PROGRAMACION MODULAR

1. Arreglos

Un arreglo es un conjunto de informaciones organizadas secuencialmente y localizables por subíndices o sub-numero de orden.

2. Vectores o arreglos unidimensionales

Los arreglos unidimensionales o vectores permiten representar muchos datos con un solo nombre de variable el nombre se hace extensivo a todos los componentes del vector, se ubica a cada uno de ellos mediante un sub índice.

- Representación grafica

V =	5	8	24	....	49
i =	1	2	3	....	N

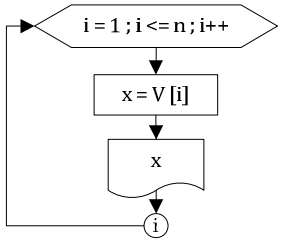
Sub índices = 1, 2, 3, 4,..., n = i, j, k,..etc.
V = Nombre del vector
V[i]= Elementos del vector = 5, 8, 24, ..., 49

V [1] = 5
V [2] = 8
V [3] = 24
V [...] = ...
V [n] = 49

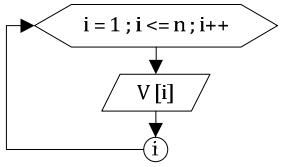
- Definición de un vector
  Un vector debe de tener elementos de un solo tipo de datos. (Deben de ser Homogéneos)

En diagrama de flujo	En java
V[50] TAMAÑO DE VECTOR	**int V[] = new int [50];**

- Recorrido de un vector

En diagrama de flujo	En java
	**for (** i = 1; i <= n; i++) **{** x = V[i] System.out.print(x); **}**

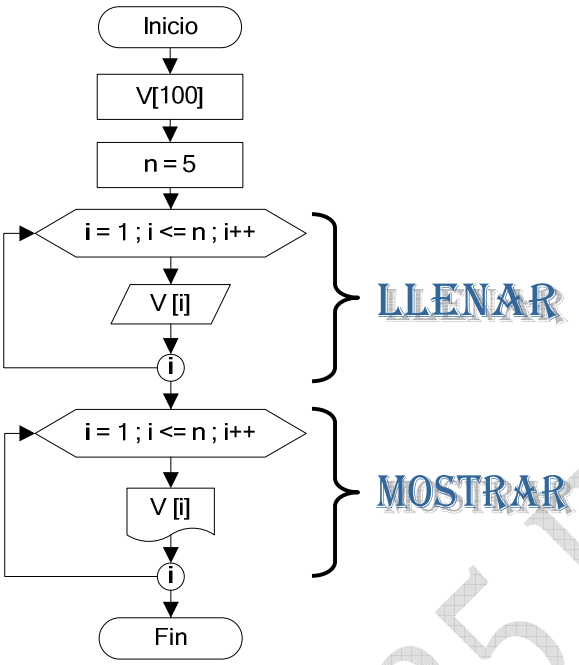
- Leer o llenar elementos

En diagrama de flujo	En java
	**for (** i = 1; i <= n; i++) **{** System.out.print ("V["+i+"]"); V[i] = Leer.datoInt (); **}**

Ejemplo 1)

Llenar de elementos enteros a un vector V de tamaño n=5, luego mostrar todos sus elementos.

En diagrama de flujo	En java

	**class** Ejemplo_1
	{
	public static void main (String [] EVHA)
	{
	int i, n=5;
	int v [] = **new int** [50];
	for (i = 1 ; i <= n ; i++)
	{
	System.out.print ("V["+i+"]");
	V[i] = Leer.datoInt ();
	}
	for (i = 1 ; i <= n ; i++)
	{
	System.out.print ("[" + v [i] + "]" + ",");
	}
	}
	}

Matrices o Arrays (bidimensionales)

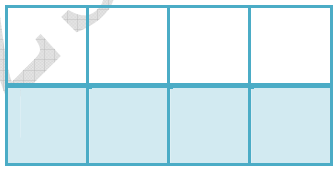
Una matriz es una estructura de datos estáticos que permiten manejar dos índices (F,C).

