquier otro lenguaje de programación permite escribir programas. Existen programas de Java especiales llamado Applets, que se ejecutan dentro de una Página Web con capacidades idénticas a las de cualquier programa tradicional.. Además, de ejecutar un applet, el servidor remoto lo transmite a su navegador a través de Internet.

El JDK

El Kit de Desarrollo de Java (JDK por sus siglas en Inglés) es un conjunto de programas que proporciona Sun y que contiene todo lo necesario para crear aplicaciones y applets de Java. Específicamente, el JDK contiene un compilador de Java, un depurador y appletviewer con el que puede ejecutar Applets fuera de cualquier Navegador. Además contiene documentación y applets de ejemplo.

Se puede bajar el JDK de Internet, puede obtener una copia del JDK para su computadora (el JDK específico para Windows, Mac o Unís) utilizando su navegador. Conéctese al sitio web de Sun:

Http:// java.sun.com/java.sun.com/products/JDK

De allí siga el enlace Downloads. Ahí encontrará varios enlaces que apuntan a páginas para bajar varias versiones del JDK.

TIPOS DE DATOS

1) Tipos de datos primitivos en Java

Un tipo define un conjunto de valores que puede tomar un dato, así como el conjunto de operaciones que se pueden realizar sobre el dato. Java soporta ocho tipos primitivos.

Tipos Enteros

Los tipos enteros se utilizan para números sin parte fraccionaria. Cada uno de los tipos enteros puede almacenar un rango diferente de valores.

Tipo	Tamaño	Rango
long	8 bytes	-9,223,372,036,854,775,808L a 9,223,372,036,854,775,807L a
int	4 bytes	-2,147,483,648 a 2,147,483,647
short	2 bytes	-32,768 a 32,767
byte	1 byte	-128 a 127

Tipos Reales (punto flotante)

Los dos tipos de punto flotante almacenan números con parte fraccionaria

Tipo	Tamaño	Rango
double	8 bytes	+/-1.79769313486231570 E +308 (15 digitos significativos)
float	4 bytes	+/- 3.40282347 E +38 (7 digitos significativos)

El tipo boolean

El tipo bolean sirve para hacer pruebas Lógicas.

Tipo	Tamaño	Rango
boolean	1 bit	true (verdadero) o false (falso)

El tipo char

El tipo char almacena caracteres alfanuméricos y unicode

Tipo	Tamaño	Rango
char	2 bytes	65536 caracteres posibles

El Unicode en Java

El juego de caracteres Unicode de dos bytes soporta los diferentes juegos de caracteres que componen el texto escrito de los diferentes idiomas. En Java, el tipo char de dos bytes soporta caracteres Unicode.

2) Tipo de datos Clase

Ademas de los 8 tipos de datos primitivos, una variable puede tener una clase como su tipo, como por ejemplo las cadenas caracteres son objetos de la clase String, también para poder utilizar colores usamos tipos de la clase Color.
Existen también las Clases Integer, Float, Double, Long , Boolean, Char, con métodos que permiten efectuar una serie acciones con estos tipos de datos.
En Java siempre se van a trabajar con clases y se tiene que crear clases para poder trabajar con ellas pues es un lenguaje orientado a Objetos.

Cadenas de Caracteres

En java una cadena es un objeto de la clase String, en vez de ser un tipo de dato primitivo, y las cadenas no se almacenan en arreglos, lo que se hace como lenguajes como C.
Por ejemplo "Java", "Programación"

IDENTIFICADORES

Un identificador es un nombre que se le da a algo en Java (Variable, Clase,método).
En Java los nombres no tienen límite de longitud, y las mayúsculas se consideran diferentes de las minúsculas. El nombre debe iniciar con una letra, después puede tener letras y/o dígitos. Una letra se define como cualquiera de los caracteres de la 'A' a la 'Z', de la 'a' a la 'z', los caracteres '_' y '$', y cualquier carácter unicode que se considere una letra en algún idioma. Un dígito se define como '0' a '9' y cualquier carácter unicode que se considere como dígito. No se pueden utilizar símbolos como '@' ni '+' ni espacios en blanco dentro de los nombres.

VARIABLES

Son simplemente nombres que el programa asocia a localidades especificas de la memoria. Como indica la palabra variable, el valor almacenado en estas localidades puede cambiar conforme avanza la ejecución del programa.
Cada variable tiene un tipo específico, el cual indica a la computadora cuanta memoria necesitan los datos y las operaciones que pueden realizar con ellos.

Declaración de variables

Tipo de dato identificador(es)
Ejemplo:

```
double  precio;
float  base,altura;
int edad;
char carácter;
String nombre; // Declaramos la variable nombre tipo cadena
boolean sw; // Declaramos una variable tipo boolean
```

Tambien se pueden inicializar las variables al declararlas:

```
int c=10;
float presion=12.90;
boolean  encontrado=false;
String apellido = "Carranza";
```

LITERALES

Los literales presentan valores específicos dentro de los programas.
Por ejemplo :

- Si incluye el numero 12 (el número 12 literal) en un programa, Java lo tomará como un tipo int. Se puede indicar que una literal entera es de tipo long agregándole al final la letra L (l o L) por ejemplo 12L.
- Si incluye el número 3.45 lo tomara como tipo double. Todas las literales de punto flotante se consideran como double en vez de float, para especificar una literal de tipo float ,se agrega la letra F (F o f) a la literal por ejemplo 3.45F
- Las literales de caracteres se expresan mediante un carácter sencillo entre apostrofes, como 'a','#' y '9'. Algunas literales de caracteres representan caracteres que no son imprimibles directamente o accesibles a través del teclado, como mostraremos en la siguiente tabla.

Escape	Significado
\n	Linea nueva
\t	Tabulador
\b	Retroceso
\r	Retorno de carro
\f	Salto de hoja
\\	Diagonal invertida
\'	Apostrofe
\"	Comillas
\o	Octal
\xd	Hexadecimal
\ud	Carácter unicode

- Los literales de cadena constan de una serie de caracteres entre comillas, como por ejemplo:

```
String autor="Luis Joyanes Aguillar";
String nombre= "Carlos";
```

Las cadenas pueden incluir codigos de caracteres de escape como por ejemplo

```
String cadena = " Esto es \nUna prueba\nde caracteres\n";
```

OPERADOR DE ASIGNACIÓN (=)

Sirve para asignar valores a las variables

```
variable = valor;
```

Ejemplo:

```
x=10;
carácter='w'
a=10;
a=a*10;
```

COMENTARIOS EN UN PROGRAMA

Como regla general, cada vez que crea un programa debe de asegurarse de incluir comentarios para explicar el proceso que se realiza.
Java proporciona 2 formas de agregar comentarios al código:

a) Comentario de varias líneas usando la pareja /* y */, como se muestra a continuación:

```
/* Esto es una prueba de un comentario de varias lineas en Java. El compilador de Java los
ignorara por completo. */
```

b) Comentario de una línea , para eso se usa la doble diagonal (//) para comentar el codigo. La doble diagonal indica al compilador que lo que esta a la derecha de ellas (sobre la misma línea) es un comentario
Ejemplo:
int a ; // Se esta declarando una variable entera

OPERADORES ARITMÉTICOS

Java tiene cinco operadores aritméticos básicos

Operador	Propósito	Ejemplo
+	Suma	5 + 25
-	Resta	12-9
*	Multiplicación	13.5 * 90.8
/	División	2.9/1.4
%	Módulo	7%2 = 1

Al utilizar valores aritméticos, Java entrega un resultado del mismo tipo que el operando de mayor precisión. Por ejemplo, al multiplicar un int por un float el resultado es de tipo float. La mayoría de los operadores aritméticos se comportan de forma previsible. La excepción es el operador de división. Si hace una división entre valores enteros el resultado también sera entero. En otras palabras si divide 7 entre 2, el resultado será 3. De esta forma si al hacer una división desea un resultado de punto flotante, debe asegurarse que el tipo de los operandos sea float o double.
Ejemplo:

```
int t;
float r;
int a=10, b=4;
t =10/4;  // El resultado es 2
r= 10f / 4f    // el resultado es 2.5
r= (float) a / (float) b;      // El resultado es 2.5
```

Conversión de un Tipo de Dato Primitivo a otro
Convertir (casting) un valor es el proceso de transformar un valor de un tipo a un valor de otro tipo.

Conversión Implicita

Por lo general no es necesario utilizar ninguna notación de conversión explícita al asignar un tipo de dato primitivo más pequeño a un tipo más grande. Por ejemplo:

```
short  z;
byte  x=40;
z =x;
```

Se esta asignando un valor byte (que puede tener valores entre –128 a 127) a una variable tipo short (que tiene valores entre –32768 a 32767), sin necesidad de una conversión explicita.
A continuación se presenta un resumen de algunas de las asignaciones que no requieren de una conversión explicita en Java. Un tipo puede ser asignado a cualquiera de los tipos que están a su izquierda.

double <= float <= long <= int <= short <= byte

Conversion Explicita

Mediante la conversión explicita se le indica al compilador de Java que es intencional la conversión (y que acepta las consecuencias de cualquier pérdida de precisión).
El formato general de conversión es :

(tipo) valor_a_convertir

Para hacer una conversión explicita, sólo debe rodear al tipo de variable con paréntesis, por ejemplo (float). Las siguientes instrucciones convierten un tipo double a float

```
float resultado;
double valor=12.789561493;
resultado = (float) valor;
```

Operador incremento ++

Permite incrementar el valor de una variable en uno.
Se puede usar el operador ++variable o variable++..
Ejemplo:

```
int alfa=50;
alfa++;  // ahora el valor de alfa es 51

float x=12.56;
++x;      // ahora el valor de x = 13.56
```

Operador decremento --

Permite disminuir el valor de una variable en uno.
Se puede usar el operador –variable o variable--.
Ejemplo :

```
float w = 200.67;
w--; // ahora el valor de la variable es 199.67
int t=34;
--t;  // ahora el valor de la variable es 33
```

El incremento /decremento pre y postfijo

La diferencia entre los operadores prefijo y postfijo es cuando realiza Java el incremento o decremento de la variable.
Al utilizar el valor prefijo (++variable o – variable), java incrementa o decrementa primero la variable y después utiliza el valor de la misma.
Ejemplo:

```
int x=6,y;
y=++x; // Aquí primero incrementa el valor de la variable y luego hace la asignación
```

El valor final de y es 7 y el valor de x es 7

Por otra parte si utiliza operadores postfijos(variable++ o variable--), Java utiliza primero el valor de la variable y luego realiza el operador incremento o decremento.
Ejemplo:

```
int x=6, y;
y=x++ ; // Aquí primero asigna y luego incrementa el valor de x
```

El valor final de y es 6 y el valor de x es 7

Los Applets de Java y las Aplicaciones

Java puede crear dos tipos de programas : applets y aplicaciones.
Un Applet es un programa que se ejecuta dentro de un navegador.
Una Aplicación de Java es un programa independiente del navegador y puede ejecutarse como cualquier otro programa.
En vista de que el applet se ejecuta dentro de un navegador, tiene la ventaja de contar con una ventana y la habilidad para responder a los eventos de la interfaz de usuario del navegador. Además, como los applets están diseñados para ejecutarse a través de la red, Java restringe en gran parte el tipo de accesos que pueden hacer los applets a los archivos del cliente.

