\_ Ejercicios Capitulo 2

0- Técnicas de diseño

* 1. **Explicar cuántas y cuáles son las técnicas de diseño y en que consisten cada una.**

Existen dos tipos de técnicas:

\_ **TOP DOWN**

\_ **BOTTOM UP**

**TOP DOWN** ( de arriba hacia abajo)

Se establecen niveles de mayar a menor complejidad, generando módulos de estructura jerárquica. Se implementara cada modulo y se pulirá hasta encontrar la solución al problema.

**BOTTOM UP**

En esta técnica los problemas se van solucionando a medida que van apareciendo, se puede dar el caso que haya algún software pre hecho y se adapte para dar solución al nuevo sistema.

* 1. **Cuáles son las diferencias entre estas dos técnicas, cuál es la más apropiada para el paradigma estructurado.**

Suponiendo que la empresa en la que se implementara el sistema esta organizada en diferentes sectores como ser Compras, Ventas, Marketing, etc.

Con la técnica de **Bottom Up** se desarrollara el software a medida que se vayan presentando los problemas de cada sector. El problema es si la empresa quiere integrar varios sectores, ya que al ser desarrollados por separado no se puedan adaptar entre ellos.

La técnica mas apropiada para el paradigma estructurado es la de **Top Down** ya queel problema de integrar varios sectores no ocurriría porque que se hubiera realizado un análisis global donde se analizarían todos los problemas, desarrollando todas las funcionalidades compatibles entre si.

## 1- Pseudocódigo

* 1. **Que es el pseudocódigo**

El pseudocódigo es un lenguaje de especificación de algoritmos que se utiliza dentro de la programación estructurada combinando el lenguaje de programación con el lenguaje del programador, en nuestro caso el castellano. Este narra los pasos de programación del algoritmo explicando con palabras lo que el programa va a realizar, el pseudocódigo es independiente del lenguaje de programación.

2-Programación Modular

* 1. **En qué consiste la programación modular**

La programación modular se utiliza con la técnica de TOP DOWN ya que debe ser solucionado a través de módulos. Un modulo es un conjunto de instrucciones que manejan variables para solucionar los pasos del problema.

* 1. **Cuantos tipos de módulos hay que diferencias tienen entre sí.**

|  |  |  |
| --- | --- | --- |
| **MODULOS** | **PROCEDIMIENTOS** | **Realizan operaciones sin devolver ningún valor** |
| **FUNCIONES** | **Realizan operaciones y devuelven un valor** |

* 1. **A cuál de estos tipos de módulos corresponde un tipo void.**

El tipo void (vacío) corresponde al tipo de modulo de procedimiento ya que no devuelve ningún valor.

* 1. **En qué consiste la notación Camello Mayúsculas para etiquetar un módulo.**

La notación Camello Mayúscula consiste en unir las palabras sin acentos, separadas con la primera letra de cada palabra en mayúsculas y el resto en minúscula.

Por ejemplo el modulo ALUMNO PRIMER AÑO se escribirá AlumnoPrimerAño.

* 1. **Plantear un problema y dar un Ejemplo distinto del dado.**

Problema: Ir a jugar al fútbol.

INICIO

IrALaCancha

PonerseLosBotines

JugarAlFutbol

Ducharse

VolverACasa

FIN

## 3-Estructuras Algorítmicas

* 1. **Definir en qué consiste una estructura algorítmica o Acción.**

Son las acciones que permiten el manejo de variables para la solución del problema.

* 1. **Cuántas y cuáles son las estructuras algorítmicas.**

Las acciones algorítmicas son:

|  |  |  |
| --- | --- | --- |
| **ACCIONES** | **SECUENCIALES** | **ASIGNACIÓN** |
| **ENTRADA** |
| **SALIDA** |
| **SELECCIÓN** | **SI(SIMPLE)** |
| **SI SINO (DOBLE)** |
| **SELECCIONAR (MULTIPLE)** |
| **ITERACIÓN** | **FOR (PARA)** |
| **WHILE (MIENTRAS)** |
| **DO WHILE (REPETIR HASTA)** |

### 3.1-Estructura algorítmica de Secuencia

* + 1. **Definir qué es una estructura algorítmica de secuencia, dar un ejemplo modular.**

La estructura algorítmica de secuencia es una serie de instrucciones (secuencia), en pseudocódigo cada acción sigue a continuación de la otra y se representa así:

Problema: Hacer mate

INICIO

Pongo el agua a calentar.

Lleno el mate de yerba.

Pongo agua caliente en el mate.

Chupo la bombilla y tomo el mate.

FIN

* + 1. **Cuántas y cuáles son las acciones de secuencia, en qué consiste cada una de ella.**

|  |  |  |
| --- | --- | --- |
| **ACCIONES DE SECUENCIA** | **SALIDA (IMPRIMIR)** | **Mostrara en pantalla un mensaje** |
| **LECTURA (LEER)** | **Lee del teclado un valor** |
| **ASIGNACION (=)** | **Guarda un valor en la memoria de una variable** |

* + 1. **Dar un ejemplo utilizando los pasos de la resolución de un algoritmo en computadora, distinto al dado en la teoría.**

**VARIABLES**

double SueldoMes, RemuneracionAnual;

