MANUAL DE NETBEANS

ISBN: 978-958-59173-5-4

Instituto Tecnológico De Soledad Atlántico ITSA (958-57393)

LOURDES DE ÁVILA

Contenido

LISTA DE TABLAS 4
1. INTRODUCCIÓN A LOS ALGORITMOS 5
1.1 FUNDAMENTOS DE ALGORITMOS 5
1.2. CREACIÓN DE PROYECTOS 6
1.3 CREACIÓN DE CLASES 7
1.4 COMPILAR Y EJECUTAR UN PROYECTO 9
2. ESTRUCTURAS BÁSICAS DE ALGORITMOS 10
2.1. INSTRUCCIONES 10
2.2 OPERADORES RELACIONALES Y MATEMÁTICOS 12
2.3 ETAPAS PARA LA SOLUCIÓN DE PROBLEMAS POR COMPUTADOR 15
2.4. ESTRUCTURAS ALGORITMICAS 16
2.5 ESTRUCTURAS REPETITIVAS 36
BIBLIOGRAFÍA 44

LISTA DE IMÁGENES

Imagen 1. Ventana de Inicio en Netbeans. 6
Imagen 2. Ventana para selección de tipo de aplicación. 6
Imagen 3. Ventana para nombrar proyecto. 7
Imagen 4. Ventana Creación de la clase. 7
Imagen 5. Ventana para dar nombre a la clase. 8

LISTA DE TABLAS

Tabla 1. Ejemplos de Nombres de Variables. 11
Tabla 2. Precedencia de Operadores 12
Tabla 3. Operadores Relacionales 14
Tabla 4. Ejemplo 1 19
Tabla 5. Operador Y 26
Tabla 6. Operador O 27
Tabla 7. Operador NO 27
Tabla 8. Precios Vs Modelo 35
Tabla 9. Precio Vs Categoría 35
Tabla 10. Estructura Repetitiva 42
Tabla 11. Intentos Vs Categoría 43

1. INTRODUCCIÓN A LOS ALGORITMOS

1.1 FUNDAMENTOS DE ALGORITMOS

1.1.1 Algoritmo

Es un conjunto de pasos lógicos que permite alcanzar la solución de un problema.

- **Características**

  - **Finito:** Debe culminar la solución.
  - **Ordenado:** Debe tener una secuencia para seguir los pasos secuencialmente.
  - **No Ambigüo:** Debe ser claro y no tener diferentes interpretaciones.

- **Ejemplos de la Vida Diaria**
  Aplicando la definición de algoritmos a las actividades de la vida diaria se muestran los siguientes ejemplos:

  - Conseguir un vaso de agua.

    Ir a la nevera llevando el vaso en la mano.
    Abrir la nevera.
    Tomar la jarra o vasija con el agua.
    Verter el agua en el vaso hasta que esté lleno.
    Regresar la vasija a la nevera.
    Cerrar la nevera.
    Tomar el agua contenida en el vaso.
    Regresar el vaso.

  - Preparar una limonada.

    Partir cuatro limones en cuatro pedazos cada uno.
    Agregar los limones cortados, 1 litro de agua y 4 cucharadas de azúcar en la licuadora.
    Licuar los ingredientes por 4 minutos
    Colar la mezcla y verterla en la jarra
    Servir la mezcla colada en 4 vasos.

1.1.2 Ingreso al Programa

En el escritorio aparece un icono que lleva por nombre Netbeans.

Dar doble click en el icono y se iniciará el programa o ir al menú inicio, escoger la opción todos los programas y escoger Netbeans y luego dar click en Netbeans

1.2. CREACIÓN DE PROYECTOS

- Una vez ingresado al programa de ir al menú **File** y escoja la opción **New Project**

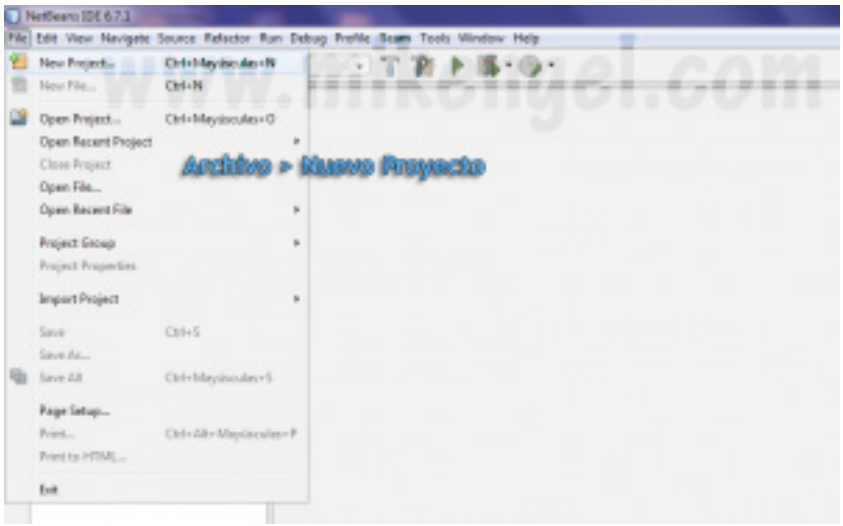


Imagen 1. Ventana de Inicio en Netbeans.

- Luego escoja la opción **General** en la ventana de la izquierda y en la derecha escoja la opción **Java Aplication** y presione el botón **Next**.

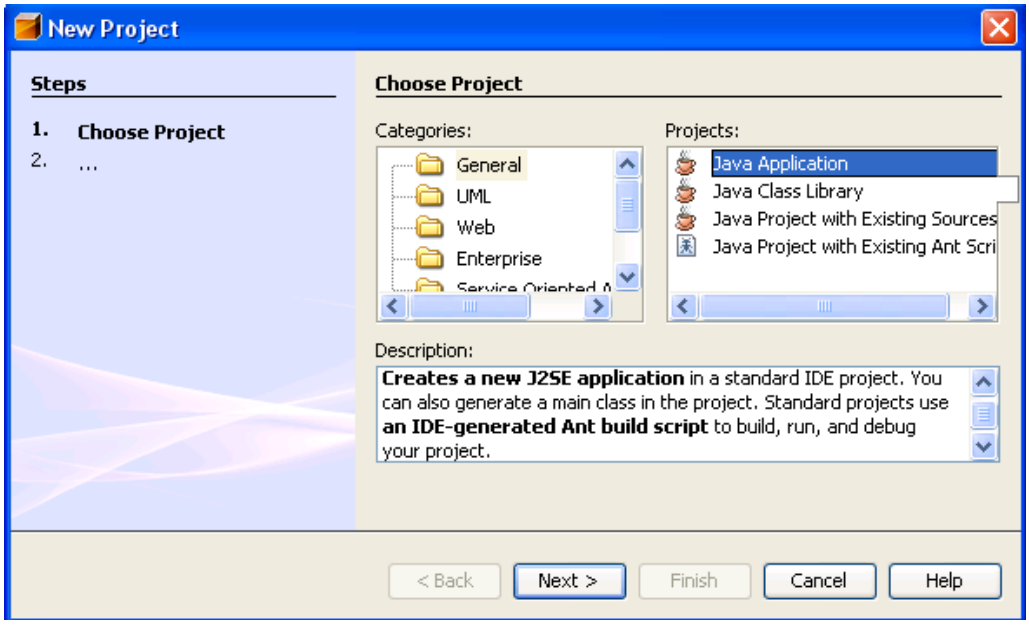


Imagen 2. Ventana para selección de tipo de aplicación.

- Le aparece una ventana, donde aparece **Project Name** en este lugar se coloca el nombre del proyecto a realizar, para el cual debe ser relacionado al objetivo del programa.

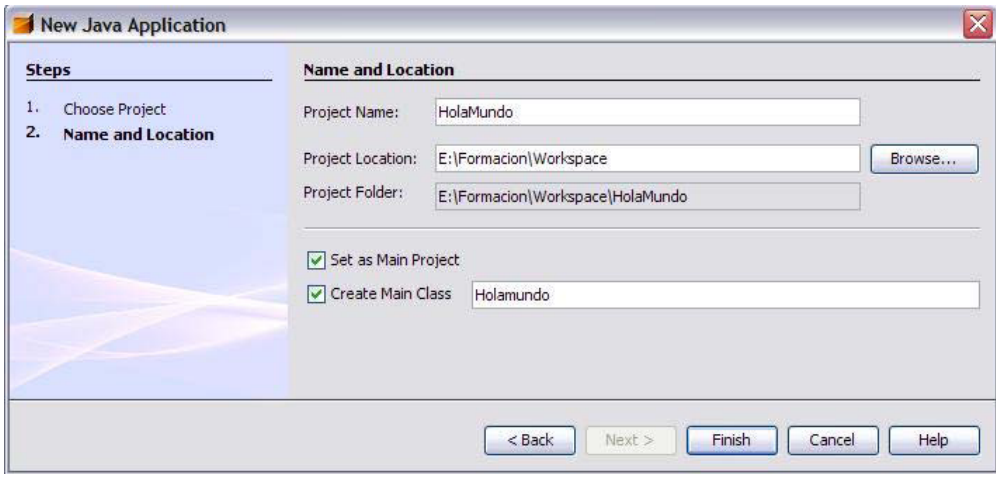


Imagen 3. Ventana para nombrar proyecto.