M =

	c1	c2	c3
f1	0 11	1 12	3 13
f2	100 21	20 22	-1 23
f3	-2 31	0 32	3 33

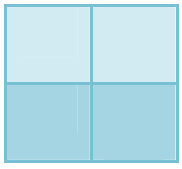
1. **Matriz rectangular**

   Se caracteriza por tener el numero de filas diferentes al número de columnas (F<>C).



2. **Matriz cuadrada**

   Se caracteriza porque el número de filas es igual al número de columnas (F=C). **(El tamaño se lo puede escribir en una sola variable (n))**



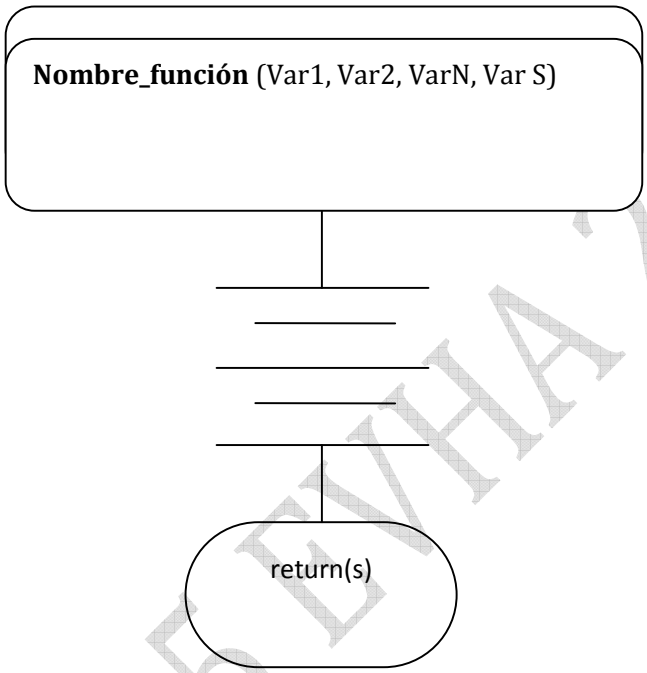
Codificación en java

```
int m[][]=new int[30] [30]
```

FUNCIONES

Concepto.- Una función es un sub programa que realiza una tarea específica y puede tener uno o varios parámetros de entrada y una única salida.

Partes de una función



Codificación Para La Función

```
public static int nombre_funcion(int
Var1, int Var2)
{
int c;
:::::::::::::::::::::::::::
return(s);
}
```

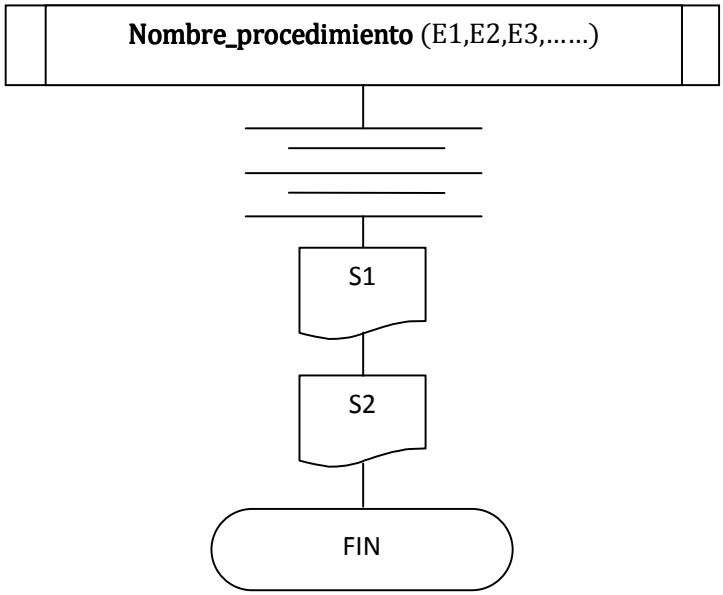
Codificación Para El Programa Principal

```
public static void main(String args[])
{
int n,r;
:::::::::::::::::::::::::
r=nombre_funcion(Var1, Var2)
System.out.print(r);
}
```

PROCEDIMIENTOS

Son un conjunto de instrucciones que resuelven un problema y tienen uno o más entrada y tienen 0,1 o más salidas.