**INICIO**

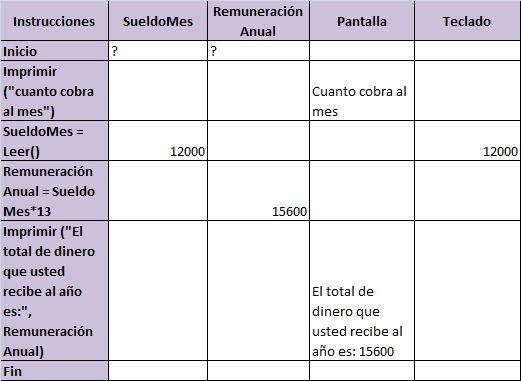
**IMPRIMIR**(“Cuanto cobra por mes”); **Acción Imprimir**

SueldoMes=**LEER(); Acción Leer**

RemuneracionAnual=SueldoMes\*13; **Acción Asignación**

**IMPRIMIR**(“El total de dinero que usted recibe al año es”,RemuneracionAnual);

**FIN**

****

* + 1. **Utilizando los pasos de resolución de un algoritmo resolver los siguientes problemas dada su definición.**
       1. **Estando en tiempos de navidad un supermercado ofrece un 10% de descuento del precio de góndola, una persona compra dos productos y quiere saber cuánto tiene que pagar.**

**VARIABLES**

Int Product1, Product2, Descuent, TotalPagar;

**INICIO**

/\* Se le pide al usuario que ingrese el valor del producto\*/

**IMPRIMIR(“Ingrese el valor del primer producto”);**

/\*Se guarda el valor del producto en Product1\*/

**Product1=LEER();**

/\* Se le pide al usuario que ingrese el valor del producto\*/

**IMPRIMIR(“Ingrese el valor del segundo producto”);**

**/**\*Se guarda el valor del producto en Product2\*/

**Product2=LEER();**

/\*Se realiza la operación para saber que descuento tendrá el cliente y se guarda en Descuent\*/

**Descuent=(Product1+Product2)\*0.1;**

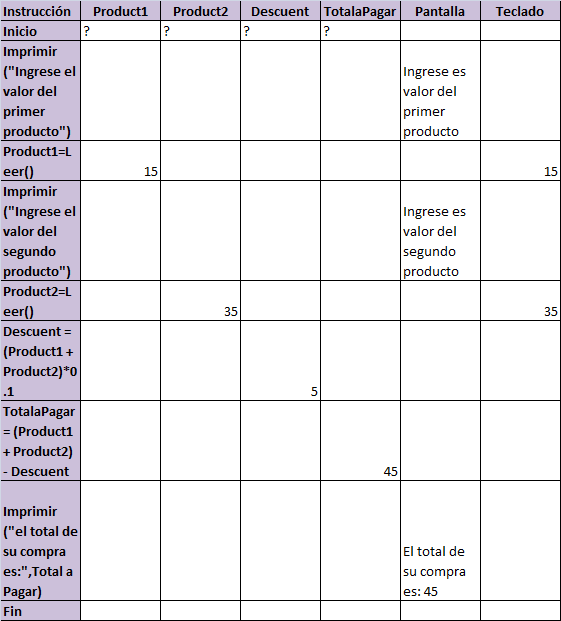
/\*Se realiza la operación para saber cuánto deberá abonar el cliente\*/

**TotalPagar=(Product1+Product2)-Descuent;**

/\*Se imprime en pantalla el total a pagar por el cliente\*/

**IMPRIMIR(“El total de su compra es:”,TotalPagar);**

**FIN**



* + - 1. **Una empresa tiene dos empleados Ingresar el sueldo de cada uno e imprimir el promedio, cual es el monto a pagar por la empresa en el concepto de sueldos.**

**VARIABLES**

int SueldoEmp1, SueldoEmp2, TotalSueld, Promedio;

**INICIO**

IMPRIMIR(“Ingrese el sueldo del primer empleado”);

SueldoEmp1=LEER();

IMPRIMIR(“Ingrese el sueldo del segundo empleado”);

SueldoEmp2=LEER();

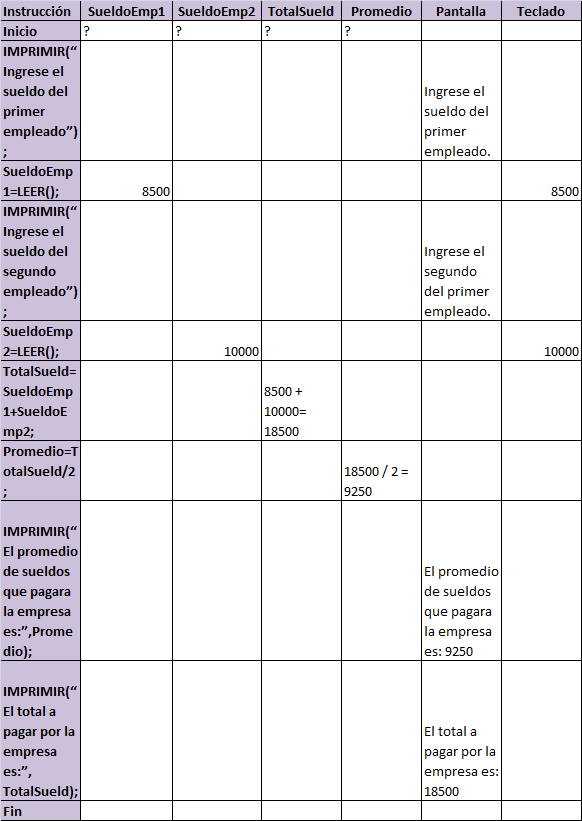
TotalSueld=SueldoEmp1+SueldoEmp2;