- Escoja el lugar donde se creará la carpeta en donde se guardará el programa. Ejemplo: Mi PC/C. Se coloca en **Project Location**.

- Desactive las dos casillas de verificación que aparecen con un chulito en color verde, se desactiva haciendo click sobre cada una de ellas, una vez desactivada se presiona el botón **Finish**

1.3 CREACIÓN DE CLASES

Una vez creado el proyecto se crea el espacio donde se escribirá el algoritmo, de la siguiente forma:

- ✓ Encima del nombre del proyecto que aparece en la parte izquierda de la ventana, se da click derecho, y en el menú que se despliega, se escoge la opción **New**, y luego la opción **Java Class**.

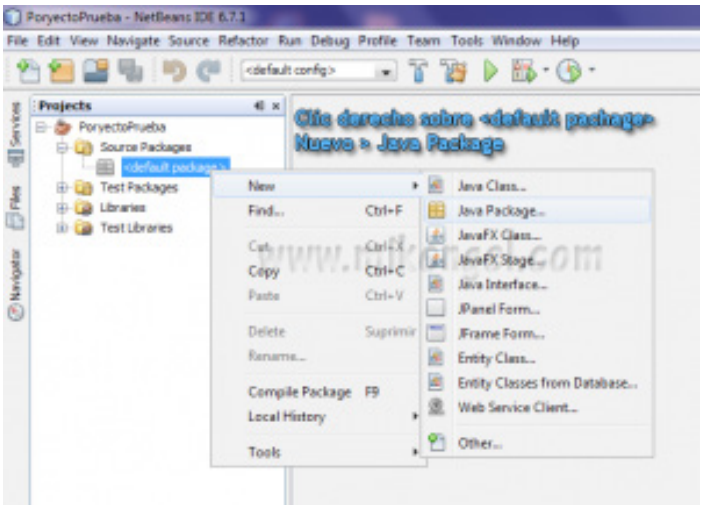


Imagen 4. Ventana Creación de la clase.

- ✓ En la ventana que se obtiene, en la parte en donde se encuentra **Class Name**, se coloca el nombre del programa.

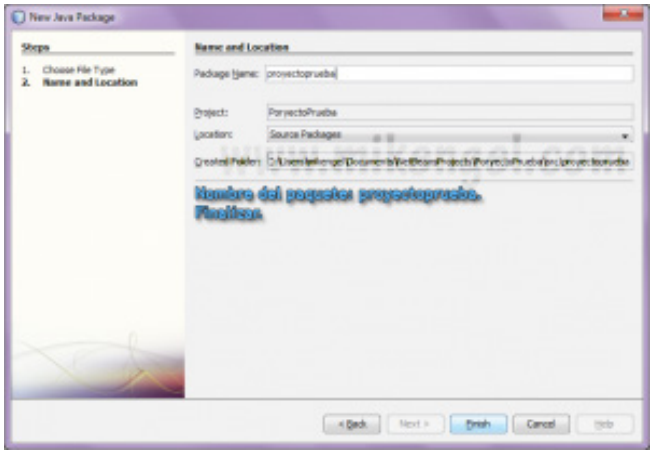


Imagen 5. Ventana para dar nombre a la clase.

- ✓ Presiona el botón **Finísh.**

Se culmina con la creación del programa el cual se crea con las siguientes instrucciones:

```
/*
 * prueba.java
 *
 * Created on 11 de julio de 2007, 12:45
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */

/**
 *
 * @author estudiante
 */
public class prueba {

    /** Creates a new instance of prueba */
    public prueba() {
    }

}
```

Esto quiere decir que se creó un programa, en la fecha y hora indicada. Además note que aparece una instrucción que es:

public class prueba {

Donde esta instrucción es la que indica que es un programa de Java y no se debe modificar o cambiar de ninguna forma.

1.4 COMPILAR Y EJECUTAR UN PROYECTO

Una vez el archivo grabado, se puede lanzar la compilación y la ejecución de su proyecto por Menú Run->RunMainProject (F6) o haciendo clic derecho sobre el programa y escoger la opción ejecutar archivo. Eso tiene por consecuencia ejecutar las diferentes instrucciones del programa para compilar, luego ejecutar la aplicación.

2. ESTRUCTURAS BÁSICAS DE ALGORITMOS

En la siguiente instrucción que es:

public prueba() {

Esta instrucción es modificada y reemplazada, para colocar el inicio del pseudocódigo de los algoritmos, de la siguiente forma:

public static void main(String args[]){
Luego aparece lo siguiente:
}

Este indica el fin del algoritmo, por lo tanto para escribir el algoritmo debe presionar la tecla **Enter** al final de la instrucción de **inicio** para que cree un espacio entre las dos líneas de código.

2.1. INSTRUCCIONES

2.1.1 Librería más Utilizadas

Librería es donde se definen operaciones que realiza el sistema, para los cual es necesario ser llamadas para poder utilizar dichas funciones, entre las cuales se tienen las siguientes:

import javax.swing.*; El cual permite enviar resultados en pantalla y leer datos por teclados entre otras operaciones.

Import java.util.*; El cual permite generar números aleatorios, manejo de fechas, colecciones y otras operaciones.

2.1.2 Declaración De Variables Y Tipos De Datos

Las **variables** son valores que pueden o no cambiar en el tiempo. Las características de una variable es que debe tener dos componentes: Nombre y tipo de dato.

Reglas para dar Nombre a Variables

- Debe iniciar con una letra.
- Puede contener después de la letra se puede colocar letras o números.
- No debe contener espacios en blanco.
- No debe contener operadores matemáticos como (+, -, *, / , etc.).

Tabla 1. Ejemplos de Nombres de Variables.

Nombre Variable	Estado (Correcto o Incorrecto)
Casa	Correcto
Casa verde	Incorrecto (Utiliza espacio en blanco)
A2	Correcto
A2B	Correcto
1b3	Incorrecto(Empieza con número)
N#	Correcto
N+d	Incorrecto(Utiliza Operadores matemáticos)

2.1.3 Ejercicios Propuestos

- Indique si los nombres de variables son correctos o incorrectos y justifique su respuesta.

a. A3456g
b. 9ft1
c. CA_45
d. a1
e. op2c_2

- ¿En cuáles de los siguientes pares es importante el orden de los enunciados? Es decir, si se modifica el orden ¿cambian los resultados finales? (Suponemos que X≠Y≠Z).

a) X=Y
Y=Z

b) X=Z
X=Y

c) X=Y
Z=X

d) Z=Y
X=Y

Manejo de Tipos de Datos en Netbeans

Existen varios tipos de datos como son: Numéricos y alfanuméricos.

- **Datos Numéricos**. Se dividen en enteros y reales, de los cuales en java se definen así:

a. Enteros: int variable1, variable2, variable3,…, variable n;

b. Reales: float variable1, variable2, variable3,…, variable n;
double variable1, variable2, variable3,…, variable n;

- **Datos Alfanuméricos**. Se dividen en carácter y cadenas, de los cuales en java se definen así:

a. Carácter: char variable1, variable2, variable3,…, variable n;
b. Cadenas: String variable;

2.1.4 ESTRUCTURA DE DATOS ESTATICAS. Son estructuras fijas entre los cuales tenemos los arreglos unidimensionales y bidimensionales.

Arreglos unidimensionales o Vectores. Se definen en java de la siguiente forma:

(Tipo de dato) [] (nombre del arreglo) =new (tipo de dato)[tamaño];

Ejemplo: int []vec=new int[50];

Arreglos bidimensionales o Matriz. Se definen en java de la siguiente forma:

(Tipo de dato) [][] (nombre del arreglo) =new (tipo de dato)[fila][columna];

Ejemplo: int [][]mat=new int[50][50];

2.2 OPERADORES RELACIONALES Y MATEMÁTICOS

Existen diferentes tipos de operadores entre los cuales se tienen los siguientes: Operadores matemáticos, lógicos y relacionales.