Un Procedimiento o subrutina es un sub programa que ejecuta un suceso especifico.



Codificación Para El Procedimiento

```
public static void
nombre_procedimiento(int s1, int s2)
{
int c;
::::::::::::::::::::::::::::
System.out.print(s);
}
```

Codificación Para El Programa Principal

```
public static void main(String args[])
{
int n,r;
::::::::::::::::::::::::
r=nombre_procedimiento(s1, s2)
System.out.print(r);
}
```

MANEJO DE CADENAS

Las cadenas de caracteres un java son objetos de la clase (String), java crea de forma automática un objeto String con el valor literal.

Métodos de la clase String:

String Cadena

String Frase

Char r

Int s

Int length ()

Este método devuelve la longitud o numero de caracteres de la cadena del objeto String.

CharAt (i)

Devuelve el contenido de la posición en (i) de la cadena.

COMPARACION DE CADENAS

- equals() devuelve booleano (true,false)
- compareTo() devuelve enteto (0 si son iguales)
- compareToIgnoreCase() ignora MAYUS o MINUS (devuelve true o false)

CONVERTIR A MAYUS

- ToUpperCase()

CONVERTIR A MINUSCULAS

- ToLowerCase()

PARA ELIMINAR ESPACIOS VACIOS

- Trim() del primero y final

PARA OBTENER UNA PARTE DE LA CADENA USAR

- Substring()

  Ejemplos:

  cad=”Bolivia Amada”;

  PARA MAYUSCULAS

```
cad=cad.ToUpperCase();
System.out.println(cad); // MAYUS
```

  PARA MINUSCULAS

```
cad=cad.ToLowerCase();
System.out.println(cad); // MINUS
```

  ELIMINA ESPACIO VACIO EN INICIO Y FINAL

```
cad=cad.Trim();
System.out.println(cad); // SIN ESPACIOS VACIOS (in,fin)
```

  PARA VER DESDE Q POSICION QUIERES

```
String cad = new String ();
String c = new String ();
cad = "      Bolivia      ";
System.out.print ("" + cad);
c = cad.substring (8); /*PARA VER DESDE QUE POSICION QUIERES VER LA CADENA*/
System.out.print ("AHORA MUESTRA DESDE CIERTA POSICION --->" + c);
```

  PARA VER DESDE QUE POSICION QUIERES VER PERO 1er Pa. >= 2do

```
String cad = new String ();
String c = new String ();
```

```
cad = "Bolivia Amada";
System.out.print ("" + cad);
c = cad.substring (5, 8); // el 2do PARAMETRO DEBE SER > o = AL PRIMER PARAMETRO //
System.out.print ("AHORA MUESTRA DESDE CIERTA POSICION --->" + c);
```

PARA VER REEMPLAZAR UN CARÁCTER

```
String cad = new String ();
String c = new String ();
cad = "Bolivia Amada";
System.out.print ("" + cad);
c = cad.replace ('a','-'); //REEMPLAZA POR UN CARCTER//
System.out.print ("AHORA MUESTRA DESDE CIERTA POSICION --->" + c);
```

To String (convierte en un String)

El dato a convertir tiene que ser un objeto

Que es un objeto

Ejemplo:

```
Int a; // no es un objeto
```

```
Integer a = new Integer(0); // Es un objeto
```

otro

```
Integer a = new Integer(123); // Es un objeto
```

ARCHIVOS EN JAVA

Si bien es cierto que ya se pueden manejar gran cantidad de datos del mismo y de diferentes tipo s de datos al mismo tiempo y que estos pueden ser almacenados en forma permanente normalmente en los dispositivos de almacenamiento estándar.

Como nota a tomar en cuenta los datos que se van almacenando en un archivo de disco se almacenan en renglones consecutivos y cada renglón en disco se conoce como **registro** del archivo favor de no confundir el concepto de registro de archivo y registro o estructura como variable ya analizada son dos cosas totalmente diferentes aunque se llamen igual.