ESTRUCTURA DE UNA APLICACIÓN DE JAVA SENCILLA

```
public class HolaMundo{
        public static void main(String args[])
        {
          System.out.println("! Hola Mundo ! ");
        }
}
```

Al escribir una aplicación sencilla de Java se debe definir un método main, que se procesa cuando el programa inicia su ejecución. En el método main se especifican las funciones que realiza el programa.
Toda aplicación se comienza declarando una clase con la palabra *public class* y un nombre, y este nombre tiene que ser idéntico al nombre del archivo que se va a grabar pero con extensión Java.
En el ejemplo al colocar public class HolaMundo se esta declarando la clase llamada HolaMundo.
El archivo se debe grabar como HolaMundo.java igual que el nombre de la clase.
Dentro de main, la aplicación utiliza el metodo System.out.println para desplegar el mensaje "¡Hola Mundo!" en la consola del Sistema.

Para compilar un programa de Java se hace lo siguiente
C:\JDK1.3\BIN>javac HolaMundo.java <Intro>

Al hacer esto se crea un archivo llamado HolaMundo.class que es un archivo que se encuentra en bytecode

Después de esto se utiliza el comando java para interpretar y ejecutar el bytecode de la aplicación.

```
C:\JDK1.3\BIN>java HolaMundo <Intro>
¡ Hola Mundo !
```

- Las instrucciones en un programa Java se escriben en minúsculas
- El símbolo de punto y coma (;) se utiliza para separar un instrucción de la siguiente.
- Un bloque de código es una serie de instrucciones y/o declaraciones de Java rodeadas por un par de llaves {}
  Por ejemplo para comenzar la clase comienza con { y termina con }. Asi mismo el método main comienza con un { y termina con un }. El uso de los bloque de código es importante para indicar que se van a realizar un conjunto de instrucciones.

ENTRADA Y SALIDA POR TERMINAL

La E/S básica por terminal con formato se lleva a cabo mediante *readLine* y *println* . La entrada estándar es System.in y la salida estándar es System.out.
El mecanismo básico para E/S con formato utiliza el tipo String. En la salida + combina dos valores de tipo String. Si el segundo argumento no es un valor de tipo String, se crea un valor temporal de tipo String para el, siempre que sea de tipo primitivo. Estas conversiones al tipo String se pueden definir también para objetos.
Ejemplo:

```
System.out.println("El resultado es "+R);
```

Un modo sencillo de realizar una lectura de la entrada es leer una sola línea almacenándola en un objeto de tipo String, para lo que se emplea readLine. Este método lee hasta que encuentra un final de línea o el final del fichero. Los caracteres leídos salvo el final de Línea (en el caso de que se encuentre), se devuelven como un nuevo String. Para emplear readLine, se debe construir un objeto BufferedReader sobre un objeto InputStreamReader que a su vez se crea a partir de System.in.
Ejemplo:

```
BufferedReader in = new BufferedReader( new InputStreamReader(System.in));
String  nombre;
System.out.println("Introduzca una  nombre : ");
nombre = in.readLine();
```

En este ejemplo se ha declarado una variable nombre de tipo String y se esta leyendo desde la consola utilizando un objeto llamado in el cual llama al método readLine para leer una cadena y asignarle a la variable nombre.

Si se desea leer valores numéricos enteros o reales, primero se lee como cadena y luego se convierten estos valores a numéricos. Para esto se utilizan los siguientes métodos
Integer.parseInt(cadena) : Convierte una cadena a un valor entero
Float.parseFloat(cadena) . Convierte una cadena a un valor real tipo float
Double.parseDouble(cadena) : Convierte una cadena a un valor real tipo double.

Aquí presentaremos algunos aplicaciones sencillas de Ejemplo:

1) Hacer un programa para calcular el area de un triangulo dada
la base y la altura

```
import java.io.*;

public class AreaTriangulo{
  public static void main(String arg[]) throws IOException
  {
    float base,altura,area;
    BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
    System.out.print("Ingrese base del triangulo :");
    base=Float.parseFloat(in.readLine());
    System.out.print("Ingrese altura del triangulo :");
    altura=Float.parseFloat(in.readLine());
    area=base*altura/2;
    System.out.println("El area del triangulo es : "+area);
  }
}
```

2) Calcular el perimetro, el area y la diagonal de un rectangulo si
se ingresan los lados.

```
import java.io.*;

public class Rectangulo{
  public static void main(String arg[]) throws IOException
  {
   double lado1,lado2,perimetro,area,diagonal;
    BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
    System.out.print("Ingrese valor del primer lado :");
    lado1=Double.parseDouble(in.readLine());
    System.out.print("Ingrese valor del segundo lado :");
    lado2=Double.parseDouble(in.readLine());
    perimetro=2*(lado1+lado2);
    area=lado1*lado2;
    diagonal=Math.sqrt(Math.pow(lado1,2)+Math.pow(lado2,2));
    System.out.println("El perimetro es : "+perimetro);
    System.out.println("El area es : "+area);
    System.out.println("La diagonal es : "+diagonal);
  }
}
```

3) Calcular el salario neto de un trabajador. Se debe leer el nombre,
horas trabajadas, precio de la hora y sabiendo que los impuestos
aplicados son el 10 por ciento sobre el salario bruto.

```
import java.io.*;

public class Trabajador{
  public static void main(String arg[]) throws IOException
  {
    String nombre;
    double sb,sn,ph,ht;
    BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
    System.out.print("Ingrese nombre del trabajador :");
    nombre=in.readLine();
    System.out.print("Ingrese numero de horas trabajadas :");
    ht=Double.parseDouble(in.readLine());
    System.out.print("Ingrese precio de la hora :");
    ph=Double.parseDouble(in.readLine());
    sb=ph*ht;
    sn=sb-0.10*sb;
    System.out.println("El Salario neto del trabajador : "+nombre+ " es : "+sn);
  }
}
```

4) Hacer un programa para convertir metros a pies y pulgadas
1 metro = 39.37 pulgadas 1 metro = 3.2 pies

```
public class Conversion{
  public static void main(String arg[]) throws IOException
  {
    double metros,pies,pulgadas;
    BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
    System.out.print("Ingrese valor en Metros :");
    metros=Double.parseDouble(in.readLine());
    pies=metros*3.2;
    pulgadas=metros*39.37;
    System.out.println("El valor en pies es : "+pies);
    System.out.println("El valor en pulgadas es : "+pulgadas);
  }
}
```

Operadores Relacionales

Operador	Función
>	Mayor que
>=	Mayor o igual que
<	Menor que
<=	Menor o igual que
= =	Igual que
! =	Diferente que

Operadores Lógicos

Operador	Funcion
&&	Y
||	O
!	Negación

Estructura Selectiva Simple : if .... else

Se utiliza para tomar decisiones. El programa prueba una condición con la instrucción if. Si la condición es verdadera, el programa ejecuta una instrucción (o un bloque de instrucciones relacionadas). Por el contrario si la condición es falsa, el programa puede ejecutar un conjunto diferente de instrucciones especificado por la cláusula else.
La sintaxis de la sentencia else es la siguiente:

```
if(condición)
  instruccion1;
else
   instruccion2;
```

o también.

```
if (condicion)
  {
    bloque de  Instrucciones1;
  }
else
   {
     bloque de Instrucciones2;
   }
```

La sentencia else es opcional. A veces se puede omitir

```
if(condición)
{
   Instrucciones;
}
```

1) Hacer un programa para que se ingrese 2 números y se reporte el mayor de ellos.

```
import java.io.*;
public class p201{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    float num1,num2;
    System.out.print("Ingrese Primer Numero :");
    num1=Float.parseFloat(br.readLine());
    System.out.print("Ingrese Segundo Numero :");
    num2=Float.parseFloat(br.readLine());
    if(num1>num2)
      System.out.println("El Mayor es : "+num1);
    else
        if(num1<num2)
            System.out.println("El Mayor es : "+num2);
      else
            System.out.println("Los numeros son iguales");
  }
}
```

2) Hacer un programa para determinar si un numero entero A es divisible por otro B.

```
import java.io.*;

public class p202{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int num1,num2;
    System.out.print("Ingrese primer numero:");
    num1=Integer.parseInt(br.readLine());
    System.out.print("Ingrese segundo numero:");
    num2=Integer.parseInt(br.readLine());
```

```
   if(num1 % num2==0)
      System.out.println(num1+" es divisible por "+num2);
   else
      System.out.println(num1+" No es divisible por "+num2);
 }
}
```