Operadores Matemáticos. Son todas las operaciones matemáticas básicas, se definen en java de la siguiente forma:

Operación	Operador
Suma	+
Resta	-
División	/
Módulo de División	%
Cociente de División	/

Ejemplo:

f=a+b; f=a-b; f=a*b; f=a/b; f=a%b; f=a/b;

Jerarquía de los Operadores

Los operadores aritméticos tienen un orden de realizarse las operaciones este orden se muestra en la siguiente tabla:

Tabla 2. Precedencia de Operadores.

Jerarquía de Operadores Aritméticos		
Operador	**Jerarquía**	**Operación**
**	(Mayor)	Potencia
()		Paréntesis
*, /, mod, div		Multiplicación, división, módulo, división entera.
+,-	(Menor)	Suma, resta

Ejemplos

1. [1]7+5-6

Solución: 12 – 6
6

2. 9+7*8-36/5

Solución: 9+56-36/5
9+56-7,2
65-7,2
57,8

3. 15/2*(7+(68-15*33+(45**2/16)/3)/15)+19

Solución: 15/2*(7+(68-15*33+(2025/16)/3)/15)+19
15/2*(7+(68-15*33+126.5625/3)/15)+19
15/2*(7+68-495+126.5625/3)/15+19
15/2*(7+(68-495+42.1875)/15+19
15/2*(7+(-427+42.1875/15)+19
15/2*(7+(-)384.8125/15+19
15/2*(7+(-)25.6541)+19
15/2*(-)18.6541+19
7.5*(-)18.6541+19
-139.9062+19
-120.9062

2.2.1 Ejercicios Propuestos

Resolver los siguientes ejercicios teniendo en cuenta la precedencia de operaciones[2].

1. 7*8*(160 mod 3**3) div 5*13-28
2. 6+3*(4/2-6+4)/2 mod 4
3. 4 * (8 - 3 * 2) + (5 / 3 * 2) * 3.0
4. (5 + 3 · 2 / 6 − 4) · (4 /2 − 3 + 6) / (7 − 8 / 2 − 2)**2
5. 6(-3(8+6-9)+6)+4(-8(1+3-6)+1)

Operadores Relacionales. Son todas las operaciones de comparación y verificaciones básicas, se definen en java de la siguiente forma:

1 CAIRÓ BATTISTUTI, Osvaldo. Metodología de la Programación. Algoritmos, diagramas de flujo y programas, México D.F.:Alfaomega.2005, p 13-23.

2 CAIRÓ BATTISTUTI, Osvaldo. Metodología de la Programación. Algoritmos, diagramas de flujo y programas, México D.F.:Alfaomega.2005.

Tabla 3. Operadores Relacionales.

Operación	Operador
Asignación	=
Igual	==
Diferente	<>
Menor	<
Mayor	>
Menor Igual	<=
Mayor Igual	>=

Ejemplo:

1. A=5 B=16
   (A**2)>(B*2)
   25 > (B*2)
   25>32
   FALSO

2. X=6 B=7.8
   (X * 5 + B ** 3 / 4) <= (X ** 3 div B)
   (X * 5 + B ** 3 / 4) <= (X ** 3 div B)
   (X * 5 + 474.552 / 4) <= (X ** 3 div B)
   (30 + 474.552 / 4) <= (X ** 3 div B)
   (30 + 118.638) < = (X ** 3 div B)
   148.638 <= (X ** 3 div B)
   148.638 <= (216 div B)
   148.638 < = 27
   FALSO

2.2.2 Ejercicios Propuestos

1. A=2 B=5
   (A – 4 * B) > (A / B) = (B / A)
2. D=6 E=3
   D**2 + (4 – E) <= (E *D + 5)
3. 9*(4 – 7 / 5) mod 3 > 8 div (4 – 9)
4. 10+ (24 – 31) ** 2 > (3 * 4 + 9 -3) < (2 – 1)
5. A=-1 B=2
   (A * 4 - B) > (A ** B) = (B * A)

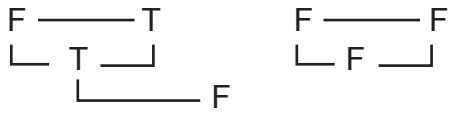
Operadores Lógicos. Son todas las operaciones que permiten interconectar verificaciones, se definen en java de la siguiente forma:

Operación	Operador
Y	&&
O	||

Ejemplo[3]**:**

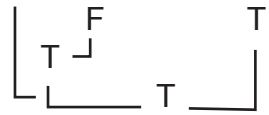
Sea: a = 10 b = 12 c = 13 d =10

1. ((a>b) or (a<c)) and ((a=c) or (a>=b))



2. ((a>=b or (a<d)) and (a>=d) and (c>d))

3. not (a=c) and (c>b)



2.2.3 Ejercicios Propuestos[4]

1. Not (99 >= 50 or 10 > = 75)
2. Not (3+4-4>5 and (not(4>=2 and 2^2<4) or 44< 5^3))
3. Not ((10 * 4) > (15 /2 * 6 > = 15 * 2 / 17 = 15))
4. (15 > = 7 * 3 ** 2 Y 8 > 3 Y 15 > 6) O NO (7 * 3 < 5 + 12 * 2 div 3 ** 2)

2.3 ETAPAS PARA LA SOLUCIÓN DE PROBLEMAS POR COMPUTADOR

Para resolver un problema es necesario seguir las siguientes etapas:

a. Identificación de Datos de entrada, Datos de Salida y proceso necesario para dar solución al problema (Análisis del Problema).
b. Desarrollar el algoritmo utilizando pseudocódigo[5].
c. Las características de un buen algoritmo son:
   - Debe tener un punto particular de inicio.
   - Debe ser definido, no debe permitir dobles interpretaciones.
   - Debe ser general, es decir, soportar la mayoría de las variantes que se puedan presentar en la definición del problema.
   - Debe ser finito en tamaño y tiempo de ejecución.
   - Diseño del Algoritmo

3 **URBAEZ, Wilder. "**Qué son los operadores y los operando, sus tipos y las prioridades de ejecución de los mismos.", Contenido de DesarrolloWeb.com, Edición Internet, Septiembre de 2005, http://www.desarrolloweb.com/articulos/2165.php, Consultada 2011

4 HERNÀNDEZ, Marvin. Ejercicios de Operadores Lógicos Recomendados, Edición Internet, http://inforutec.blogspot.com/2008/08/ejercicios-de-operadores-logicos.html, Agosto 2008.

5 LOPEZ García, Juan Carlos. "Educación Básica Algoritmos y Programación, Guía para Docentes, Edición Internet, http://www.eduteka.org/pdfdir/AlgoritmosProgramacion.pdf, Segunda Edición 2007,2009

d. Realizar Pruebas para la comprobación de los cálculos realizados.
e. Realizar programa computacional en el lenguaje requerido.
f. Ejecución del Programa.

Datos de Entrada: Son los datos que provee el problema o que son necesarios para realizar los cálculos.
Datos de Salida: Son los datos que se desean calcular en el problema.
Proceso: Son las operaciones, verificaciones que deben realizarse con los datos de entrada para calcular los datos de salida.

2.4. ESTRUCTURAS ALGORITMICAS

Existen diversas instrucciones que son necesarias al momento de desarrollar una solución entre las básicas se encuentran las siguientes:

2.4.1 ESTRUCTURAS DE ENTRADA O LEER

Pseudocódigo

Las estructuras de entrada permiten ingresar datos al programa a través del teclado.

Dichas se representan en Pseudocódigo a través del siguiente código:

Leer Nombre de la variable;

Ejemplo

Leer Variable

Instrucción de Entrada o Leer en Netbeans

Las instrucciones de entrada son aquellas que permiten ingresar datos a la solución es decir, **lea** o **leer.** Existen dos formas de realizar las lecturas en Netbeans como son:

Por consola:

La consola es el que se encuentra en la parte inferior de la ventana, donde tiene como título en la pestaña output.

- Como primer paso debe definir el proceso de lectura de la siguiente forma:

  BufferedReader **variable** = new BufferedReader(new InputStreamReader(System.in));

  Variable: la variable es colocada de acuerdo al criterio personal, esta instrucción permite simplificar el proceso de lectura.

Pero para realizar este primer paso es necesario definir en las librerías la siguiente:

import java.io.*;

- Como segundo paso

Variable a leer= Tipo de dato de la variable.parse tipo de dato de la variable(variable.readline()).

Ejemplo:

num=Integer.parseInt(lectura.readLine());

Teniendo en cuenta que la variable num es definida como entera es decir int.

Por Ventana de Windows:

- Como primer paso debe declarar la variable a leer:

Pero para realizar este primer paso es necesario definir en las librerías la siguiente:

import javax.swing.*;

- Como segundo paso

Variable a leer= Tipo de dato de la variable.parse tipo de dato de la variable(JOptionPane.showInputDialog("Mensaje")).