Promedio=TotalSueld/2;

IMPRIMIR(“El promedio de sueldos que pagara la empresa es:”,Promedio);

IMPRIMIR(“El total a pagar por la empresa es:”, TotalSueld);

**FIN**

****

* + - 1. **Ingresando el radio de una circunferencia, visualizar el Perímetro.**

**CONSTANTES**

PI = 3.1416

**VARIABLES**

double Radio, Perimetro;

**INICIO**

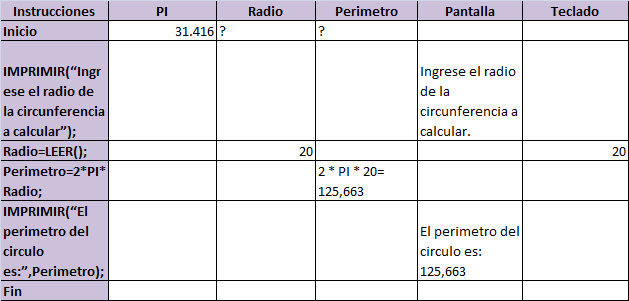
IMPRIMIR(“Ingrese el radio de la circunferencia a calcular”);

Radio=LEER();

Perimetro=2\*PI\*Radio;

IMPRIMIR(“El perimetro del circulo es:”,Perimetro);

**FIN**



### 3.2 Estructura algorítmica de selección simple.

**3.2.1 Explicar que es una acción de selección.**

La acción de selección consiste en analizar los problemas cuando se da un algoritmo **que pasa si**. Esto se produce porque no todos los problemas pueden tener una respuesta.

**3.2.2** **Dar un ejemplo de programación modular de un algoritmo que utilice la acción de selección.**

En el primer ejemplo se ve un problema que no necesita acción de selección.

Problema: Pinchar una rueda

**INICIO**

Voy conduciendo mi coche

Pincho una rueda

Saco la rueda de auxilio

Cambio la rueda

Continuo con mi viaje

**FIN**

En este caso no hace falta ninguna pregunta del tipo que pasa si. Pero qué pasa si la rueda de auxilio esta pinchada. El ejemplo será el siguiente:

Problema: Pinchar una rueda y tener la rueda de auxilio pinchada.

**INICIO**

Voy conduciendo mi coche

Pincho una rueda

**SI** (La rueda de auxilio esta pinchada)

Camino hasta una gomeria

Emparcho la rueda de auxilio

Vuelvo al vehículo

**FIN SI**

Cambio la rueda

Continuo con mi viaje

**FIN**

**3.2.3**  **Utilizando los pasos de resolución de un algoritmo resolver los siguientes problemas dada su definición.**

**3.2.3.1 IMPUESTO A LAS GANANCIAS La AFIP Considera un mínimo no imponible al ingreso mensual de 1080 $/mes, más un valor de deducción especial de 5184$/mes más 1200 por cónyuge por mes , más 600 por hijo por mes, más 450$ /mes por familiar a cargo.**

**Ejemplo:**

**El mínimo no imponible de una persona con esposa y dos hijos con la madre a cargo tiene un mínimo imponible de 1080+5184+1200+2\*600+450=9114**

**Si la persona gana más de 9114 entra a pagar ganancias Construir un programa que ingresando los datos del contribuyente y el ingreso mensual determine si es pasible de pagar ingresos brutos.**

**VARIABLES**

Int Sueldo, ConyugueACargo, Hijos, FamilACargo, MinNoImp,

String Conyugue;

**INICIO**

IMPRIMIR(“Cuanto gana por mes”);

Sueldo=LEER();

IMPRIMIR(“Tiene conyugue”);

Conyugue=LEER();

SI (Conyugue == “si”) .

ConyugueACargo=1;

SI NO

ConyugueACargo=0;

FIN SI

IMPRIMIR(“Cuantos hijos tiene?”);

Hijos=LEER();

IMPRIMIR(“Cuantos familiares a cargo tiene”);

FamilACargo=LEER();

MinNoImp=1080+5184+(1200\*ConyugueACargo)+(600\*Hijos)+(FamilACargo\*450);

SI Sueldo > MinNoImp

IMPRIMIR(“Usted debe pagar ingresos brutos”);