3) Hacer un programa para que calcule e imprima los valores de las raíces reales de una ecuación cuadrática

$$AX^2 + BX + C = 0$$

```
import java.io.*;
import java.lang.*;
public class P203{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    double A,B,C,X1,X2,D;
    System.out.print("Coeficiente A: ");
    A=Double.parseDouble(br.readLine());
    System.out.print("Coeficiente B: ");
    B=Double.parseDouble(br.readLine());
    System.out.print("Coeficiente C: ");
    C=Double.parseDouble(br.readLine());
    if(A==0)
      System.out.println("NO ES UNA ECUACION CUADRATICA");
    else
       {
         D=Math.pow(B,2)-4*A*C;
         if(D>=0)
           {
             X1=(-B+Math.sqrt(D))/(2*A);
             X2=(-B-Math.sqrt(D))/(2*A);
             System.out.println("X1 = "+X1);
             System.out.println("X2 = "+X2);
           }
         else
             System.out.println("RAICES IMAGINARIAS ");
       }
  }
}
```

4) La tasa de interés sobre un préstamo es del 8% si la cantidad es menor o igual que S/.200, pero es del 6% si excede a 200. Hacer un programa para que ingrese la cantidad y reporte el interés y el monto total.

```
import java.io.*;

public class p204{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
```

```
   double cantidad, interes,montoTotal;
   System.out.print("Cantidad a prestar : ");
   cantidad=Double.parseDouble(br.readLine());
   if(cantidad<=200)
     interes=cantidad*0.06;
   else
     interes=cantidad*0.08;
   montoTotal=cantidad+interes;
   System.out.println("El interes es : "+interes);
   System.out.println("El monto total es   : "+montoTotal);
  }
}
```

5) Hacer un programa de tal manera que se ingrese las 2 evaluaciones de un alumno y reporte APROBADO si el promedio es mayor o igual a 10.5 y DESAPROBADO en caso contrario

```
import java.io.*;

public class P205{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    double nota1, nota2, promedio;
    System.out.print("NOTA 1 : ");
    nota1=Double.parseDouble(br.readLine());
    System.out.print("NOTA 2 : ");
    nota2=Double.parseDouble(br.readLine());
    promedio=(nota1+nota2)/2;
    if(promedio>=10.5)
        System.out.println("APROBADO CON promedio "+promedio);
    else
        System.out.println("DESAPROBADO CON promedio "+promedio);
  }
}
```

6) La comisión de las ventas totales es como sigue :

a) Si ventas < 80, entonces no hay comisión
b) Si 80 <= ventas <= 600 entonces la comisión es igual al 12% de las ventas.
c) Si ventas>600 entonces las comisión es igual al 15% de las ventas.

```
import java.io.*;

public class p206{
  public static void main(String arg[]) throws IOException
  {
    double ventas, comision;
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    System.out.print("INGRESE ventas : ");
    ventas=Double.parseDouble(br.readLine());
```

```
   if(ventas<80)
     System.out.println("NO HAY comision");
   else
      {
       if(ventas<=600)
           comision=ventas*0.12;
       else
           comision=ventas*0.15;
       System.out.println("La comisión es : "+comision);
      }
 }
}
```

7) Hacer un programa para calcular el pago semanal de un trabajador. Se debe ingresar el nombre, pago por hora y el numero de horas trabajadas Si normalmente se trabaja 40 horas a la semana y por cada hora extra trabajada se paga 1.5 veces la hora normal, reportar el nombre y el pago semanal del trabajador.

```
import java.io.*;

public class p207{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    String nombre;
    double ps,ph,ht;
    System.out.print("Ingrese nombre del trabajador :");
    nombre=br.readLine();
    System.out.print("Ingrese pago por hora :");
    ph=Double.parseDouble(br.readLine());
          System.out.print("Ingrese numero de horas trabajadas :");
    ht=Double.parseDouble(br.readLine());
    if (ht<=40)
        ps=ph*ht;
    else
        ps=40*ph+(ht-40)*1.5*ph;
    System.out.println("El Pago semanal es : "+ps);
  }
}
```

8) Se repartirá la herencia entre los hijos de un señor como sigue: Si la cantidad de hijos es menor que 4; se repartirá exactamente entre el numero de hijos; si son 4 o mas hijos, la mitad le tocara al hermano mayor y el resto se dividirá entre los demás hermanos. Se debe ingresar la herencia y el numero de hijos.

```
import java.io.*;

public class p208{
  public static void main(String arg[]) throws IOException
  {
```

```
   BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
   double herencia,t,m,d;
   int numHijos;
   System.out.print("Ingrese herencia :");
   herencia=Double.parseDouble(br.readLine());
   System.out.print("Ingrese numero de hijos :");
   numHijos=Integer.parseInt(br.readLine());
   if(numHijos<4)
    {
      t=herencia/numHijos;
      System.out.println("A todos les toca : "+t);
    }
   else
    {
       m=herencia/2;
       d=m/(numHijos-1);
       System.out.println("Al mayor le toca : "+m);
       System.out.println("A los demas : "+d);
    }
 }
}
```

9) Una empresa comercial desea hacer un programa para calcular el precio neto de un articulo de acuerdo a lo siguiente:

a) Si la venta es al contado se le da el 40% de descuento.
b) Si la venta es a plazos y:
t<12 meses se recarga el 30%
t>=12 meses se recarga el 60%

Se debe ingresar el precio del articulo, el codigo de venta (c) contado, (p) plazos y si la venta es a plazos se debe ingresar el tiempod de pago

```
import java.io.*;

public class p209{
  public static void main(String arg[]) throws IOException
  {
   BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
   String cod;
   double pa,pn,t;
   System.out.print("Ingrese Precio del Articulo :");
   pa=Double.parseDouble(br.readLine());
   BufferedReader C = new BufferedReader(new InputStreamReader(System.in));
   System.out.print("Ingrese Codigo de Venta (c) Contado (p) plazos : ");
   cod=br.readLine();
   if (cod.equals("c"))
     pn=pa*0.60;
   else
     {
       System.out.print("Ingrese tiempo de pago : ");
       t=Double.parseDouble(br.readLine());
```

```
        if(t<12)
            pn=1.3*pa;
        else
            pn=1.6*pa;
       }
    System.out.println("El Precio Neto del Articulo es : "+pn);
  }
}
```

10) En un triangulo se cumple lo siguiente :
s>a, s>b, s>c donde s: semiperímetro , a,b,c:Lados del triangulo
Hacer un programa para que se ingresen los valores de los lados
del triangulo y si estos valores cumplen las condiciones calcular
el área del triangulo en caso contrario reportar DATOS INCORRECTOS

```
import java.io.*;

public class p210{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    double a,b,c,s,area;
    System.out.print("Ingrese Valor del Primer lado :");
    a=Double.parseDouble(br.readLine());
    System.out.print("Ingrese Valor del Segundo lado :");
    b=Double.parseDouble(br.readLine());
    System.out.print("Ingrese Valor del Tercer lado :");
    c=Double.parseDouble(br.readLine());
    s=(a+b+c)/2;
    if(s>a && s>b && s>c)
     {
      area=Math.sqrt(s*(s-a)*(s-b)*(s-c));
      System.out.println("El area del Triangulo es : "+area);
     }
    else
       System.out.println("DATOS INCORRECTOS");
  }
}
```

11) Calcular el valor de la función de acuerdo a lo siguiente :
$y = x^2 + 5$ Si x<=0
y = 3x-1 si 0<x<2
$y= x^2 - 4x + 5$ si x>=2
Se debe ingresar el valor de x y reportar el valor de y

```
import java.io.*;

public class p211{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    double x,y;
```

```
   System.out.print("Ingrese valor de x : ");
   x=Double.parseDouble(br.readLine());
   if(x<=0)
       y=Math.pow(x,2)+5;
   else
      if(x<2)
          y=3*x-1;
      else
          y=Math.pow(x,2)-4*x+5;
   System.out.println("El valor de y es : "+y);
  }
}
```

12) Los empleados de una fabriva trabajan en dos trunos: diurno y nocturno.
Se desea calcular el jornal diario de acuerdo a los siguiente puntos:
- La tarifa de las horas diurnas es de S/. 1.5
- La tarifa de las horas nocturnas es de S/. 2.25
En caso de ser domingo la tarifa aumentara en S/.1 en el turno diurno
y S/. 1.25 en el turno nocturno.
Se debe leer el turno, las horas trabajadas y el dia de la semana.

```
import java.io.*;

public class p212{
  public static void main(String arg[]) throws IOException
  {

   BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
   String valor,turno,dia;
   double ht,jd;

   System.out.print("Ingrese turno (d) diurno (n) nocturno :");
   turno=br.readLine();
   System.out.print("Ingrese Numero de Horas trabajadas : ");
   ht=Double.parseDouble(br.readLine());
   System.out.print("Ingrese Dia de Semana (d) domingo (otra letra ) otro dia  :");
   dia=br.readLine();
   if(turno.equals("d"))
     if(!dia.equals("d"))
       jd=1.5*ht;
     else
       jd=2.5*ht;
    else
     if(!dia.equals("d"))
       jd=2.25*ht;
     else
       jd=3.5*ht;
   System.out.println("El Jornal diario es : "+jd);
  }
}
```

Clase String

Esta clase almacena una cadena de caracteres. Una cadena es una serie de caracteres que se trata como una unidad. Una cadena puede incluir letras, dígitos y diversos caracteres especiales como +,-,*,/,$ y otros.
Una cadena en Java es un objeto de la clase String.
Los literales de cadena o constantes de cadena en Java se escriben como una secuencia de caracteres cerradas entre comillas.
Ejemplo:

"Computación", "Java es fácil",

Métodos para la manipulación de Cadenas.

cadena.**length()** : El método length Devuelve el numero de Caracteres de una cadena.
Ejemplo:

```
String nombre = "Francisco";
int n;
n = cadena.length();
System.out.println("El nombre : "+ nombre + "tiene "+ n + "Caracteres");
```

cadena1.**equals**(cadena2) : El método equals devuelve true si las dos cadenas son iguales y false en caso contrario..

cadena1.**equalsIgnoreCase**(cadena2) : El método equalsIgnoreCase no toma en cuenta la diferencia entre letras mayúsculas y minúsculas de cada String al realizar la comparación.
cadena1.**compareTo**(cadena2) : El método compareTo devuelve un numero entero:

- > 0 Si cadena1 es mayor que cadena2
- = 0 Si las cadenas son exactamente iguales
- < 0 Si cadena1 es menor que cadena2

Este método emplea una comparación lexicográfica.

cadena1.**compareToIgnoreCase**(cadena2) : El método compareToIgnoreCase no toma en cuenta la diferecia entre las mayúsculas y minúsculas de cada String al realizar la comparación.

cadena.**charAt**(indice) : El método charAt delvuelve el carácter que indica el indice.
Ejemplo:

```
String nombre = "Felipe";
Char x;
X=nombre.charAt(3); // x toma el valor del carácter i.
```

Los indices de una cadena comienzan en 0.

cadena.**toUpperCase**() : Convierte la cadena a mayúsculas

cadena.**toLowerCase()** : Convierte la cadena en minúsculas

cadena.**trim()** : Elimina los espacios al inicio y al final de la cadena.

cadena1.**startsWith**(cadena2) : Devuelve verdadero si cadena1 comienza con cadena2.

cadena1.**endsWith**(cadena2) : Devuelve verdadero si cadena1 finaliza con cadena2.

cadena.**indexOf(**caracter) : Devuelve el índice de la primera ocurrencia del carácter.

cadena.**indexOf**(carácter,posicion) : Devuelve el índice de la primera ocurrencia del carácter , comenzando la búsqueda a partir de posición.

cadena1.**indexOf**(cadena2) : devuelve el índice de la primera ocurrencia de cadena2 en cadena1.

cadena1.**indexOf**(cadena2,posición): Devuelve el índice de la primera ocurrencia de la cadena2 en cadena1, comenzado la búsqueda a partir de posición.

cadena.**lastIndexOf(**caracter) : Devuelve el índice de la ultima ocurrencia del carácter.

cadena.**lastIndexOf**(carácter,posicion) : Devuelve el índice de la ultima ocurrencia del carácter , comenzando la búsqueda a partir de posición.

cadena1.**lastIndexOf**(cadena2) : devuelve el índice de la ultima ocurrencia de cadena2 en cadena1.

cadena1.**lastIndexOf**(cadena2,posición): Devuelve el índice de la ultima ocurrencia de la cadena2 en cadena1, comenzado la búsqueda a partir de posición.

cadena(caracter1,caracter2) : Devuelve una nueva cadena que es el resultado de reemplazar todas las ocurrencias del caracter1 por el caracter2

cadena.substring(indice) : Devuelve una subcadena que esta conformada por todos los caracteres que comienzan en indice e incluye todos los caracteres siguientes hasta el fin de la cadena.

Cadena.substring(indiceInicial, indiceFinal) : Devuelve una subcadena que esta conformada por todos los caracteres cuyo indice comienza en indiceInicial e incluye todos los caracteres hasta el IndiceFinal-1.

String.valueOf() : Convierte tipos primitivos a cadenas. La clase String sobrecarga el método valueOf para soportar cada uno de los tipos primitivos de Java.

Ejemplo:

```
String cad1 = String.valueOf(12);
String cad2 = String.valueOf(true);
String  cad3 = String.valueOf('T');
String cad4 = String.valueOf(12.9867);
```

ESTRUCTURA SELECTIVA MÚLTIPLE : switch

```
switch(expresión)
{
  case  cte1 :    Instrucciones1;
                  break;
  case  cte2 :    Instrucciones2:
                  break;
  case   cte3 :   Instrucciones3;
                  break;
   .
   .
   .
   default :   Instrucciones;
}
```

expresión : Puede ser de los tipos primitivos byte, char, short o int no puede ser de otro tipo. Esta expresión se compara con cada uno de las constantes que se encuentran en los case, si es igual a alguna de ellas se ejecutan las expresiones correspondientes y se sale del switch. Si no es igual a ninguna de ellas se ejecutan las instrucciones que siguen a default.
La sentencia default es opcional.
En la sentecia switch solo se compara por igualdad no por otra relación.

1) Ingresar un numero entre 1 y 12 y reportar el mes que le corresponde.

```
import java.io.*;

public class p401{
 public static void main(String arg[]) throws IOException
 {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int num;
  System.out.print("Ingrese numero entre 1 y 12 : ");
  num=Integer.parseInt(br.readLine());
  switch(num)
  {
    case 1  : System.out.println("ENERO");
              break;
    case 2  : System.out.println("FEBRERO");
              break;
    case 3  : System.out.println("MARZO");
              break;
    case 4  : System.out.println("ABRIL");
              break;
    case 5  : System.out.println("MAYO");
              break;
    case 6  : System.out.println("JUNIO");
              break;
    case 7  : System.out.println("JULIO");
              break;
```

```
      case 8  : System.out.println("AGOSTO");
                break;
      case 9  : System.out.println("SETIEMBRE");
                break;
      case 10 : System.out.println("OCTUBRE");
                break;
      case 11 : System.out.println("NOVIEMBRE");
                break;
      case 12 : System.out.println("DICIEMBRE");
                break;
      default :
                System.out.print("Numero equivocado");
    }
  }
}
```

2) Ingresar un numero entero, y si este termina en 2,5 u 8 reportar el cuadrado del numero, si este termina en 4,7 o 9 reportar el numero multiplicado por 5 y reportar el mismo numero en otro caso.

```
import java.io.*;

public class p402{
  public static void main(String arg[]) throws IOException
  {
          BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int num;
    System.out.print("Ingrese numero entero : ");
    num=Integer.parseInt(br.readLine());
    switch(num % 10)
    {
      case 2  : case 5 : case 8:
                System.out.print("El cuadrado del numero es : "+num*num);
                break;
      case 4  : case 7 : case 9:
                System.out.print("El Numero multiplicado por 5 es : "+num*5);
                break;
      default :
                System.out.print("El numero ingresado es :"+num);
    }
  }
}
```

3) Ingresar una letra entre a y e y reportar el mensaje de acuerdo a:

a	excelente
b	bueno
c	regular
d	malo
e	pesimo

```
import java.io.*;

public class p403{
  public static void main(String arg[]) throws IOException
  {
          BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
          String cadena;
    char letra;
    System.out.print("Ingrese letra (a-e) : ");
    cadena=br.readLine();
    letra=cadena.charAt(0);
    switch(letra)
    {
      case 'a': case 'A':
              System.out.println("EXCELENTE");
              break;
      case 'b': case 'B':
              System.out.println("BUENO");
              break;
      case 'c': case 'C':
              System.out.println("REGULAR");
              break;
      case 'd': case 'D':
              System.out.println("MALO");
              break;
      case 'e': case 'E':
              System.out.println("PESIMO");
              break;
      default :
              System.out.println("letra equivocada");
    }
  }
}
```

4) Ingresar 2 numeros y luego escoger la operacion que se quiere hacer con ellos y reportar el resultado

```
import java.io.*;

public class p404{
  public static void main(String arg[]) throws IOException
  {
    double num1,num2,res;
    int op;
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    System.out.print("Ingrese Primer Numero : ");
    num1=Double.parseDouble(br.readLine());
    System.out.print("Ingrese Segundo Numero : ");
    num2=Double.parseDouble(br.readLine());
    System.out.println("Escoger operacion :");
    System.out.println("[1] Suma");
    System.out.println("[2] Resta");
```

```
   System.out.println("[3] Multiplicacion");
   System.out.println("[4] Division");
   System.out.print("Ingrese Opcion (1-4) : ");
   op=Integer.parseInt(br.readLine());
   switch(op)
   {
     case 1  : res=num1+num2;
               System.out.println("La Suma es : "+res);
               break;
     case 2  : res=num1-num2;
               System.out.println("La Resta es : "+res);
               break;
     case 3  : res=num1*num2;
               System.out.println("La multiplicacion es :"+res);
               break;
     case 4  : if(num2!=0)
                {
                  res=num1/num2;
                  System.out.println("La division es :"+res);
                 }
               else
                  System.out.println("No se puede dividir entre cero");
               break;
     default :
               System.out.println("Numero de Opcion Fuera de Rango");
   }
 }
}
```

PROCESOS REPETITIVOS

1) while (mientras)

```
while(condición)
{
    Instrucciones;
}
```

En este proceso se verifica la *condición*, si esta es verdadera se ejecutan *Instrucciones* y automáticamente se vuelve de nuevo a verificar la condición. Este proceso se ejecuta hasta que la condición llegue a ser falsa.
A este proceso se le conoce como de Entrada controlada, pues primero se verifica la condición y luego se ejecutan las instrucciones

2) do... while (Hacer ... mientras)

```
do{

    Instrucciones;

}while(condición);
```

En este proceso primero se realizan las *instrucciones* y luego se verifica la *condición*, si esta es verdadera, se realizan de nuevo las *Instrucciones*. Este proceso se ejecuta hasta que la *condición* llegue a ser falsa.
A este proceso se le conoce como de Salida controlada pues la condición se encuentra al final.

3) **for**

```
for(exp1; exp2; exp3)
{
   Instrucciones;
}
```

Donde :
exp1 : Instrucciones de Inicialización. Se ejecuta 1 vez y se va a exp2.
exp2 : Condición que controla el proceso Repetitivo. Si esta expresión es verdadera se ejecutan las Instrucciones y se va a exp3. Si esta expresión es falsa el proceso termina.
exp3 : Instrucciones de incremento de variables. Se ejecutan y luego se va a exp2.

Las sentencias break y continue
Los sentencias break y continue alteran el flujo de control.