Ejemplo:

num=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el valor"));

num=Float.parseFloat(JOptionPane.showInputDialog("Ingrese el valor"));

num=Double.parseDouble(JOptionPane.showInputDialog("Ingrese el valor"));

Teniendo en cuenta que la variable num es definida como entera es decir int.

2.4.2 ESTRUCTURAS DE SALIDA O ESCRIBIR

Pseudocódigo

Las estructuras de salida se representan en Pseudocódigo a través del siguiente código:

Escribir ("Mensaje", Variable);

Ejemplo:

Escribir (“El resultado es”, N);

Por consola:

System.out.println(“Mensaje”+variable)

Ejemplo:

System.out.println(“El resultado es ”+num);

La variable donde se almacena el resultado es num, y la coma es reemplazada por un símbolo + que significa unir el mensaje con el valor almacenado en la variable, o en el lenguaje técnico concatenación.

Por Ventana de Windows:

Pero para realizar lo siguiente es necesario definir en las librerías la siguiente:

import javax.swing.*;

Luego,

JOptionPane.showMessageDialog(“Mensaje”+variable).

Ejemplo:

JOptionPane.showMessageDialog(“el resultado es ”+num)*;*

Ejemplo del código completo en netbeans donde se realiza una entrada, es decir, una lectura, y una escritura, es decir, escribir.

```
import java.io.*;

public class ejercicio1 {

public static void main(String[]args)throws IOException {

BufferedReader lectura = new BufferedReader(new InputStreamReader(System.
in));
int num;

System.out.println(“Ingrese numero: “);

num = Integer.parseInt(lectura.readLine());


System.out.println(“su número es “+num);
  }
}
```

Ejemplo del código completo en netbeans donde se realiza una entrada, es decir, una lectura, y una escritura utilizando ventanas de Windows, es decir, escribir.

```
import javax.swing.*;
public class ejercicio1 {

    /** Creates a new instance of ejercicio1 */
    public static void main(String args[]) {
        int n,d;
      n=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el valor del
numerador"));
        JOptionPane.showMessageDialog(null,"El valor ingresado es "+n);
        d=n*n;
         JOptionPane.showMessageDialog(null,"El valor de la potencia es "+d);
    }
}
```

2.4.3 ESTRUCTURAS SECUENCIALES

En este caso se ejecutan las instrucciones una tras otra, es decir de forma lineal.

Operador Asignación

Este operador tiene gran importancia para plantear la solución a un problema, debido a que permite guardar valores en la variable.

Se debe tener en cuenta que en una solución algorítmica los datos o expresiones matemáticas deben ser guardadas en las variables utilizando este operador, y se representa por el siguiente símbolo: =

Ejemplo:

1. a1=2+3
   b=a1+4

Tabla 4. Ejemplo 1.

Número de Asignación	a1	b
1	5	
2		9

2. Supongamos que las variables I, ACUM y J son de tipo entero, REA y SUM de tipo real, CAR de tipo carácter y BAND de tipo booleano[6]. Consideremos también que tenemos que realizar las siguientes asignaciones:

6 CAIRÓ BATTISTUTI, Osvaldo. Metodología de la Programación. Algoritmos, diagramas de flujo y programas, México D.F.:Alfaomega.2005, p 13-23.

1. I<- 0
2. I<-I+1
3. ACUM<-0
4. J<-5**2 div 3
5. Car<-‘a’
6. ACUM<- J div I
7. REA<- ACUM/3
8. BAND<-(8>5)y(15<2**3)
9. SUM<-ACUM*5/J**2
10. I<-I*3
11. REA<-REA/5
12. BAND<-BAND o (I=J)
13. I<-REA
14. CAR<-J

Tabla de Memoria							
Número de Asignación	**I**	**J**	**ACUM**	**REA**	**SUM**	**CAR**	**BAND**
1	~~0~~						
2	~~1~~						
3			~~0~~				
4		8					
5						~~a´~~	
6			8				
7				~~2.66~~			
8							~~FALSO~~
9					~~0.625~~		
10	~~3~~						
11				0.532			
12							FALSO
13	ERROR						
14						ERROR	

Ejercicios Propuestos

Realizar las tablas de Memoria de las siguientes asignaciones.
1. Se supone que las variables A, B y J son de tipo entero, C y D de tipo real, CAD de tipo carácter y DEC de tipo booleano. Consideremos también que tenemos que realizar las siguientes asignaciones:

1. A<- 1
2. A<-A+3
3. J<-0
4. J<-4**3 mod 3
5. Cad<-‘Casa Verde’
6. B<- J div A
7. C<- B/3
8. DEC<-(8>5)y(15<2**3)

9. D<-B*5/J**2
10. A<-A*3
11. C<-C/5
12. DEC<-DEC o (I=J)
13. A<-C
14. CAD<-J

2. Se supone que las siguientes variables inician con los siguientes valores BETA=200, CES=0.5, CONT=800, SUM=600, AME=100, PAG=40, TEN=10, RES=2, MAX=2000, ULT=0.20

1. TOTA<-BETA*CES+10
2. ROT<-PAG
3. MAN<- MAX / TEN
4. FAC<-PAG * AME / 2
5. BOTA<-CONT + SUM –MAX
6. FAC<-RES * TEN * AME
7. TOTA<-60 + PAG – AME
8. ZOR<-MAX * ULT / 2 + ROT
9. CARO<-BETA + AME + TEN + ULT
10. NEKO<- MAX * AME + PAG – RES * CARO

Estructura Inicio – Fin

Esta estructura permite delimitar la solución del problema a través del Pseudocódigo así:

Inicio
Instrucciones;
Fin

Ejemplo.

- Realizar un algoritmo que dados dos (2) números, calcule la suma de dichos números.
  Datos de Entrada: Número1 y Número2
  Datos de Salida: Suma
  Proceso: Número1 + Número2
  Solución

```
Inicio
  Escribir("Ingrese el Primer Número")
  Leer n1
  Escribir("Ingrese el Segundo Número")
  Leer n2
  S=n1+n2
  Escribir("La suma es ", S)
Fin
```

 Realizar un algoritmo que dados base y altura. Calcule el área del triángulo y el rectángulo.

Datos de Entrada: base, altura
Datos de Salida: Área del triángulo y Área del Rectángulo
Proceso: Área del Triángulo = (b*h)/2
Área del Rectángulo =b*h

Solución

```
Inicio
  Escribir("Ingrese la base del Triángulo")
  Leer bt
  Escribir("Ingrese la altura del triángulo")
  Leer ht
  At=(b*h)/2
  Escribir("El área del Triángulo es ", At)
  Escribir("Ingrese la base del Rectángulo")
  Leer br
  Escribir("Ingrese la altura del Rectángulo")
  Leer hr
  Ar=(b*h)
  Escribir("El área del Rectángulo es ", Ar)
Fin
```

Solución en Netbeans

```
import javax.swing.*;
public class Áreas{
   public static void main(String args[]) {
     double bt,ht,At,br,hr,Ar;
     bt=Double.parseDouble(JOptionPane.showInputDialog("Ingrese la base del
    Triángulo"));
ht=Double.parseDouble(JOptionPane.showInputDialog("Ingrese la altura del
    Triángulo"));
At=(b*h)/2;
JOptionPane.showMessageDialog(null,"El área del triángulo es " "+At);
      br=Double.parseDouble(JOptionPane.showInputDialog("Ingrese la base del
    Rectángulo"));
hr=Double.parseDouble(JOptionPane.showInputDialog("Ingrese la altura del
    Rectángulo"));
Ar=(br*hr);
       JOptionPane.showMessageDialog(null,"El área del Rectángulo es " "+Ar);
   }
}
```

 Dado el nombre y tres 3 calificaciones de un estudiante obtenidas a lo largo del semestre. Realice un algoritmo que muestre el nombre del estudiante y el promedio de sus calificaciones.