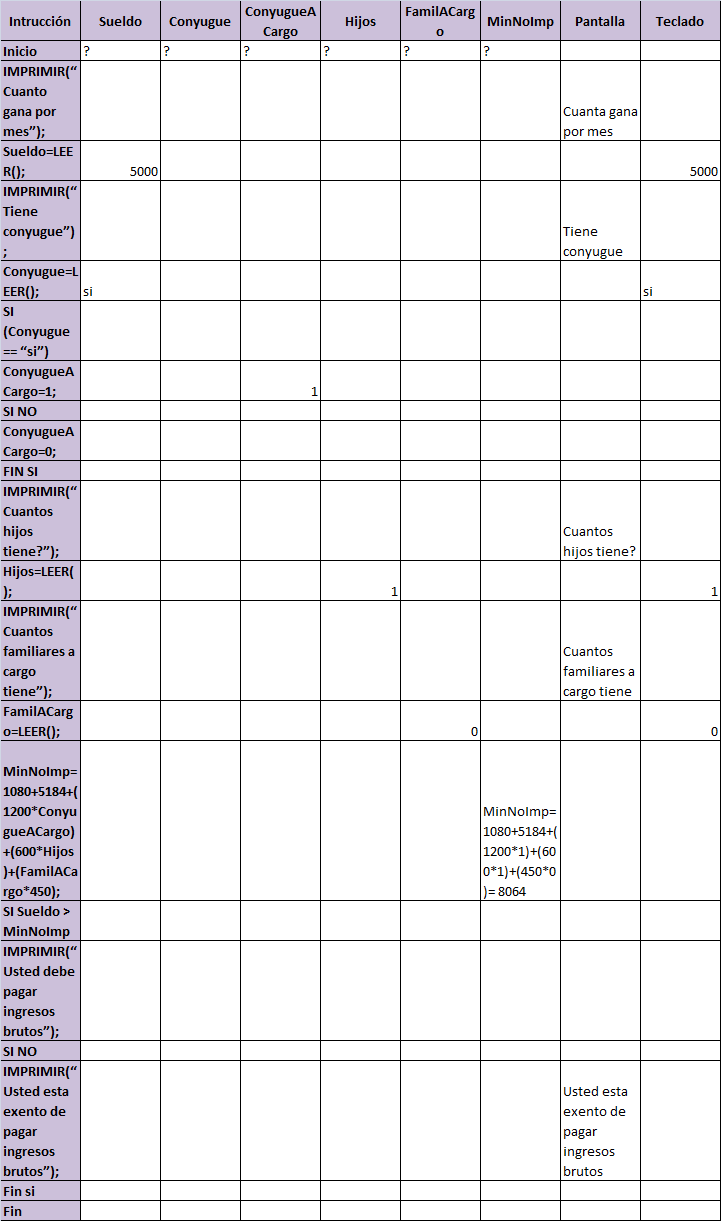
SI NO

IMPRIMIR(“Usted esta exento de pagar ingresos brutos”);

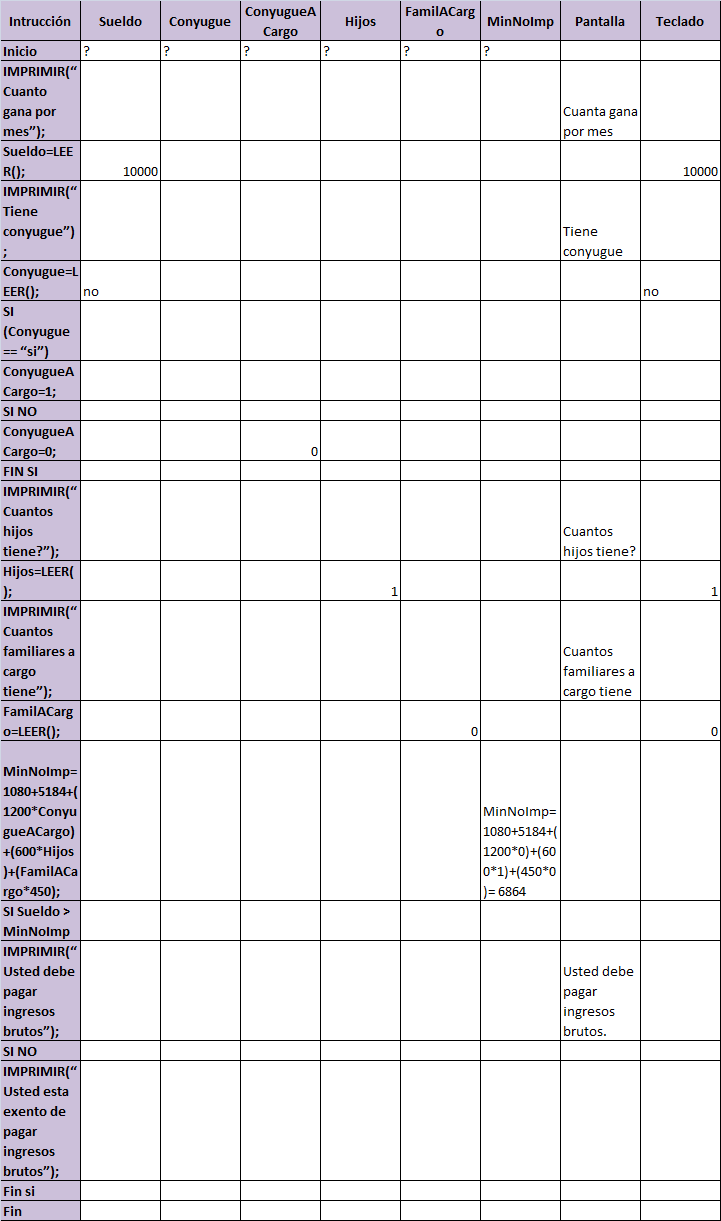
FIN SI

FIN

Ejemplo en el caso que el contribuyente no pague ingresos brutos.



Ejemplo para el caso en el que el contribuyente tenga que pagar ingresos brutos.