Sentencia break
La sentencia break, cuando se ejecuta en una estructura while, for, do..while o switch, causa la salida inmediata de la estructura. La ejecución continua con el siguiente instrucción después de la estructura. Un uso común de la sentencia break es terminar antes de tiempo de un ciclo (for, while, do..while) o saltarse el resto de una estructura switch
Ejemplo:

Setencia continue
La sentencia continue, cuando se ejecuta en una estructura while, for o do...while, se salta el resto de las instrucciones del cuerpo de la estructura y continua con la siguiente iteración del ciclo.
En las estructuras while, do...while, la prueba para continuar el ciclo se evalúa inmediatamente después de ejecutarse la sentencia continue. En la estructura for, se ejecuta la expresión de incremento y luego se ejecuta la prueba para ejecutar el ciclo.
Ejemplo:

Las sentencias break y continue rotulados
La sentencia **break** solo puede causar la salida de la estructura while, for, do..while o switch que lo encierra directamente. Si se quiere salir de una serie de estructuras anidadas, se puede usar la sentencia break rotulada. Esta sentencia cuando se ejecuta en un while, for, do...while o switch, causa la salida inmediata de esa estructura y de cualquier cantidad de estructuras que la encierren; la ejecución del programa continua con la primera instrucción después de la sentencia compuesta rotulada (Es decir una serie de instrucciones encerrados en llaves y precedidos por un rótulo)

La sentencia **continue** rotulada, al ejecutarse en una estructura repetitiva (while, for o do..while), se salta el resto de las instrucciones del cuerpo de esa estructura y cualquier cantidad de estructuras repetitivas que la encierren y continua con la siguiente iteración de la estructura de repetición rotulada (es decir, una estructura repetitiva precedida por un rótulo) que la encierra.

Ejercicios de Procesos Repetitivos

1) Se desea calcular independientemente la suma de los pares e impares comprendidos entre 1 y 50 (incluidos los extremos).

```
import java.io.*;

public class p301{
  public static void main(String arg[]) throws IOException
  {
   int i,sp=0,si=0;
   for(i=1;i<=50;i++)
     if(i%2==0) sp=sp+i;
     else si=si+i;
   System.out.println("La suma de pares es : "+sp);
   System.out.println("La suma de impares es : "+si);
  }
}
```

2) Calcular y visualizar la suma y el producto de los números impares comprendidos entre 20 y 80.

```
import java.io.*;

public class p302{
  public static void main(String arg[]) throws IOException
  {
   int i,si=0;
   double pi=1;
   for(i=20;i<=80;i++)
     if(i%2!=0)
       {
         si=si+i;
         pi=pi*i;
       }
   System.out.println("La suma es : "+si);
   System.out.println("El producto es : "+pi);
  }
}
```

3) Leer n numeros enteros y obtener el promedio de los positivos y el promedio de los negativos.

```
import java.io.*;

public class p303{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int n,i,x,sp=0,sn=0,cp=0,cn=0;
    double pp,pn;
    do{
```

```
      System.out.print("Valor de n : ");
      n=Integer.parseInt(br.readLine());
    }while(n<=0);
    for(i=1;i<=n;i++)
    {
       System.out.print("Ingrese numero : ");
       x=Integer.parseInt(br.readLine());
       if(x>0)
        {
          cp++;
          sp=sp+x;
        }
       else
          if(x<0)
           {
             cn++;
             sn=sn+x;
           }
    }
    if(cp>0)
     {
       pp=(double)sp/cp;
       System.out.println("El Promedio de positivos es : "+pp);
     }
    else
       System.out.println("No se Ingresaron Positivos");
    if(cn>0)
     {
       pn=(double)sn/cn;
       System.out.println("El Promedio de Negativos es : "+pn);
     }
    else
       System.out.println("No se Ingresaron Negativos");

  }
}
```

4) Calcular la suma de los cuadrados de los números desde el 1 hasta el 15.

```
import java.io.*;

public class p304{
  public static void main(String arg[]) throws IOException
  {
   int i,sc=0;
   for(i=1;i<=15;i++)
     sc=sc+i*i;
   System.out.println("La suma de los cuadrados de los primeros 15 números  es : "+sc);
  }
}
```

5) Se ingresan n numeros. Se pide calcular el promedio de ellos

```
import java.io.*;
public class p305{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int n,i;
    double x,s=0,p;
    do{
       System.out.print("Valor de n : ");
       n=Integer.parseInt(br.readLine());
    }while(n<=0);
    for(i=1;i<=n;i++)
    {
        System.out.print("Ingrese numero : ");
        x=Double.parseDouble(br.readLine());
        s=s+x;
    }
    p=s/n;
    System.out.println("El Promedio es : "+p);
  }
}
```

6) Ingresar n numeros enteros, visualizar la suma de los numeros pares de la lista, cuantos pares existen y cual es la media de los numeros impares.

```
import java.io.*;
public class p306{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int n,i,x,sp=0,si=0,cp=0,ci=0;
    double mi;
    do{
       System.out.print("Valor de n : ");
       n=Integer.parseInt(br.readLine());
    }while(n<=0);
    for(i=1;i<=n;i++)
    {
        System.out.print("Ingrese numero : ");
        x=Integer.parseInt(br.readLine());
        if(x%2==0)
         {
           cp++;
           sp=sp+x;
         }
        else
           {
             ci++;
             si=si+x;
           }
    }
```

```
   if(cp>0)
     {
       System.out.println("La suma de los numeros pares es : "+sp);
       System.out.println("La cantidad de numeros pares es : "+cp);
     }
    else
       System.out.println("No se Ingresaron numeros pares");
    if(ci>0)
     {
       mi=(double)si/ci;
       System.out.println("La media de los impares es : "+mi);
     }
    else
       System.out.println("No se Ingresaron numeros impares");
 }
}
```

7) Desarrolle un programa que determine en un conjunto de numeros naturales.
   a) Cuantos son menores de 15
   b) Cuantos son mayores de 50
   c) Cuantos estan comprendidos entre 25 y 45.

```
import java.io.*;

public class p307{
  public static void main(String arg[]) throws IOException
 {
   BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
   int n,i,c1=0,c2=0,c3=0;
   double x;
   do{
      System.out.print("Valor de n : ");
      n=Integer.parseInt(br.readLine());
   }while(n<=0);
   for(i=1;i<=n;i++)
    {
       System.out.print("Ingrese numero : ");
       x=Double.parseDouble(br.readLine());
       if(x<15) c1++;
       if(x>50) c2++;
       if(x>25 && x<45) c3++;
    }
   System.out.println("La cantidad de numeros menores que 15 es : "+c1);
   System.out.println("La cantidad de numeros mayores de 50 es : "+c2);
   System.out.println("La cantidad de numeros compredios entre 25 y 45 es : "+c3);
  }
}
```

8) Calcular el factorial de un numero n>=0

```
import java.io.*;
```

```
public class p308{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int n,i;
    double f=1;

    do{
      System.out.print("Ingrese numero positivo o cero : ");
      n=Integer.parseInt(br.readLine());
    }while(n<0);
    for(i=1;i<=n;i++)
      f=f*i;
    System.out.println("El factorial es : "+f);
  }
}
```

10) Imprimir las 10 primeras potencias de 4.

```
import java.io.*;

public class p310{
  public static void main(String arg[]) throws IOException
  {
    int i;
    double p=1;
    System.out.println("Las 10 primeras potencias de 4 son");
    for(i=1;i<=10;i++)
    {
      p=p*4;
      System.out.println(4+" elevado a la "+i+" es "+p);
    }
  }
}
```

11) Calcular la suma de los n terminos de la serie :

s=1 - 1/2 + 1/3 - 1/4 + 1/5 - 1/6 + ... 1/n

```
import java.io.*;

public class p311{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int n,i;
    double s=0;
    do{
      System.out.print("Valor de n : ");
      n=Integer.parseInt(br.readLine());
    }while(n<=0);
```

```
   for(i=1;i<=n;i++)
    {
       if(i%2==0) s=s-1.0/i;
       else s=s+1.0/i;
    }
   System.out.println("La sumatoria es : "+s);
 }
}
```

12) Ingresar n números, Calcular el máximo y el mínimo de ellos.

```
import java.io.*;
public class p312{
  public static void main(String arg[]) throws IOException
  {
    int n,i;
    double x,maximo,minimo;
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    do{
       System.out.print("Valor de n : ");
       n=Integer.parseInt(br.readLine());
    }while(n<=0);
    maximo=-1e30;
    minimo=1e30;
    for(i=1;i<=n;i++)
    {
       System.out.print("Ingrese numero : ");
       x=Double.parseDouble(br.readLine());
       if(x>maximo) maximo=x;
       if(x<minimo) minimo=x;
    }
    System.out.println("El maximo es : "+maximo);
    System.out.println("El minimo es : "+minimo);
  }
}
```

3) Realizar un programa que escriba los n terminos de la serie de
Fibonacci
1, 1, 2, 3, 5, 8, 13, 21, ...

```
import java.io.*;
public class p313{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int n,i;
    double p=1,s=0,t;
    do{
       System.out.print("Numero de terminos : ");
       n=Integer.parseInt(br.readLine());
    }while(n<=2);
```

```
   for(i=1;i<=n;i++)
   {
      t=p+s;
      System.out.print(t+"  ");
      p=s;
      s=t;
   }
   System.out.println();
 }
}
```

14) Leer Numeros (el ultimo numero es -99) y obtener el mayor.

```
import java.io.*;

public class p314{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int n,i=0;
    double x,mayor;
    mayor=-1e30;
    do{
        System.out.print("Ingrese numero (-99 para finalizar) : ");
        x=Double.parseDouble(br.readLine());
        if(x!=-99)
         {
           i++;
           if(x>mayor) mayor=x;
         }
    }while(x!=-99);
    if(i>0)
      System.out.println("El mayor es : "+mayor);
    else
      System.out.println("No se ingresaron numeros");
  }
}
```

15) Calcular la sumatoria
s = 1 + x + x^2/2! + x^3/3! + x^4/4! + ... + x^n/n!
Se debe ingresar x real y n entero positivo

```
import java.io.*;

public class p315{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int n,i;
    double p=1,x,f=1,s=1;
```

```
   System.out.print("Ingrese valor de x : ");
   x=Double.parseDouble(br.readLine());
   do{
     System.out.print("Valor de n : ");
     n=Integer.parseInt(br.readLine());
   }while(n<0);
   for(i=1;i<=n;i++)
    {
      f=f*i;
      p=p*x;
      s=s+p/f;
    }
   System.out.println("La sumatoria es : "+s);
 }
}
```

METODOS

Los métodos en java permiten modularizar sus programas. Todas las variables declaradas en las definiciones de métodos son variables locales es decir solo se conocen en el método que se definen. Casi todos los métodos tienen una lista de parámetros que permiten comunicar información entre métodos. Los parámetros de un método también son variables locales.