Datos de Entrada: nombre, nota 1, nota 2, nota 3
Datos de Salida: Promedio
Proceso: Promedio= Suma de notas/ número de notas

Solución en Pseudocódigo

Inicio
Escribir("Ingrese el nombre del Estudiante)
Leer nombre
Escribir("Ingrese la primera nota)
Leer n1
Escribir("Ingrese la segunda nota)
Leer n2
Escribir("Ingrese la tercera nota)
Leer n3
P=(n1+n2+n3)/3
Escribir("El promedio de ",nombre," es de ",P)
Fin

Solución en Netbeans

```
import javax.swing.*;
public class Promedio {
    public static void main(String args[]) {
       String nombre;
       double n1,n2,n3;
        nombre=JOptionPane.showInputDialog("Ingrese el valor del numerador");
n1=Double.parseDouble(JOptionPane.showInputDialog("Ingrese el número"));
n2=Double.parseDouble(JOptionPane.showInputDialog("Ingrese el número"));
n3=Double.parseDouble(JOptionPane.showInputDialog("Ingrese el número"));
P=(n1+n2+n3)/3;
         JOptionPane.showMessageDialog(null,"El promedio de "+nombre+ "es de
     "+P);
    }
}
```

Ejemplos en Netbeans de Estructuras Secuenciales

- Este programa calcula la potencia cuadrada de un número dado

```
import javax.swing.*;
public class ejercicio1 {

    /** Creates a new instance of ejercicio1 */
    public static void main(String args[]) {
        int n,d;
    n=Integer.parseInt (JOptionPane.showInputDialog("Ingrese el valor del
      numerador"));
   d=n*n;
        JOptionPane.showMessageDialog(null,"El valor de la potencia es "+d);
    }
  }
```

Para crear un nuevo programa en el mismo proyecto solo debe volver a seguir los pasos para crear una clase que son:

- ✓ Encima del nombre del proyecto que aparece en la parte izquierda de la ventana, se da click derecho, y en el menú que se despliega, se escoge la opción **New**, y luego la opción **Java Class**.
- ✓ En la ventana que se obtiene, en la parte en donde se encuentra **Class Name**, se coloca el nombre del programa.
- ✓ Presiona el botón **Finísh.**

De esta forma se creará un nuevo espacio para realizar otro programa, recuerde que el nombre de la clase debe ser referente al objetivo del programa, además que no debe contener espacios y debe empezar por letra.

2.4.4 Ejercicios Propuestos

1. Realizar un algoritmo que dado el valor de la compra y el valor a pagar por el cliente, calcular el valor a devolver al cliente.

2. Realizar un algoritmo que dado la cantidad de mujeres y hombres de un grupo, calcular el porcentaje de hombres y mujeres del grupo.

3. Realizar un algoritmo dadas las notas del primer parcial y del segundo parcial de un estudiante donde el primer y segundo parcial equivalen al 30% y 30% respectivamente, Calcular que nota debe obtener para ganar la asignatura teniendo en cuenta que el final equivale al 40% y la nota para ganar la asignatura es de 3.0.

4. Realizar un algoritmo que dado el lado de un cuadrado, calcule al área y perímetro del cuadrado.

5. Realizar un algoritmo que dado dos números enteros y muestre su suma, resta, multiplicación, división.

6. Realizar un algoritmo que una vez se ingrese una medida expresada en centímetros la convierta en pulgadas (1pulgada = 2,54 centímetros).

7. Realizar un algoritmo que exprese en horas, minutos y segundos un tiempo expresado en segundos.

8. Realizar un algoritmo que dado el total de kilómetros recorridos, el precio de la gasolina (por litro), el dinero de gasolina gastado en el viaje y el tiempo que se ha tardado (en horas y minutos) y que calcule:

- Velocidad media (en km/h y m/s).
- Consumo de gasolina (en litros y euros) por cada 100 km.
- Consumo de gasolina (en litros y euros) por cada km.

9. La presión, el volumen y la temperatura de una masa de aire se relacionan por la fórmula:

masa = (presión * volumen)/(0.37 * (temperatura + 460))

Pedir los datos de Presión, volumen y temperatura.

10. Calcular el número de pulsaciones que una persona debe tener por cada 10 segundos de ejercicio, si la fórmula es:

num_pulsaciones = (220 - edad)/10

11. En un hospital existen tres áreas: Ginecología, Pediatría, Traumatología. El presupuesto anual del hospital se reparte conforme a la siguiente Tabla:

Área	Porcentaje del presupuesto
Traumatología	30%
Ginecología	40%
Pediatría	30%

Obtener la cantidad de dinero que recibirá cada área, para cualquier monto presupuestal.

12. Todos los lunes, miércoles y viernes, una persona corre la misma ruta y cronometra los tiempos obtenidos. Determinar el tiempo promedio que la persona tarda en recorrer la ruta en una semana cualquiera.

13. En una gasolinera tiene el siguiente problema, los surtidores registran lo que surten en galones, pero el precio de la gasolina está fijado en litros. El algoritmo debe calcular e imprimir lo que hay que cobrarle.
1 Galón tiene 3785 litros.

14. Realizar un algoritmo que dado como datos el radio y la altura de un cilindro, calcule e imprima el área y su volumen.

15. Realizar un algoritmo que calcule la distancia entre dos puntos, dado como datos las coordenadas de los puntos P1 y P2.

$$D=\sqrt{(x1-x2)^2+(y1-y2)^2}$$

16. La relación entre los lados (a ,b) de un triángulo rectángulo y la hipotenusa (h) viene dada por la formula

$$a2 + b2 = h2 \quad h: \text{hipotenusa}$$

Escribir un algoritmo que lea la longitud de los dos lados y calcule la hipotenusa.

2.4.5 OPERADORES LÓGICOS

Los operadores lógicos son resultados de evaluación de expresiones que responden a condiciones que permiten la toma de decisiones, los operadores utilizados son: Y, O, NO.

- **Y o también conocido como AND**. De este operador se pueden obtener los siguientes resultandos:
  V= verdadero
  F=Falso

Tabla 5. Operador Y

Expresión 1(E1)	Expresión 2(E2)	E1 Y E2
V	V	V
V	F	F
F	V	F
F	F	F

En este operador el resultado de la evaluación de la expresión será verdadero solo si todas las expresiones de esta son verdaderas, de lo contrario será falsa. Se debe tener en cuenta que para utilizar estas expresiones se utilizan los operadores relacionales como se mostró anteriormente en el ítem de operadores, entre los cuales están <,>, <=, >=, =, <>.

En java se representa con el siguiente símbolo: **&&**

Ejemplo:

((a<b) && (b>c) && (c=d) && (d<>f))

- **Conocido como OR.** De este operador se pueden obtener los siguientes resultandos:
  V= verdadero
  F=Falso

Tabla 6. Operador O

Expresión 1(E1)	Expresión 2(E2)	E1 O E2
V	V	V
V	F	V
F	V	V
F	F	F

En este operador el resultado de la evaluación de la expresión será falsa solo si todas las expresiones de esta son falsas, de lo contrario será verdadera. Se debe tener en cuenta que para utilizar estas expresiones se utilizan los operadores relacionales como se mostró anteriormente en el ítem de operadores, entre los cuales están <,>, <=, >=, =, <>.

En java se representa con el siguiente símbolo: **||**

Ejemplo:

((a<b) || (b>c) || (c=d) || (d<>f))

- **NO o también conocido como NOT.** De este operador se pueden obtener los siguientes resultandos:
  V= verdadero
  F=Falso

Tabla 7. Operador NO

Expresión 1(E1)	NO (E1)
V	F
V	F
F	V
F	V

Se debe tener en cuenta que para utilizar estas expresiones se utilizan los operadores relacionales como se mostró anteriormente en el ítem de operadores, entre los cuales están <,>, <=, >=, =, <>.

En java se representa con el siguiente símbolo: **||**

Ejemplo:

NO ((a<b) || (b>c) || (c=d) || (d<>f))
NO ((a<b) || (b>c) || (c=d) || (d<>f))

2.4.6 ESTRUCTURAS CONDICIONALES

Las estructuras condicionales permiten verificar el estado de expresiones lógicas como son las comparaciones entre otras.

La estructura básica se representa en Pseudocódigo así:

```
Si (Condición) entonces
        Instrucciones
F-si
```

Condición: es una o varias expresiones lógicas, en el cual si son varias expresiones deben ser unidas a través de los operadores lógico Y, O.

Expresión Lógica: Son operadores unidos con operaciones relacionales como <, >, <= ,>= ,= y <>. Ejemplo (a<b) , (b>=c).

Otras formas de representar esta estructura son las siguientes:

La estructura **Si-sino-F-si**

Se utiliza en el caso donde se tengan alternativas para valorar si son verdaderas o falsas.
La condición o condiciones valoradas en el **Si** de la estructura determina si es **verdadera** realiza las instrucciones dentro de esta estructura en caso de que sea **Falsa** se realizan las instrucciones colocadas en la estructura del **sino**.

Pseudocódigo:

```
Si (Condiciones) Entonces
    Instrucciones
Sino
   Instrucciones
Fin-si
```

La estructura **Si en cascada**.

Se utiliza en situaciones donde se realizan varias verificaciones distintas para dar solución a los diferentes problemas.