Operaciones básicas con archivos:

A. **ESCRIBIR O GRABAR**: Es la operación mas elemental con un archivo consiste en tomar un o unos datos en variables de cualquier tipo (escalar, mezcla de datos, arreglo, structuras) y almacenarlas en un archivo de datos en disco.
B. **LEER:** Operación consistente en sacar los datos del archivo en disco y mandarlo o cargar la variable respectiva

Organización de archivos:

En general existen dos tipos de archivos:

A. **Archivos Secuenciales**.- En este caso los datos se almacenan en forma consecutiva y no es posible leer ningún registro directamente es decir para leer el registro n, se deberá recorrer o accesar los n-1 registros anteriores.
B. **Archivos Directos o Random**.- Para este caso si se puede acceder o leer un renglón n cualquiera.

Tipo de archivos:

A) En general, existen tantos tipos de archivos como tipos de datos existen es decir existen archivos de bytes, de chars, de ints, de floats, etc.

Nota: Ya que se decide utilizar algún archivo especifico de datos (caracteres, strings, formateados, registros o arreglos) solo utilizar las funciones de escritura y lectura de ese tipo de archivo por ningún motivo mezcle funciones de lectura y escritura de otro tipo de archivos.

Almacenamiento en arhivos:

A. Modo Texto: en este caso los datos son almacenados usando codigo ascii y por tanto son plenamente visibles usando cualquier editor.
B. Modo Binario: en este caso los datos son almacenados en notación hexadecimal y por tanto se ocupa un editor binario para reconocerlos sin embargo un archivo binario es mas compacto que un archivo texto.

ARCHIVOS SECUENCIALES EN JAVA

Existen además muchas otras operaciones asociadas a archivos las mas elementales son:

1.- **Creación de Archivo**.- En este proceso se pretende solamente crear un archivo nuevo en disco, con su nombre, tipo y especialidad de almacenamiento de datos apropiado.

2.- **Apertura de Archivos**.- En este caso se pretende abrir un archivo ya existente en disco para procesarlo, ya sea cargar o grabar datos en sus registros, o leer algún registro en especial para mandarlo a una variable de cualquier tipo.

No confundir creación con apertura creación es un proceso que solo se ejecuta una sola vez en la vida de un archivo, mientras que apertura siempre se esta realizando por los programas especializados en algún proceso.

3.-**Cierre de archivos**.- Es la operación mas importante en cualquier programa que maneje archivos o se cierra el archivo como ultima instrucción del programa o se vera el anuncio ABORT,RETRY,FAIL.

4.-**Altas en archivo**.- En este proceso se carga una clase en memoria con sus datos pertinentes y se graba la clase en el archivo en disco.

5.-**Lectura de archivo**.- En este proceso, se abre el archivo y se manda el registro de disco, a una clase en memoria para su procesamiento.

6.- **Consulta de archivos**.- En este proceso se pretende desplegar todos los registros del archivo en disco a la pantalla ya sea consola o mejor aún, a una pagina html

7.-**Busqueda en archivos**.- Una de las operaciones mas comunes, consiste en que el usuario pide toda la información de algún renglón en disco, proporcionando la información de algún campo, generalmente el campo clave de la clase.

8.- **Filtros**.- En este proceso el usuario esta interesado en algún conjunto de renglones con características comunes (condición), por ejemplo todos los alumnos de "sistemas", o todos los empleados que ganen mas de $500.00 pesos, o todos los clientes que sean de "Vasco", etc

9.-**Modificaciones de registros o archivos**.- Problema muy común, donde los datos originales ya grabados se tienen que cambiar o actualizar, por ejemplo el nombre no era "juan" es "juana", o la calificación no es 100 es 20, etc.

10.- **Bajas de registros**.- También muy común este proceso por ejemplo el alumno ya egreso, el cliente huyo, etc.

GRABACION ARCHIVO SECUENCIAL EN JAVA

Grabación y lectura son los dos procesos mas comunes con archivos en cualquier lenguaje de programación.