La sintaxis para declarar un método es la siguiente :

```
tipo_de_valor_devuelto  nombre_del_método ( lista_de_parámetros)
{
        Declaraciones e instrucciones
}
```

Donde :
tipo_de_valor_devuelto : Indica el tipo de valor que se devuelve al método invocador. Si un método no devuelve un valor, el tipo_de_valor_devuelto se declara como void.
nombre_del_método : Es cualquier identificador válido
lista_de_parámetros : Es una lista separada por comas que contiene las declaraciones de las variables que se pasarán al método. Si un método no recibe parámetros las lista_de_parámetros se deja vacía.
El cuerpo del método es el conjunto de declaraciones e instrucciones que constituyen el método.
Cuando un programa encuentra una llamada a un método, se transfiere el control desde el punto de invocación al método invocado, se ejecuta el método y el control regresa al invocador.
Un método invocado puede devolver el control al invocador de tres maneras.
Si el método no devuelve valor, el control se devuelve al llegar a la llave derecha que termina el método o ejecutando la sentencia

```
return;
```

Si el método devuelve un valor, la sentencia

```
return expresión
```

Devuelve el valor de Expresión.
Por ejemplo si se desea calcular la suma de dos números enteros. Se podría definir el siguiente método:

```
int suma(int x, int y)
{
   return x+y;
}
```

el tipo_de_valor_devuelto es int, tiene dos parámetros enteros x e y.

Supongamos que deseamos un método para mostrar el mensaje "Java es fácil de aprender". Se podría definir el método de la siguiente manera

```
void mensaje()
{
        System.out.println("Java es fácil de Aprender");
}
```

En este caso el método no devuelve un valor, por eso el tipo_de_valor_devuelto es void. Lo que hace solamente el método es mostrar un mensaje.

Ejercicios utilizando Métodos

1) Hacer un programa para calcular el mayor de 3 números reales.

```
import java.io.*;

class Metodos01
{
    public static float mayor(float a, float b, float c)
   {
          float may=a;
          if(b>may)
            may=b;
          if(c>may)
           may=c;;
          return c;
  }

  public static void main(String args[]) throws IOException
  {
     BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
     int a, b,c;

    System.out.print("Primer numero : ");
    a=Float.parseFloat(br.readLine());
   System.out.print("Segundo numero : ");
    b=Float.parseFloat(br.readLine());
   System.out.print("Tercer numero : ");
   c=Float.parseFloat(br.readLine());

    System.out.println("el mayor es  : "+mayor(a,b,c));
  }
}
```

2) Calcular el MCD de dos números enteros.

```
import java.io.*;

class Metodos02{
    public static int mcd(int a, int b)
   {
          int d=2,p=1;
          while(d<=a && d<=b)
            if(a%d==0 && b%d==0)
             {
                p=p*d;
                a=a/d;
                b=b/d;
             }
            else
                  d++;
           return p;
    }

    public static void main(String args[]) throws IOException
   {
      BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      int x,y,m;
     do{
           System.out.print("Ingrese primer numero :");
           x=Integer.parseInt(br.readLine());
     }while(x<0);

    do{
          System.out.print("Ingrese el segundo numero : ");
          y=Integer.parseInt(br.readLine());
    }while(y<0);
    m=mcd(x,y);
    System.out.println("El m.c.d de "+x+" y "+y+" es : "+m);
  }
}
```

3) Programa para reportar todos los divisores de un número entero N

```
import java.io.*;
class Metodos03{
  public static void reporteDivisores(int n)
  {
    int d;
    System.out.println("Los divisores del numero son :");
    for(d=1;d<=n;d++)
       if(n%d==0) System.out.print(d+" ");
    System.out.println();
  }
```

```
  public static void main(String args[]) throws IOException
  {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    int num;
    do{
              System.out.print("Ingrese numero :");
              num=Integer.parseInt(br.readLine());
    }while(num<=0);
    reporteDivisores(num);
  }
}

4) Programa reportar los factores primos de un número entero n
import java.io.*;
class Metodos04{
  public static void reporteFactoresPrimos(int n)
  {
     int d=2;
     while(n>1)
     {
       while(n % d !=0) d=d+1;
       System.out.print(d+" ");
        n=n/d;
      }
   }

   public static void main(String args[]) throws IOException
   {
      BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
     int num;
     do{
           System.out.print("Ingrese numero :");
             num=Integer.parseInt(br.readLine());
      }while(num<=0);
      reporteFactoresPrimos(num);

  }
}

// Programa para ingresar un numero y se reporte si es primo

import java.io.*;

class Metodos05{
  public static boolean esPrimo(int n)
  {
    int d=1;
    do{
                      d=d+1;
           }while( n%d!=0 && d*d<=n);
           if(d*d>n) return true;
```

```
        else return false;
 }

 public static void main(String args[]) throws IOException
 {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
  int num;
  do{
          System.out.print("Ingrese numero :");
          num=Integer.parseInt(br.readLine());
        }while(num<=0);
        if(esPrimo(num))
           System.out.println("El numero es primo");
        else
                System.out.println("El numero no es primo");
 }
}
```

Cuando una clase tiene métodos estáticos se puede invocar a esos métodos de la siguiente manera: NombreClase.nombreMetodo, como se hace con los métodos de la clase Math, para la raiz cuadrada se coloca Math.sqrt(x), para la potencia Math.pow(x,n).
Por ejemplo crearemos una clase llamada Aritmética que contenga métodos estaticos esPrimo (que verifica si un numero es primo), factorial, mcd(Calcula el m.c.d. de dos números) , reporteDivisores y reporteFactoresPrimos.
Estos métodos como son estaticos pueden ser llamados como :

Aritmética.nombre_metodo(parámetros)

Como se ve en el programa que sigue a la clase Aritmetica

```
class Aritmetica{
 public static boolean esPrimo(int n)
 {
  int d=1;
  do{
                d=d+1;
        }while(n%d!=0 && d*d<=n);
        if(d*d>n) return true;
        else return false;
 }

 public static int factorial(int n)
 {
        int f=1,i;
        for(i=1;i<=n;i++)
                f=f*i;
        return f;
 }

 public static int mcd(int a, int b)
 {
        int d=2,p=1;
```

```
        while(d<=a && d<=b)
        if(a%d==0 && b%d==0)
        {
           p=p*d;
          a=a/d;
          b=b/d;
        }
        else
             d++;
        return p;
}
  public static void reporteDivisores(int n)
  {
        int d;
        System.out.println("Los divisores del numero son :");
        for(d=1;d<=n;d++)
          if(n%d==0) System.out.print(d+" ");
        System.out.println();
  }

 public static void reporteFactoresPrimos(int n)
 {
        int d=2;
        while(n>1)
        {
          while(n % d !=0) d=d+1;
      System.out.print(d+" ");
      n=n/d;
        }
        System.out.println();
   }
}
```

Programa que usa la clase Aritmética

```
class PruebaAritmetica{
    public static void main(String args[]) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int n;
        do{
                System.out.print("Ingrese numero :");
                n=Integer.parseInt(br.readLine());
        }while(n<0);
        System.out.println("el Factorial es : "+Aritmetica.factorial(n));
        if(Aritmetica.esPrimo(n))
             System.out.println("El numero es primo");
        else
             System.out.println("El numero no es primo");
     }
}
```

Duración y alcance de un identificador

La ***duración*** de un identificador determina cuando dicho identificador existe en la memoria.
El ***alcance*** de un identificador es la porción del programa en la que se puede hacer referencia a dicho identificador.

Los identificadores que representan variables locales de un método (esto es, los parámetros y la variables declaradas en el cuerpo del método) tienen ***duración automática***. Las variables de duración automática se crean cuando se ingresa en el bloque en el que se declaran, existen mientras dicho bloque está activo y se destruyen cuando se sale del bloque.
Java también tiene métodos de ***duración estática***. Las variables y métodos de duración estática existen desde el momento en el que la clase en la que se definen se carga en la memoria para ejecutarse, hasta que el programa termina. El espacio de almacenamiento para las variables de duración estática se asigna e inicializa una vez cuando su clase se carga en la memoria. En el caso de los métodos de duración estática comienza a existir cuando su clase se carga en la memoria.

Los ***alcances*** para un identificador son alcance de clase y alcance de bloque.
Una variable de ejemplar declarada fuera de cualquier método tiene alcance de clase. El identificador correspondiente se "conoce" en todos los métodos de la clase.
Los identificadores declarados de un bloque tienen alcance de bloque. El alcance de bloque termina en la llave derecha que cierra el bloque.
Las variables locales declaradas al principio de un método tienen alcance de bloque, lo mismo que los parámetros del método, que se consideran variables locales en ese método.
Cualquier bloque puede contener declaración de variables.

Recursividad

Un método es recursivo cuando se llama a si mismo ya sea directamente o indirectamente.

Si un método recursivo se invoca con un caso base, simplemente devuelve un resultado. Si el método se invoca con un problema más complejo, divide el problema en dos o más partes conceptuales: una parte del método sabe como resolver y una versión un poco más pequeña del problema original. Dado que este nuevo problema se asemeja al problema original, el método emite una llamada recursiva para trabajar con el problema reducido.

Para que la recursividad termine, cada vez que el método recursivo se llama a si mismo con una versión un poco más sencilla del problema original, la secuencia de problemas cada vez menores debe convergir hacia el caso base. Cuando el método reconoce el caso base, devuelve el resultado a la llamada de método previa, y se inicia una secuencia de devoluciones que termina cuando la llamada del método original devuelve el resultado final.

Recursividad vs Iteración
Tanto la iteración como la recursión se basan en una estructura de control.: La iteración usa una estructura de repetición; la recursión una estructura de selección.