Pseudocódigo:

```
Si (Condiciones) Entonces
 Si (Condiciones) Entonces
    Instrucciones
 Fin-si
Sino
   Si (Condiciones) Entonces
    Instrucciones
  Sino
    Si (Condiciones) Entonces
    Instrucciones
```

```
    Sino
      Instrucciones
    …. Pueden colocar otras condiciones de ser necesario….
   Fin-si
  Fin-si
Fin-si
…. Los cierres necesarios de acuerdo a las condiciones colocadas.
```

Ejemplo

1. Realizar un algoritmo que dada la edad de una persona le indique si puede o no votar.

   Datos Entrada: edad
   Datos Salida: Votar o no votar
   Proceso: En Colombia las personas que pueden votar son aquellas que tienen de 18 años de edad en adelante.

Solución:

```
Inicio
Escribir("Ingrese su Edad")
Leer edad
Si (edad>=18) Entonces
  Escribir ("Usted puede Votar en Colombia")
Sino
    Escribir ("Usted no puede Votar en Colombia")
Fin-Si
```

2. Desarrolle un algoritmo que permita leer tres valores y almacenarlos en las variables A, B y C respectivamente. El algoritmo debe imprimir cual es el mayor. Recuerde constatar que los tres valores introducidos por el teclado sean valores distintos.

   Datos Entrada: Tres valores A, B y C
   Datos Salida: El valor mayor
   Proceso: Comparación entre A, B y C

   Solución:

```
Inicio
Escribir("Ingrese primer valor")
Leer a
Escribir("Ingrese segundo valor")
Leer b
Escribir("Ingrese tercer valor")
Leer c
Si (a>b) y (a>c) Entonces
  Escribir ("El valor mayor es ",a)
Sino
 Si (b>a) y (b>c) Entonces
```

```
    Escribir (“El valor mayor es ”,b)
   Sino
     Escribir (“El valor mayor es ”,c)
   Fin-Si
  Fin-Si
 Fin
```

En Netbeans esta estructura se representa así:

```
if(condiciones){
     Instrucciones;
 } else {
     Instrucciones;
}
```

```
If (condiciones) {
  Instrucciones

}
```

```
if(condiciones){
  Instrucciones
}else{
if (condiciones) {
  Instrucciones
}else{
  Instrucciones
 }
}
```

Se puede utilizar cualquiera de las estructuras anteriores según sea necesario.

Ejemplos de Estructuras condicionales en Netbeans

- Este programa verifica si un número es positivo o negativo.

```
import javax.swing.*;
public class ejercicio1 {

    /** Creates a new instance of ejercicio1 */
    public static void main(String args[]) {
        int n1,n2;
        double d;
            n1=Integer.parseInt(JOptionPane.showInputDialog(“Ingrese el valor del
     numerador”));
            n2=Integer.parseInt(JOptionPane.showInputDialog(“Ingrese el valor del
     denominador”));
        if(n2==0){
                JOptionPane.showMessageDialog(null,”La división por cero no está
     definida”);
        }else{
             d=n1/n2;
             JOptionPane.showMessageDialog(null,”El valor de la potencia es “+d);
        }
    }
  }
```

 Este programa verifica si una división puede o no realizarse.

```
import javax.swing.*;
public class ejercicio1 {

    /** Creates a new instance of ejercicio1 */
    public static void main(String args[]) {
        int n1,n2;
        double d;
            n1=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el valor del
     numerador"));
            n2=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el valor del
     denominador"));
        if(n2==0){
                JOptionPane.showMessageDialog(null,"La división por cero no está
     definida");
        }else{
            d=n1/n2;
            JOptionPane.showMessageDialog(null,"El valor de la potencia es "+d);
        }
    }
}
```

2.4.7 Ejercicios Propuestos

1. [78]A un trabajador le pagan según sus horas y una tarifa de pago por horas. si la cantidad de horas trabajadas es mayor a 80 horas. la tarifa se incrementa en un 13% para las horas extras. Realizar un algoritmo que le permita calcular el salario del trabajador dadas las horas trabajadas y la tarifa.

2. A un trabajador le descuentan de su sueldo el 10% si su sueldo es menor o igual a 100000. por encima de 100000 y hasta 200000 el 5% del adicional, y por encima de 200000 el 3% del adicional. Realizar un algoritmo que le permita calcular el descuento y sueldo neto que recibe el trabajador dado su sueldo.

3. Dado un monto calcular el descuento considerando que por encima de 100000 el descuento es el 10% y por debajo de 100000 el descuento es el 2%.

4. Dado un tiempo en segundos, Realizar un algoritmo que le permita calcular los segundos restantes que le correspondan para convertirse exactamente en minutos.

5. Dado un tiempo en minutos, Realizar un algoritmo que le permita calcular los días, horas y minutos que le corresponden.

6. Dada las horas trabajadas de una persona y la tarifa de pago. Realizar un algoritmo que le permita calcular su salario e imprimirla.

7 **BELTRAN, Fabio**. Prácticas Introducción a la Programación.

7. Realizar un algoritmo que le permita emitir la factura correspondiente a una compra de un artículo determinado, del que se adquieren una o varias unidades. El IVA es del 16% y si el precio bruto (precio venta más IVA) es mayor de 1000 pesos se debe realizar un descuento del 5%.

8. Realizar un algoritmo que le permita ¿Calcular las raíces de una ecuación de segundo grado ($ax^2+bx+c=0$)? Tenga en cuenta que puede utilizar la formula general.

9. ¿Dada la duración en minutos de una llamada realizar un algoritmo que le permita calcular el costo, considerando?
   ℭ. Hasta tres minutos el costo es 0.50.
   𝔽. Por encima de tres minutos es 0.50 más 0.1 por cada minuto adicional a los tres primeros.

10. ¿Dado el monto de una compra realizar un algoritmo que le permita calcular el descuento considerado
    a. Descuento es 10% si el monto es mayor a $50000
    b. Descuento es 20% si el monto es mayor a $20000 bolivianos y menor o igual a $50000.
    c. no hay descuento si el monto es mayor o igual a $10000.

12. Ingresar un número determinado de segundos, realizar un algoritmo que le permita convierta en horas, minutos y segundos.

13. Realizar un algoritmo que permita ingresar la hora, minutos y segundos y que me calcule la hora en el siguiente segundo (“0=< H =<23”, “0=< M =<59” “0=< S=<59”).

14. Realizar un algoritmo que permita que al dar la hora hh, mm, ss, muestre las horas, minutos y segundos y también nos calcule la hora después de 2 segundos.

15. Realizar un algoritmo que permita que lea 2 números y deducir si están en orden creciente o decreciente.

16. A un profesor le pagan según sus horas y una tarifa de pago por horas. Si la cantidad de horas trabajadas es mayor a 40 horas, la tarifa se incrementa en un 50 % para las horas extras. Realizar un algoritmo que calcule el salario del profesor dadas las horas trabajadas y la tarifa.

17. Realizar un algoritmo que Calcule el perímetro de una circunferencia dado su radio. Luego calcule el perímetro de la misma si se reduce al 50%. Luego calcule el perímetro de la misma si se reduce al 25% con respecto al resultado anterior.

2.4.7 ESTRUCTURAS SELECTIVAS

La estructura selectiva permite escoger dentro de un conjunto de opciones una dentro del conjunto.

La estructura selectiva se representa en Pseudocódigo así:

```
DD (Variable) Haga
        Caso valor1: Instrucciones;
        Caso valor2: Instrucciones;
        Caso valor3: Instrucciones;
        .
        Caso valor n: Instrucciones;
        En otro caso: Instrucciones;
Fin-DD
```

Variable: Almacena el valor seleccionado del conjunto de opciones.