****

**3.2.3.2 DESCUENTOS POR COMPRAS El supermercado "el chino" resuelve hacer durante esta semana un descuento especial Si el Cliente hace una compra por mas de 500$ le hará un descuento del 20% sobre lo que facture mas de 500$.**

**POR EJEMPLO**

**si el cliente compra 200$ debe pagar 200$**

**si el cliente Compra 600$ (100$ más de 500$) tendrá un descuento de 100\*20%=20 por lo que pagará 600-20= 580$**

**Hacer un algoritmo que calcule cuanto debe pagar un clienta dado un monto de compra.**

VARIABLES

double Compra, Desc, TotalApagar;

INICIO

Imprimir(“Ingrese el valor de la compra”);

Compra=LEER();

SI Compra > 500

Desc = (Compra – 500) \* 0.2;

SI NO

Desc = 0;

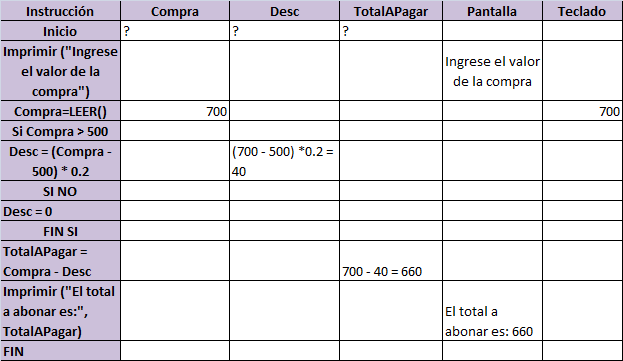
FIN SI

TotalAPagar=Compra – Desc;

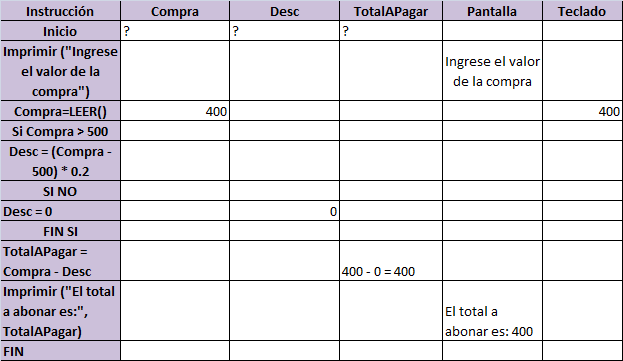
Imprimir(“El total a abonar es:”, TotalAPagar);

FIN

Ejemplo en prueba de escritorio para el caso en que la compra es superior a 500

****

Ejemplo en prueba de escritorio para el caso en que la compra es inferior a 500

****

**3.3 Estructura algorítmica de selección doble.**

**3.3.1 Que analiza la selección doble.**

La selección doble analiza los problemas en los que hay dos posibilidades en los que pasa SI y en los que pasa SI NO.

**3.3.2 Dar un ejemplo modular de un algoritmo que de selección doble.**

Problema: Despertarse para ir a trabajar.

INICIO

Suena el despertador

SI (Me despierto rápido y me levanto)

Tomo el colectivo

SI NO

Tomo un taxi

FIN SI

IMPRIMIR(“Fin del programa”)

FIN

* + 1. **Utilizando los pasos de resolución de un algoritmo resolver los siguientes problemas dada su definición.**
       1. **En una Clínica para la atención al paciente Hay dos colas el encargado envía a los pacientes femeninos a la cola de la derecha, y a los pacientes masculinos a la cola de la izquierda, idear un algoritmo que tras consultar el sexo del paciente envíe al paciente a la cola correspondiente.**

**VARIABLES**

String Sexo;

**INICIO**

Imprimir (“Ingrese su sexo F femenino y M masculino”);

Sexo = Leer();

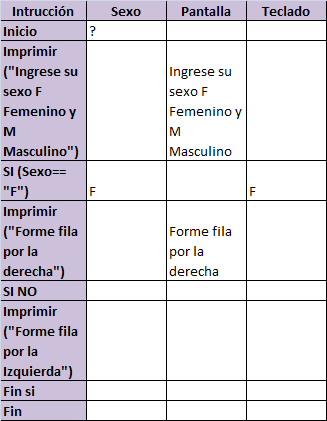
SI (Sexo == “F”)

Imprimir(“Forme fila por la derecha”);

SI NO

Imprimir (“Forme fila por la izquierda”)

FIN SI

****

* + - 1. **La división entre dos números sólo se puede hacer si el divisor es distinto de cero ,Ingresar dos números dividendo y divisor y en caso que se pueda hacer emitir el resultado y si no emitir un mensaje indicando que no se puede hacer y porqué.**

VARIABLES

Int Dividendo, Divisor, Resultado;

INICIO

Imprimir (“Ingrese el dividendo de la operación”);

Dividendo = LEER();

Imprimir (“Ingrese el divisor de la operación”);

Divisor = LEER();

SI (Divisor == 0)

Imprimir (“ La operación no se puede realizar porque el divisor es 0”);

SI NO

Resultado = Dividendo / Divisor;

Imprimir (“El resultado de la operación es:”, Resultado);

FIN SI

FIN

Prueba de escritorio para el caso en el que el divisor es distinto de 0.



Prueba de escritorio para el casa en el que el divisor es igual a 0.



**3.3.3.3 Ingresar un número entero y determinar si es par o impar emitiendo un mensaje correspondiente.**

Ayuda: un número es par si es divisible por dos es decir el resto de dividir por dos es igual a cero.