Tanto la iteración como la recursión implican repetición: la iteración emplea explícitamente una estructura de repetición; la recursión logra la repetición mediante llamadas de métodos repetidas

Tanto la iteración como la recursión incluyen una prueba de terminación; la iteración termina cuando deja de cumplirse la condición para continuar el ciclo; la recursión termina cuando se reconoce el caso base.

Ejercicios

1) Hacer un programa para calcular el factorial de un número.

```
import java.io.*;

class recursividad01{

    public static int factorial(int n)
    {
        if(n==0) return 1;
        else return n*factorial(n-1);
    }

    public static void main(String args[]) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      int num;
      do{
            System.out.print("Ingrese numero :");
            num=Integer.parseInt(br.readLine());
          }while(num<=0);
          System.out.println("El factorial es : "+factorial(num));
    }
}
```

2) Calcular la potencia de x elevado a la n en forma recursiva. x real y n entero positivo

```
// Programa para calcular el factorial de un numero

import java.io.*;

class recursividad02{

    public static double potencia(double x, double n)
    {
        if(n==0) return 1;
        else return x*potencia(x,n-1);
    }

    public static void main(String args[]) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int n;
        double x;
        System.out.print("Valor de x :");
        x= Double.parseDouble(br.readLine());
        do{
                        System.out.print("valor de n : ");
                        n=Integer.parseInt(br.readLine());
                }while(n<=0);
                System.out.println(x+" elevado a la "+n+" es igual a "+potencia(x,n));
    }
}
```

3) Hacer un programa para que reporte los n términos de la serie de Fibonacci

```
import java.io.*;

class recursividad03{

    public static int fibonacci(int n)
    {
        if(n==1) return 1;
        else
            if(n==2)
              return 1;
            else
                return fibonacci(n-1) + fibonacci(n-2);
    }

    public static void main(String args[]) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int n,i ;
        do{
                        System.out.print("Número de terminos de la serie : ");
                        n=Integer.parseInt(br.readLine());
                }while(n<=0);
                for(i=1;i<=n;i++)
                  System.out.print(fibonacci(i) + " ");
                System.out.println();
    }
}
```

4) Programa para calcular el máximo común divisor de dos números.

```
import java.io.*;

class recursividad04{

    public static int mcd(int a,int b)
    {
        if(a%b==0) return b;
        else return mcd(b,a%b);
    }

    public static void main(String args[]) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int x,y;
        do{
                    System.out.print("Ingrese primer numero :");
                    x=Integer.parseInt(br.readLine());
           }while(x<=0);
           do{
```

```
                        System.out.print("Ingrese segundo numero :");
                        y=Integer.parseInt(br.readLine());
            }while(y<=0);
            System.out.println("El mcd de "+x+" y "+y+" es : "+mcd(x,y));
    }
}
```

5) Programa para reportar un numero al reves

```
import java.io.*;
class recursividad05{

    public static void reves(int n)
    {
        System.out.print(n % 10);
        if( n/10 !=0 ) reves(n/10);

    }

    public static void main(String args[]) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int num;
        do{
            System.out.print("Ingrese numero :");
            num=Integer.parseInt(br.readLine());
          }while(num<=0);
          System.out.print("Numero al reves :");
          reves(num);
          System.out.println();
    }
}
```

6) Programa para convertir un numero de base 10 a base b (entre 2 y 9)

```
import java.io.*;
class recursividad06{

    public static void conversionBase(int n, int b)
    {
        if(n<b)
            System.out.print(n);
        else
            {
              conversionBase(n/b,b);
              System.out.print(n%b);
            }
    }
```

```
    public static void main(String args[]) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int num,b;
        do{
                System.out.print("Ingrese numero :");
                num=Integer.parseInt(br.readLine());
          }while(num<=0);
          do{
                System.out.print("Base a la que quiere convertir : ");
                b=Integer.parseInt(br.readLine());
          }while(b<2 || b>9);
          System.out.print("El numero "+num+" en base "+b+" es : ");
          conversionBase(num,b);
          System.out.println();
    }
}
```

Sobrecarga de Métodos

Java permite definir varios métodos con el mismo nombre en tanto dichos métodos tengan diferentes juegos de parámetros (con base en el número y el orden de los parámetros). Esto se denomina sobrecarga de métodos. Cuando se invoca un método sobrecargado, el compilador de Java selecciona el método adecuado examinando el número, los tipos y el orden de los argumentos en la llamada. La sobrecarga de métodos suele utilizarse para crear varios métodos con el mismo nombre que realizan tareas similares, pero sobre datos de diferentes tipos.

```
import java.io.*;
class sobrecarga01{
    public static int cuadrado(int x)
    {
        return x*x;
    }
    public static float cuadrado(float x)
    {
        return x*x;
     }
     public static double cuadrado(double x)
    {
         return x*x;
    }

    public static void main(String args[]) throws IOException
    {
        int a=3;
        float b=3.4f;
        double c=12.5;
        System.out.println("El cuadrado de "+a+ " es : "+cuadrado(a));
        System.out.println("El cuadrado de "+b+" es : "+cuadrado(b));
        System.out.println("El cuadrado de "+c+" es : "+cuadrado(c));
    }
}
```

ARREGLOS

Un arreglo es un grupo contiguo de posiciones de memoria relacionadas entre sí. Estas posiciones están relacionadas por el hecho de que todas tienen el mismo nombre y el mismo tipo. Para referirnos a una posición o elemento en particular dentro del arreglo, especificamos el nombre del arreglo y el subíndice del elemento.
Un subíndice puede ser un entero o una expresión entera. Si un programa emplea una expresión como subíndice, la expresión se evalúa para determinar el elemento específico del arreglo.
Los arreglos de Java siempre comienzan con el elemento 0.

Declaración de un arreglo
En java existen 2 formas de declarar arreglos.

Tipo de dato identificador[];
ó
Tipo de dato []identificador;

Por ejemplo si se desea declarar un arreglo de enteros, se podria hacer de esta manera:

int numeros[]; ó int []numeros;

Creación de un Arreglo
Después de declarar un arreglo, se tiene que crearlo. Para crearlo se coloca la palabra clave new seguida del tipo de dato y del tamaño del arreglo entre corchetes..

Ejm:

```
numeros = new int[100];  // se esta creando un arreglo de 100 elementos enteros
```

También se puede declarar y crear un arreglo en una sola instrucción. Por ejemplo:

```
double promedios[] = new double[50];
```

Se está declarando y creando 50 elementos de tipo double

Inicialización de un arreglo
En Java se pueden inicializar arreglos al declararlos. Cuando se especifica valores iniciales dentro de la declaración de un arreglo, Java realiza la operación new y define el tamaño del arreglo de forma automática.
Ejemplo:

```
int arreglo1 [] = { 1, 2, 3, 4, 5};

boolean arreglo[] = { true, false, true};

String meses[] = { "Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio"};
```

Acceso a los arreglos
Cada valor dentro de un arreglo se conoce como *elemento del arreglo*. Para acceder a un elemento de un arreglo especifique el nombre del arreglo con el índice del elemento entre corchetes [].
Ejemplo:

```
int numeros = { 12, 20, 60, 80, 100 };

for(int  indice = 0; indice < 5 ; indice++)
```

```
            System.out.println(arrego[indice]);
```

En este ejemplo el tamaño del arreglo es 5. En java el índice del primer elemento del arreglo es cero y la ultima posición es el tamaño del arreglo. Debido a esto, para iterar todos los elementos del arreglo, el programa utiliza loa valores 0 a 4 para la variable del ciclo.

Como obtener la longitud de un arreglo

En java un arreglo es un objeto. La única variable miembro de un objeto arreglo es la variable length (longitud), que contiene el tamaño del arreglo.
La variable length es de solo lectura, ya que el tamaño del arreglo no puede cambiar después de crearlo aquel.
Ejemplo : El siguiente código muestra como utilizar la variable length dentro de un ciclo for que itera por todos los elementos del arreglo.

```
int arreglo[] = {12, 20, 60, 80, 90};

for(indice = 0; indice<arreglo.length; indice++)
        System.out.println(arreglo[indice]);
```

Referencias a arreglos

Java utiliza referencias para apuntar a los arreglos.
Por ejemplo:
Las siguientes instrucciones utilizan las referencia *arreglo* para acceder a dos arreglos distintos.

```
import java.io.*;

class ReferenciaArreglo{

  public static void main(String args[])
  {
        int primero[] = {1, 2, 3, 4};
        int segundo[] = {5, 6, 7, 8, 9, 10};
        int arreglo[];
        arreglo=primero;
        System.out.println("Primer Arreglo ");

        for(int indice=0;indice<arreglo.length;indice++)
          System.out.println(arreglo[indice]);

        arreglo=segundo;
        System.out.println("Segundo Arreglo ");

        for(int indice=0;indice<arreglo.length;indice++)
          System.out.println(arreglo[indice]);
  }
}
```

Ejercicios

1) Hacer un programa para ingresar n valores reales en un arreglo y los muestre en la pantalla, además reportar el mayor, el menor y el promedio.

```
import java.io.*;
public class arre01{
```

```
  public static void main(String arg[])throws IOException
  {
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    double x[],mayor,menor,promedio,suma=0;
    int n,i;

    do{
       System.out.print("Cantidad de elementos del arreglo : ");

       n=Integer.parseInt(br.readLine());
    }while(n<=0 || n>100);

    x=new double[n];

    for(i=0; i<n;i++)
    {
      System.out.print("x["+i+"]: ");
      x[i]=Double.parseDouble(br.readLine());
    }

    System.out.println("Elementos del arreglo");
    for(i=0; i<n;i++)
      System.out.println("x["+i+"]: "+x[i]);

    // Calculo del mayor y menor

    mayor=menor=x[0];
    for(i=1; i<n; i++)
      if (x[i]>mayor) mayor=x[i];
      else
          if(x[i]<menor) menor=x[i];

    // Calculo de la suma de los elementos
    for(i=0; i<n; i++)
       suma=suma+x[i];

    promedio=suma/n;
    System.out.println("El mayor es " +mayor);
    System.out.println("El menor es:"+ menor);
    System.out.println("El promedio es : "+ promedio);
  }
}
```