En Netbeans esta estructura se representa así:

```
switch (variable) {
        Case valor1: Instrucciones;
        break;
        Case valor2: Instrucciones;
        break;
        Case valor3: Instrucciones;
        break;
        .
        .
        .
        Case valor n: Instrucciones;
        break;
        default: Instrucciones;
}
```

Ejemplos de Estructuras condicionales en Netbeans

Desarrollar un programa que permita al usuario realizar la operación matemática básica deseada (+, -, *, /) entre dos números enteros.

```
import javax.swing.*;
public class ejercicio1 {
  public static void main(String args[]) {
      int n1,n2,op;
      double d;
        op=Integer.parseInt(JOptionPane.showInputDialog("Menú de
        Opciones\n"+"1.Suma\n"+"2.Resta\n"+"3.Multiplicacion\n"+"4.División\
        n"+"Escoja la opción deseada"));
        n1=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el valor del
        numerador"));
        n2=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el valor del
        denominador"));
   switch(op){
      case 1: d=n1+n2;
          JOptionPane.showMessageDialog(null,n1+" + "+n2+" = "+d);
      break;
      case 2: d=n1-n2;
          JOptionPane.showMessageDialog(null,n1+" - "+n2+" = "+d);
       break;
       case 3: d=n1*n2;
           JOptionPane.showMessageDialog(null,n1+" * "+n2+" = "+d);
       break;
       case 4:if(n2==0){
                  JOptionPane.showMessageDialog(null,"La división por cero no está
        definida");
             }else{
               d=n1/n2;
               JOptionPane.showMessageDialog(null,n1+" / "+n2+" = "+d);
             }
         break;
         default:
               JOptionPane.showMessageDialog(null,"Opción incorrecta ");
        }
      }
}
```

2.4.8 Problemas Propuestos

1. [9]Supóngase que el importe del seguro obligatorio de un coche depende del modelo del coche, del color y de la edad del conductor. Sean dos modelos de coche A y B y los precios del seguro según el color:

9 CAIRÓ BATTISTUTI, Osvaldo. Metodología de la Programación. Algoritmos, diagramas de flujo y programas, México D.F.:Alfaomega.2005,

9 CAIRÓ BATTISTUTI, Osvaldo. Metodología de la Programación. Algoritmos, diagramas de flujo y programas, México D.F.:Alfaomega.2005,

Tabla 8. Precios Vs Modelo

MODELO	COLOR	PRECIO
A	BLANCO	240.41
A	METALIZADO	330.00
A	OTROS	270.50
B	BLANCO	300.00
B	METALIZADO	360.50
B	OTROS	330.00

Si el conductor tiene menos de 26 años, el precio se incrementa un 25%; si tiene entre 26 y 30 años se incrementa un 10%; si tiene entre 31 y 65 años el precio no se modifica; si tiene más de 65 años el precio se incrementará un 10%. Además, en cualquier caso, hay que considerar que si el conductor tiene menos de 2 años el permiso de conducir, el precio se incrementará un 25% adicional. Diseñar un algoritmo que calcule el precio del seguro para un determinado modelo y un determinado conductor.

2. Realizar un algoritmo que dada las figuras circulo, triangulo, rectángulo y cuadrado, según desee el usuario calcular el perímetro de la figura.

3. Realizar un algoritmo que dadas las siguientes figuras cubo, esfera, cono cilindro, según desee el usuario, calcular el volumen de la figura.

4. Escriba un algoritmo que lea tres números enteros de un supuesto triángulo, determine si realmente forman un triángulo, y muestre el tipo de triángulo que es (si es un triángulo).

a) triángulo: La suma de dos cualesquiera de los lados debe ser mayor que el otro.
b) equilátero: todos los lados son iguales.
c) isósceles: solo dos lados son iguales.
d) escaleno: no tiene dos lados iguales.

5. [10]Realice un algoritmo que permita calcular lo que hay que pagarle a un trabajador teniendo en cuenta su sueldo y las horas extras trabajadas. Para el pago de horas extras se toma en cuenta la categoría del trabajador-

Tabla 9. Precio Vs Categoría

TABLA DE CATEGORÍAS	
CATEGORÍA	PRECIO HORAS EXTRAS
1	$30000
2	$38000
3	$50000
4	$70000

Cada trabajador puede tener como máximo 30 horas extras, si tienen más sólo se les pagarán 30. A los trabajadores con categoría mayor a 4 no debemos pagarles horas extras.

2.5 ESTRUCTURAS REPETITIVAS

Las estructuras repetitivas permiten repetir los procesos un gran número de veces según sea el caso.

Estas estructuras son utilizadas en procesos donde se requiere realizar las mismas operaciones una gran cantidad de veces. Teniendo en cuanta lo anterior las estructuras repetitivas se clasifican en:

CON BASE A UN CONTADOR

Contador: Es la variable que se encarga de enumerar los objetos que pertenecen a un conjunto de acuerdo a una condición.

Acumulador: Es la variable encargada de adicionar los valores que se requieren totalizar según sea el caso.
La estructura de ciclo repetitivo que pertenece a esta categoría es:

Ciclo Repetitivo Para

Sintaxis en pseudocódigo:

Para (i=Valor Inicial, Valor final, Incremento o decremento) haga
Instrucciones;
FPara

En Netbeans:

for (i=Valor inicial; i<=Valor Final; i++ o i--){
Instrucciones;
}

Ejemplos de Estructura Repetitiva para en Netbeans:

1. Elaborar un programa que muestre los números pares comprendidos entre 10 y 20 inclusive.

Solución en Netbeans:

```
import javax.swing.*;
public class ejercicio1 {
   public static void main(String args[]) {
      int i;
      for (i=10;i<=20;i++){
         JOptionPane.showMessageDialog(null, i+",");
         i=i+2;
      }
   }
}
```

2. Realizar un programa que calcule la tabla de sumar de un valor dado hasta el 10.

Solución en Netbeans:

```
import javax.swing.*;
public class ejercicio1 {
   public static void main(String args[]) {
      int i, n, R;
    n=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el valor para calcular
la tabla de sumar "));
      for (i=1;i<=10;i++){
        R=n+i;
         JOptionPane.showMessageDialog(null, n+"+"+i+"="+R);
      }
   }
}
```

3. Realizar un programa que dado N valores, calcule la cantidad de números pares e impares.

Solución en Netbeans es:

```
import javax.swing.*;
public class ejercicio1 {
   public static void main(String args[]) {
      int i, n, num, p=0,im=0;
      n=Integer.parseInt(JOptionPane.showInputDialog("Ingrese la cantidad"));
      for (i=1;i<=n;i++){
        num=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el valor"));
        if(num % 2==0){
           p=p+1
        }else{
            im=im+1
       }
      }
              JOptionPane.showMessageDialog(null, "En el conjunto de valores
hay"+p+"pares");
JOptionPane.showMessageDialog(null, "En el conjunto de valores
hay"+im+"impares");
   }
}
```

2.5.1 PROBLEMAS PROPUESTOS DE LA ESTRUCTURA REPETITIVA PARA

1. Un estudiante tiene problemas para manejar las tablas de multiplicar, por lo tanto requiere para su ayuda un algoritmo que permita que dado un número N, calcule l tabla de multiplicar de dicho número.

2. Una profesora de matemáticas requiere enseñar a sus estudiantes de una forma fácil y rápida como determinar si un número es primo o no lo es (Número primo es aquel que solo posee dos divisores y esos divisores son la unidad (1) y mismo número), para lo cual se requiere que se realice un algoritmo que dado un número determine si es un número primo o no es primo.

3. El Depto. de Seguridad Publica y Transito de Barranquilla, desea saber, de los n autos que entran a la ciudad, cuantos entran con Stikers de cierto color. Conociendo el último dígito de la placa de cada automóvil se puede determinar el color del Stikers utilizando la siguiente relación:

DÍGITO	COLOR
1 o 2	amarilla
3 o 4	rosa
5 o 6	roja
7 o 8	verde
9 o 0	azul

Realice un algoritmo que asignes los Stikers teniendo en cuenta la tabla anterior.

3. Un docente requiere realizar un programa que permita ingresar las notas de los tres cortes de N estudiantes que se especifica así: Primer parcial el cual tiene un valor del 30% de la nota definitiva, Segundo parcial el cual tiene un valor del 30% de la nota definitiva y examen final el cual tiene un valor del 40% de la nota definitiva y el nombre, calcular la cantidad de estudiantes que obtuvieron notas definitivas entre 4.0 y 5.0 y mostrar el nombre de dichos estudiantes, mostrando además que estos estudiantes obtuvieron un beca por buen rendimiento académico.
4. Realizar un algoritmo dadas las edades de un grupo y los sexos del grupo, Calcular el promedio de edades de hombres, mujeres y de todo el grupo de alumnos.

5. Dado un conjunto de N valores realizar un algoritmo para encontrar el menor valor del conjunto números dados.

6. Dado un conjunto de N valores realizar un algoritmo para encontrar el mayor valor del conjunto números dados.

7. Un grupo de N miembros de un club contra la obesidad desean saber cuánto han bajado o subido de peso desde la última vez que se reunieron. Para esto se debe realizar un ritual de pesaje en donde cada uno se pesa en m básculas distintas para así tener el promedio más exacto de su peso. Si existe diferencia positiva entre este

promedio de peso y el peso de la última vez que se reunieron, significa que subieron de peso. Pero si la diferencia es negativa, significa que bajaron. Lo que el problema requiere es que por cada persona se imprima un letrero que diga: “SUBIO” o “BAJO” y la cantidad de kilos que subió o bajo de peso.