Código fuente de grabación:

Programa.java

```
import java.lang.*;
import java.io.*;
class prog22
{
  public static void main(String[] args)
 {
        // crear un objeto de tipo archivo
        DataOutputStream archivo = null;
        // creando e inicializando los campos del registro
        // observar que se debe usar clases numericas apropiadas
        int clave=0;
        String nombre=new String("");
        int edad=0;
        // creando objeto teclado
        BufferedReader teclado = new BufferedReader(new
        InputStreamReader(System.in));
        // abriendo archivo, capturando y grabando datos
        try
        {
```

```
//* Creando y grabando a un archivo, esta larga la instrucción*/
archivo = new DataOutputStream( new
FileOutputStream("c:\\pfacil\\archivo1.dat",true) );
System.out.println("dame clave: ");
clave = Integer.parseInt(teclado.readLine());
System.out.println("dame nombre: ");
nombre=teclado.readLine();
System.out.println("dame edad: ");
edad = Integer.parseInt(teclado.readLine());
//grabando al archivo
archivo.writeInt(clave);
archivo.writeUTF(nombre);
archivo.writeInt(edad);
archivo.close();
}

catch(FileNotFoundException fnfe) { /* Archivo no encontrado */ }
catch (IOException ioe) { /* Error al escribir */ }
} // cierra main
} // cierra clase
```

Probar la corrida del programa

Se usa una clase llamada FileOutputStream, especializada en archivos con muchos metodos y constructores para crear, manipular y procesar archivos el constructor usado solo lleva dos parametros, el primero todo la ruta o path a donde quedara el archivo(cuidado con no poner la doble diagonal \\) y el segundo parametro es

la palabra "true", esto es para que el archivo se abra en modo llamado "APPEND", es decir que cada nuevo registro se vaya escribiendo al final del archivo, si no se pone este parametro(true), un nuevo registro se sobreescribiria sobre el registro anterior.

Sin embargo en el programa no se uso solo FILEOUTPUSTREAM (solo para crear el archivo), también se usa DATAOUTPUTSTREAM, esta segunda clase es derivada de la anterior y comparte muchos de sus métodos, la diferencia es que fileoutputstream esta especializada en grabar y leer bytes, mientras que dataoutputstream esta especializada en grabar y leer datos formateados, observar que los métodos que uso el objeto archivo para grabar o almacenar datos se especializan en algún tipo de dato en especial, sus otros métodos son:

Method Summary	
void	flush() Flushes this data output stream.
int	size() Returns the current value of the counter written, the number of bytes written to this data output stream so far.
void	write(byte[] b, int off, int len) Writes len bytes from the specified byte array starting at offset off to the underlying output stream.
void	write(int b) Writes the specified byte (the low eight bits of the argument b) to the underlying output stream.
void	writeBoolean(boolean v) Writes a boolean to the underlying output stream as a 1-byte value.
void	writeByte(int v) Writes out a byte to the underlying output stream as a 1-byte value.
void	WriteBytes (String s) Writes out the string to the underlying output stream as a sequence of bytes.
void	writeChar(int v) Writes a char to the underlying output stream as a 2-

	byte value, high byte first.
void	WriteChars (String s) writes a string to the underlying output stream as a sequence of characters.
void	writeDouble(double v) Converts the double argument to a long using the doubleToLongBits method in class Double, and then writes that long value to the underlying output stream as an 8-byte quantity, high byte first.
void	writeFloat(float v) Converts the float argument to an int using the floatToIntBits method in class Float, and then writes that int value to the underlying output stream as a 4-byte quantity, high byte first.
void	writeInt(int v) Writes an int to the underlying output stream as four bytes, high byte first.
void	writeLong(long v) Writes a long to the underlying output stream as eight bytes, high byte first.
void	writeShort(int v) Writes a short to the underlying output stream as two bytes, high byte first.
void	writeUTF (String str) writes a string to the underlying output stream using UTF-8 encoding in a machine-independent manner.

TABLA TOMADA DEL API DE JAVA

Observar que la grabación lleva un try-catch filenotfound y ioexception, que son obligatorios o no compila el programa.

No olvidar cerrar el archivo, con la instrucción archivo.close