VARIABLES

Int Num, Result;

INICIO

Imprimir (“Ingrese un numero”);

Num = LEER();

Result = Num % 2;

SI (Result == 0)

Imprimir (“El numero ingresado es par”);

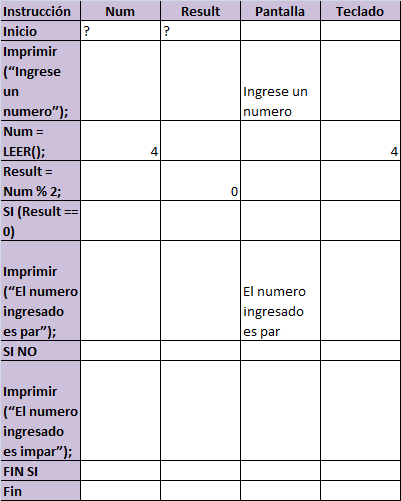
SI NO

Imprimir (“El numero ingresado es impar”);

FIN SI

FIN

Prueba de escritorio para el caso en el que el numero ingresado es par.



Prueba de escritorio para el caso en el que el numero sea impar.



#### 3.4 Estructura algorítmica de selecciones anidadas

* + 1. **Utilizando los pasos de resolución de un algoritmo resolver los siguientes problemas dada su definición.**
       1. **SUPERMERCADOS CARRETON**

Este supermercado celebra el aniversario el 30 de Enero Cuando el cliente hace una compra , se le pide el documento , si el cliente nació el 30 de Enero se le descuenta el 30% de su compra, en cambio si nació en el mismo mes del aniversario se le descuenta el 10% de su compra ,si no nació en el mismo mes deberá pagar sin descuento.

Hacer un algoritmo que ingresando el monto de una compra y el día y mes del nacimiento, indique el monto a pagar.

VARIABLES

double Compra, totPag;

int Mes, Dia;

INICIO

Imprimir (“Ingrese el monto de la compra”);

Compra = LEER();

Imprimir (“Ingrese el mes de su cumpleaños siendo enero, 2 febrero, etc”);

Mes = LEER();

Imprimir (“Ingrese el día de su cumpleaños”);

Dia = LEER();

SI (Mes == 1)

SI (Dia == 30)

TotPag = Compra - ( Compra \* 0.3);

SI NO

TotPag = Compra – (Compra \* 0.1);

FIN SI

SI NO

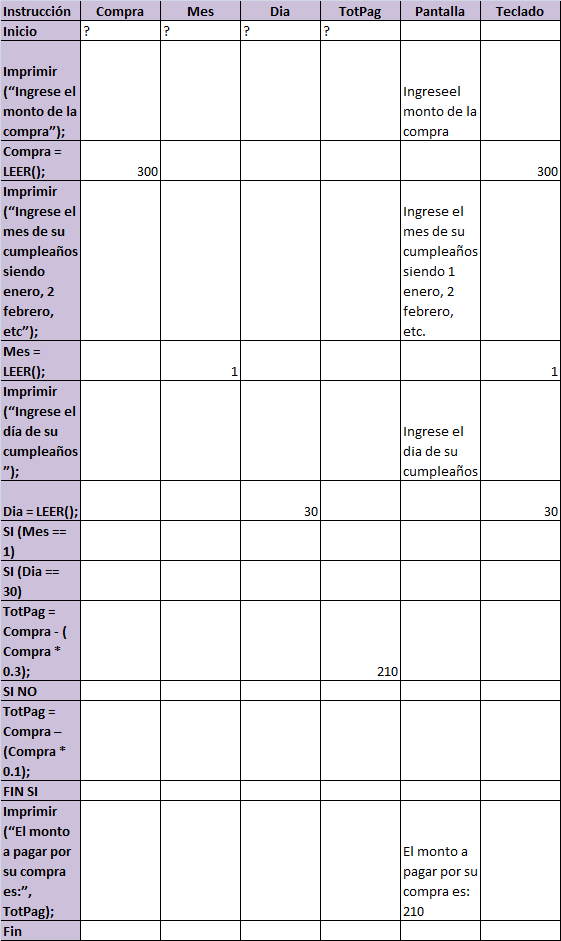
TotPag = Compra;

FIN SI

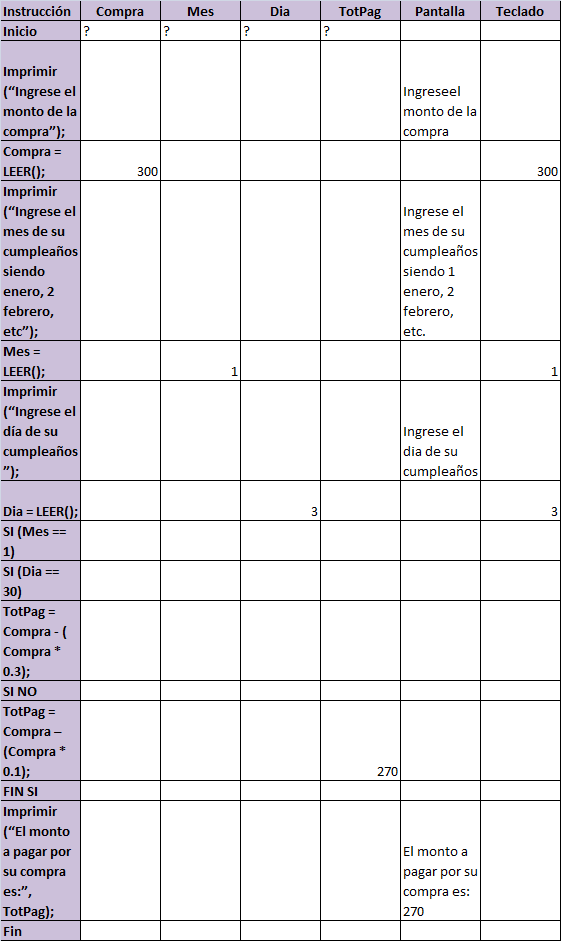
Imprimir (“El monto a pagar por su compra es:”, TotPag);

FIN

Prueba de escritorio para el caso en el que el cliente nació el 30/1.