8. Se desea obtener el promedio de g grupos que están en un mismo año escolar; siendo que cada grupo puede tener n alumnos que cada alumno puede llevar m materias y que en todas las materias se promedian tres calificaciones para obtener el promedio de la materia. Realizar un algoritmo para desplegar el promedio de los grupos, el promedio de cada grupo y el promedio de cada alumno.

9. Calcular la suma siguiente:

100 + 98 + 96 + 94 + . . . + 0 en este orden

10. Leer por cada alumno de Diseño estructurado de algoritmos su número de control y su calificación en cada una de las 5 unidades de la materia. Al final que escriba el número de control del alumno que obtuvo mayor promedio. Suponga que los alumnos tienen diferentes promedios.

11. Realizar un algoritmo que calcule la potencia de un número dada la base y el exponente. Ejemplo
BaseExponente = Base1*Base2*Base3*….*Baseexponente
34=3*3*3*3=81

12. Realizar un algoritmo que calcule el factorial de un número, teniendo en cuenta que el factorial está definido así:
Fact(3)=1*2*3=6
Además el factorial de 0 y 1 es 1.

2.5.2 Estructuras Repetitivas con Base a una Condición

Este tipo de estructuras repetitivas se caracteriza por:

- No necesariamente se conoce el número de veces que se repite el proceso.
- Se utilizan las condiciones para generar la repetición del proceso, siempre y cuando se cumpla la condición, es decir, sea verdadera. El ciclo culmina cuand0o la condición se convierte en falsa, por lo tanto es necesario que las variables que participan en la condición sean modificadas dentro del ciclo para que la condición pueda convertirse en falsa.
- Las operaciones que se repiten son las que son comunes para los objetos a los cuales se les realizará dichas operaciones.
- Al igual que el ciclo anterior se utilizan conceptos de Acumulador y Contador.

Ciclo Repetitivo Mientras Que

Sintaxis en pseudocódigo del ciclo:

Mq (Condición) haga
Instrucciones
FMq

Sintaxis en Netbeans del ciclo:

while (condición) {
Instrucciones;
}

Ciclo Repetitivo Haga Mientras Que

Características:

- La condición de repetición del ciclo es evaluada al final de la realización de las operaciones especificadas dentro del ciclo.
- Este solo culmina la repetición cuando la condición es falsa.

Sintaxis en pseudocódigo del ciclo:

Haga
instrucciones
}mq(condición)

Sintaxis en Netbeans del ciclo:

do{
instrucciones;
}while(condición);

Ejemplos

1. Desarrolle un programa que permita calcular Promedio de Notas de N estudiantes; finaliza cuando N = 0.

Solución en Netbeans:

```
import javax.swing.*;
public class ejercicio1 {
   public static void main(String args[]) {
      int N,C;
      double S=0,P,nota;
          N=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el número de
estudiantes"));
C=N;
     While (N!=0) {
       nota=Double.parseDouble(JOptionPane.showInputDialog("Ingrese la nota"));
```

```
        S=S+nota;
        N=N-1;
      }
P=S/C;
      JOptionPane.showMessageDialog(null, “El promedio del grupo es”+P);
   }
}
```

2. Realizar un programa que dados varios números , los acumule hasta que se ingrese el valor de cero. Mostrar el valor acumulado y la cantidad de valores.

Solución en Netbeans:

```
import javax.swing.*;
public class ejercicio1 {
   public static void main(String args[]) {
      int C=0;
      double S=0,num;
       num=Double.parseDouble(JOptionPane.showInputDialog(“Ingrese el número
“));

     While (num != 0) {
        S=S+num;
num=Double.parseDouble(JOptionPane.showInputDialog(“Ingrese el valor”));
        C=C+1;
      }
 JOptionPane.showMessageDialog(null, “El valor Acumulado es ”+S+” de “+C+”
números”));
   }
}
```

3. Realizar un algoritmo que calcule la cantidad de divisores de un número entero.

Solución en Netbeans:

```
import javax.swing.*;
public class divisores {
   public static void main(String args[]) {
      int C=0,n,I,D;
      n=Integer.parseInt(JOptionPane.showInputDialog(“Ingrese el número “));
      I=1;
      D=0;
     do {
       if(n%I==0){
        D=D+1;
       }
       I=I+1;
     }while(I<=n);
     JOptionPane.showMessageDialog(null, n+“ contiene ”+D+” Divisores “));
   }
}
```

Tabla 10. Estructura Repetitiva

PRIMITIVA REPETITIVA		

Mq (Condición) haga
Instrucciones
FMq
	while (condición) {
	Instrucciones;
	}
		while (condición) {
		Instrucciones;
		}

Para (i=Valor Inicial, Valor final, Incremento o decremento) haga
Instrucciones;
FPara
	for (i=Valor inicial; i<=Valor Final; i++ o i--){
	Instrucciones;
	}
		for (i=Valor inicial; i<=Valor Final; i++ o i--){
		Instrucciones;
		}

Haga instrucciones }mq(condición)	do{ instrucciones; }while(condición);	do{ instrucciones; }while(condición);

2.5.3 Problemas Propuestos de la Estructura Repetitiva con Base a una Condición

1. Realizar un algoritmo que dado un número calcular el producto de los dígitos distintos.

Ejemplo: 284=2*8*4=64.

2. Realizar un algoritmos que dado un número calcular la suma de los dígitos hasta que el resultado de dicha suma dé como resultado un número de una cifra.

Ejemplo: 9856=9+8+5+6=28=2+8=10=1+0=1

3. Se tienen como datos los importes de las ventas de cada una de las sucursales de una empresa, junto con el código de sucursal (1, 2, 3, 4 ó 5).- Cada sucursal puede tener varias ventas. Los datos no están ordenados por código de sucursal. Un código igual a cero indica fin de datos.- Obtener el total de ventas para cada sucursal.

4. [11]Realizar un algoritmo que pregunte al usuario un número comprendido en el rango de 1 a 5. El algoritmo deberá validar el número, de manera que no continúe la ejecución del programa mientras no se escriba un número correcto.

5. [12]Hacer un programa que de vueltas y en cada vuelta realice las potencias de 5. En cada vuelta se pide al operador SI DESEA CONTINUAR (1 o 0) si el operador ingresa un cero ya no continuará dando vueltas.

---

10 Urbaez. Wilder. Estructuras Cíclicas. http://www.desarrolloweb.com/articulos/2249.php

11 Barueto Enrique. Ejercicios de Estructuras Repetitivas. http://enriquebarrueto0.tripod.com/clases/clase5/clase5_2.htm

11 Barueto Enrique. Ejercicios de Estructuras Repetitivas Soluciones http://enriquebarrueto0.tripod.com/ebarrueto1/solu2.htm

6. Implemente el siguiente juego: el programa seleccionará un número aleatorio entre 0 y 100 y el jugador debe acertarlo [13]. En cada intento el jugador propondrá una solución y se le informará si el número a acertar es menor o mayor que el propuesto. El juego termina cuando se acierte la cifra o haya realizado un máximo de 12 intentos en cuyo caso se le mostrará al jugador la calificación obtenida según la siguiente tabla:

Tabla 11. Intentos Vs Categoría

Número de intentos	Categoría
1-3	Suertudo
4-6	Genio
7	No está mal
8	Se puede mejorar
>=0	Que Pasa Amigo

BIBLIOGRAFÍA

1. CAIRÓ BATTISTUTI, Osvaldo. Metodología de la Programación. Algoritmos, diagramas de flujo y programas, México D.F.:Alfaomega.2005.

2. URBAEZ, Wilder. “Qué son los operadores y los operando, sus tipos y las prioridades de ejecución de los mismos.”, Contenido de DesarrolloWeb.com, Edición Internet, Septiembre de 2005, http://www.desarrolloweb.com/articulos/2165.php, Consultada 2011

3. HERNÀNDEZ, Marvin. “Ejercicios de Operadores Lógicos Recomendados”, Edición Internet, http://inforutec.blogspot.com/2008/08/ejercicios-de-operadores-logicos.html, Agosto 2008.

4. LOPEZ García, Juan Carlos. “Educación Básica Algoritmos y Programación, Guía para Docentes, Edición Internet, http://www.eduteka.org/pdfdir/AlgoritmosProgramacion.pdf, Segunda Edición 2007,2009

5. BELTRAN, Fabio. Prácticas Introducción a la Programación.

6. Urbaez. Wilder. Estructuras Cíclicas. http://www.desarrolloweb.com /articulos /2249.php

7. Barueto Enrique. Ejercicios de Estructuras Repetitivas. http://enriquebarrueto0.tripod.com/ clases/clase5/clase5_2.htm

8. Barueto Enrique. Ejercicios de Estructuras Repetitivas Soluciones http://enriquebarrueto0.tripod.com /ebarrueto1/solu2.htm