2) Programa para ingresar n valores reales en un arreglo y calcular la desviación standard.

```
import java.io.*;

public class arre02{
  public static void main(String arg[])throws IOException
  {
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    double x[],suma=0,p,ds;
```

```
    int n,i;

    do{
       System.out.print("Cantidad de elementos del arreglo : ");
       n=Integer.parseInt(br.readLine());
            }while(n<=0);

    x=new double[n];

    for(i=0; i<n;i++)
    {
      System.out.print("x["+i+"]: ");
      x[i]=Double.parseDouble(br.readLine());
    }

    System.out.println("Elementos del arreglo");
    for(i=0; i<n;i++)
      System.out.println("x["+i+"]: "+x[i]);

    for(i=0; i<n; i++)
       suma=suma+x[i];
    p=suma/n;

    suma=0;
    for(i=0;i<n;i++)
      suma=suma + Math.pow(x[i]-p,2);
    ds=Math.sqrt(suma/(n-1));
    System.out.println("La desviacion standard es : "+ds);
  }
}
```

3) Programa para ingresar n valores reales en un arreglo y luego invierta el arreglo.

```
import java.io.*;

public class arre03{
  public static void main(String arg[])throws IOException
  {

    double x[],temp;

    int n,i,j;
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    do{
       System.out.print("Cantidad de elementos del arreglo : ");
       n=Integer.parseInt(br.readLine());
    }while(n<=0 || n>100);

    x=new double[n];
```

```
    for(i=0; i<n;i++)
    {
      System.out.print("x["+i+"]: ");
      x[i]=Double.parseDouble(br.readLine());
    }

    System.out.println("Arreglo Ingresado");
    for(i=0; i<n;i++)
       System.out.println("x["+i+"]: "+x[i]);

    for(i=0,j=n-1;i<n/2;i++,j--)
     {
       temp=x[i];
       x[i]=x[j];
       x[j]=temp;
     }
    System.out.println("Arreglo Invertido");
    for(i=0; i<n;i++)
       System.out.println("x["+i+"]: "+x[i]);

  }
}
```

4) Programa para ingresar 2 vectores de n elementos reales cada uno y reportar el producto escalar de ellos

```
import java.io.*;

public class arre04{
  public static void main(String arg[])throws IOException
  {
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    double x[],y[],pe;
    int n,i;

    do{
       System.out.print("Numero de elementos de los vectores : ");
       n=Integer.parseInt(br.readLine());
    }while(n<=0);

    x=new double[n];
    y=new double[n];

    System.out.println("Ingreso de datos del primer vector");
    for(i=0; i<n;i++)
    {
      System.out.print("x["+i+"]: ");
      x[i]=Double.parseDouble(br.readLine());
    }

    System.out.println("Ingreso de datos del segundo vector");
```

```
    for(i=0; i<n;i++)
    {
      System.out.print("y["+i+"]: ");
      y[i]=Double.parseDouble(br.readLine());
    }

    System.out.println("Elementos del primer vector");
    for(i=0; i<n;i++)
      System.out.println("x["+i+"]: "+x[i]);

    System.out.println("Elementos del Segundo vector");
    for(i=0; i<n;i++)
      System.out.println("y["+i+"]: "+y[i]);

    pe=0;
    for(i=0;i<n;i++)
      pe=pe+x[i]*y[i];
    System.out.println("El Producto escalar es : " +pe);
  }
}
```

5) Programa para ingresar n nombres en un arreglo y luego Ingresar un Nombre y buscar si este se encuentra en el arreglo ingresado.

```
import java.io.*;

public class arre05{
  public static void main(String arg[])throws IOException
  {
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    String nombres[],nombus;
    int n,i,pos;

    do{
       System.out.print("Cantidad de nombres a ingresar : ");
       n=Integer.parseInt(br.readLine());
    }while(n<=0);

    nombres=new String[n];

    for(i=0; i<n;i++)
    {
      System.out.print("Nombre["+i+"]: ");
      nombres[i]=br.readLine();
    }

    System.out.println("Elementos del arreglo");
    for(i=0; i<n;i++)
      System.out.println("Nombre["+i+"]: "+nombres[i]);
    System.out.print("nombres a buscar : ");
    nombus=br.readLine();
```

```
    pos=-1;
    for(i=0;i<n ; i++)
      if(nombres[i].compareToIgnoreCase(nombus)==0)
           {
         pos=i;
         break;
                    }
    if(pos!=-1)
      System.out.println("Nombre se encuentra en la posición"+pos);
    else
       System.out.println("Nombre no se encuentra en el arreglo");
  }
}
```

6) Programa para ingresar n nombres en un arreglo y luego reportarlos en orden alfabético.

```
import java.io.*;
public class arre06{
  public static void main(String arg[])throws IOException
  {
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    String nombres[],temp;
    int n,i,j;
    do{
       System.out.print("Cantidad de nombres a ingresar : ");
       n=Integer.parseInt(br.readLine());
    }while(n<=0);
    nombres=new String[n];
    for(i=0; i<n;i++)
    {
      System.out.print("Nombre["+i+"]: ");
      nombres[i]=br.readLine();
    }

    System.out.println("nombres ingresados");
    for(i=0; i<n;i++)
      System.out.println("Nombre["+i+"]: "+nombres[i]);

    for(i=0;i<n-1;i++)
      for(j=i+1;j<n;j++)
        if(nombres[i].compareToIgnoreCase(nombres[j])>0)
         {
           temp=nombres[i];
           nombres[i]=nombres[j];
           nombres[j]=temp;
         }

    System.out.println("nombres en Orden Alfabetico");
    for(i=0; i<n;i++)
      System.out.println("Nombre["+i+"]: "+nombres[i]);
}
```

7) Programa para ingresar n numeros en un arreglo y luego ingresar un numero y si este se encuentra eliminarlo en caso contrario reportar dato no se encuentra

```
import java.io.*;

public class arre07{
  public static void main(String arg[]) throws IOException
  {
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

    double x[],num;
    int n,i,p;
    do{
       System.out.print("Cantidad de elementos del arreglo : ");
       n=Integer.parseInt(br.readLine());
    }while(n<=0);

    x=new double[n];

    for(i=0; i<n;i++)
    {
      System.out.print("x["+i+"]: ");

     x[i]=Double.parseDouble(br.readLine());
    }

    System.out.println("Datos ingresados ");
    for(i=0; i<n;i++)
         System.out.println("x["+i+"]: "+x[i]);
    System.out.println("Numero a buscar : ");
    num=Double.parseDouble(br.readLine());
    p=-1;
    for(i=0;i<n ;i++)
     if(x[i]==num)
     {
       p=i;
       break;
            }

    if(p!=-1)
     {
       for(i=p;i<n-1;i++)
         x[i]=x[i+1];
       n=n-1;
       System.out.println("Nuevo arreglo ");
       for(i=0; i<n;i++)
         System.out.println("x["+i+"]: "+x[i]);
     }
    else
        System.out.println("El numero no se encuentra en el arreglo");
}
```

8) Programa para ingresar n puntos del plano cartesiano y reportar la ecuacion de la recta de regresion

```
import java.io.*;

public class arre08{
  public static void main(String arg[])throws IOException
  {
     BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
     double x[],y[],sx=0,sy=0,sxy=0,sxx=0,M,B;

     int n,i;

     do{
        System.out.print("Cantidad de Puntos : ");
        n=Integer.parseInt(br.readLine());
     }while(n<=0);

     x=new double[n];
     y=new double[n];

     System.out.println("Ingreso de datos");
     for(i=0; i<n;i++)
     {
       System.out.print("x["+i+"]: ");
       x[i]=Double.parseDouble(br.readLine());
       System.out.print("y["+i+"]: ");
       y[i]=Double.parseDouble(br.readLine());
     }

     for(i=0;i<n;i++)
      {
        sx=sx+x[i];
        sy=sy+y[i];
        sxx=sxx+x[i]*x[i];
        sxy=sxy+x[i]*y[i];
      }

      M = (n*sxy-sx*sy)/(n*sxx -sx*sx);
      B = (sxx*sy -sxy*sx)/(n*sxx -sx*sx);

      System.out.println("La ecuacion de la recta es : ");
      System.out.println("Y = "+M+"X + "+B);
   }
}
```

9) Ingresar los nombres y las notas de n alumnos y reportar la lista en orden alfabetico y en orden de merito

```
import java.io.*;

public class arre09{
```

```
public static void main(String arg[])throws IOException
 {
  BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
  String nombres[],tempNombre;
  int n,i,j,notas[],tempNota;
  do{
     System.out.print("Numero de Alumnos : ");
     n=Integer.parseInt(br.readLine());
  }while(n<=0);
  nombres=new String[n];
  notas=new int[n];
  for(i=0; i<n;i++)
  {
    System.out.print("Nombre["+i+"]: ");
    nombres[i]=br.readLine();
    System.out.print("Nota["+i+"]:");
    notas[i]=Integer.parseInt(br.readLine());
  }

  for(i=0;i<n-1;i++)
    for(j=i+1;j<n;j++)
      if(nombres[i].compareToIgnoreCase(nombres[j])>0)
        {
          tempNombre=nombres[i];
          nombres[i]=nombres[j];
          nombres[j]=tempNombre;
          tempNota=notas[i];
          notas[i]=notas[j];
          notas[j]=tempNota;

        }
  System.out.println("Lista en Orden Alfabetico");
  for(i=0; i<n;i++)
    System.out.println(nombres[i]+"    "+notas[i]);
  for(i=0;i<n-1;i++)
    for(j=i+1;j<n;j++)
      if(notas[i]<notas[j])
        {
          tempNombre=nombres[i];
          nombres[i]=nombres[j];
          nombres[j]=tempNombre;
          tempNota=notas[i];
          notas[i]=notas[j];
          notas[j]=tempNota;
        }
  System.out.println("Lista en Orden de Merito");
  for(i=0; i<n;i++)
    System.out.println(nombres[i]+"    "+notas[i]);

 }
```