Prueba de escritorio para el caso en que el cliente nació el 3/01.



* + - 1. **SUPERDESCUENTOS EN ANIVERSARIO PATRIO EN ROTISERÍA EL GAUCHITO**

La rotisería EL GAUCHITO ha decidido el día de la patria hacer descuento para los productos que se compren en ese día.

10% para los lácteos

15% para pollos

18% para carne porcina

20% para carne vacuna

30% para embutidos

Hacer un programa que calcule el descuento de un cliente si compra dos productos ingresando para cada uno el precio sin descuento y el tipo de producto.

VARIABLES

double ValProduct1, ValProduct2, DescProduct1, DescProduct2, DescTot;

string TipoProduct1, TipoProduct2;

INICIO

Imprimir(“Ingrese el precio del primer producto”);

ValProduct1 = LEER();

Imprimir(“Ingrese el tipo de producto”);

TipoProduct1 = LEER();

Imprimir(“Ingrese el precio del segundo producto”);

ValProduct2 = LEER();

Imprimir(“Ingrese el tipo de producto”);

TipoProduct2 = LEER();

SI (TipoProduct1 == “lacteos”)

DescProduct1 = (ValProduct1 \* 0.1);

SI NO

SI (TipoProduct1 == “pollo”)

DescProduct1 = (ValProduct1 \* 0.15);

SI NO

SI (TipoProduct1 == “carne porcina”)

DescProduct1 = (ValProduct1 \* 0.18);

SI NO

SI (TipoProduct1 == “carne vacuna”)

DescProduct1 = (ValProduct1 \* 0.2);

SI NO

DescProduct1 = (ValProduct1 \* 0.3);

FIN SI

FIN SI

FIN SI

FIN SI

SI (TipoProduct2 == “lacteos”)

DescProduct2 = (ValProduct2 \* 0.1);

SI NO

SI (TipoProduct2 == “pollo”)

DescProduct2 = (ValProduct2 \* 0.15);

SI NO

SI (TipoProduct2 == “carne porcina”)

DescProduct2 = (ValProduct2 \* 0.18);

SI NO

SI (TipoProduct2 == “carne vacuna”)

DescProduct2 = (ValProduct2 \* 0.2);

SI NO

DescProduct2= (ValProduct2 \* 0.3);

FIN SI

FIN SI

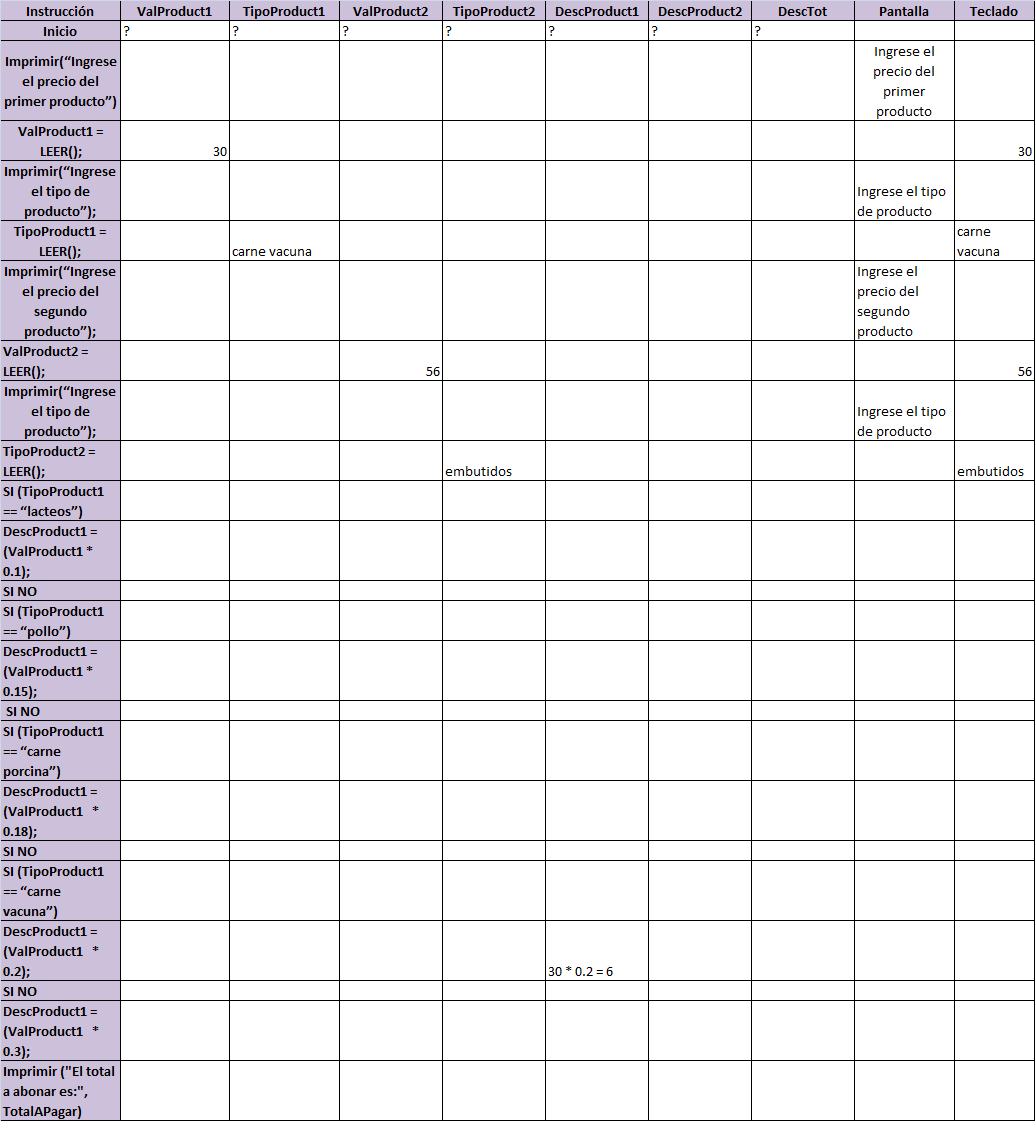
FIN SI

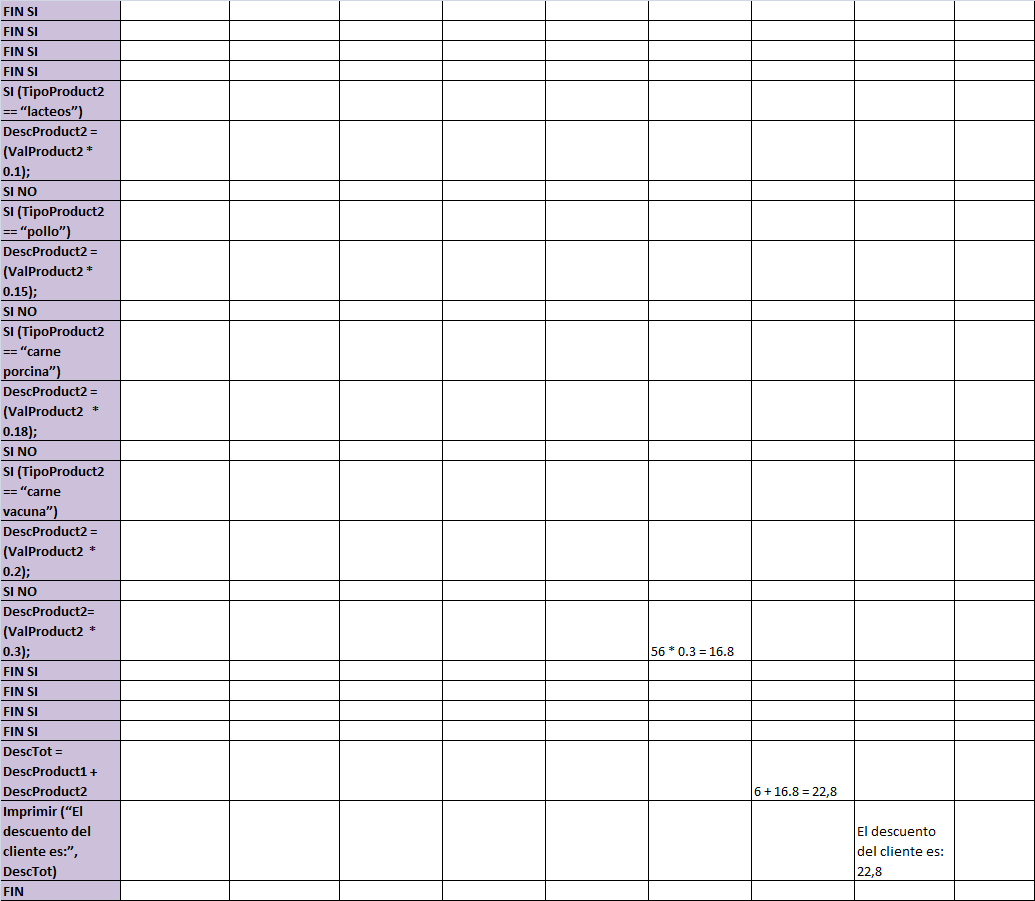
FIN SI

DescTot = DescProduct1 + DescProduct2;

Imprimir (“El descuento del cliente es:”, DescTot);

FIN





#### 3.5 Estuctura de selección múltiple

3.5.1 Utilizando los pasos de resolución de un algoritmo resolver los siguientes problemas dada su definición.

3.5.1.1 Ingresar dos números y hacer un menú que ejecute si pulsa 1 la suma ,si 2 la resta ,si 3 el producto ,si 4 la división y otro ingreso que salga del programa indicando que no se ha ingresado una opción válida.

CONSTANTE

Const string OPCIONINC = “No ha ingresado un numero del 1 al 4”;

VARIABLES

Int Num1, Num2;

String Opcion;

double Result;

INICIO

Imprimir(“Ingrese el primer numero”);

Num1 = LEER();

Imprimir(“Ingrese el segundo numero”);

Num2 = LEER();

Imprimir(“Ingrese 1 si quiere que se sumen, 2 que se resten, 3 que se multipliquen y 4 que se dividan”);

Opcion = LEER();

Seleccionar (Opcion)

En caso que sea “1”

Result = Num1 + Num2;

FIN CASO

En caso que sea “2”;

Result = Num1 – Num2;

FIN CASO

En caso que sea “3”;

Result = Num1 \* Num2;

FIN CASO

En caso que sea “4”;

Result = Num1 / Num2;

FIN CASO

Si no se cumple ninguno de los casos;

Imprimir (OPCIONINC);

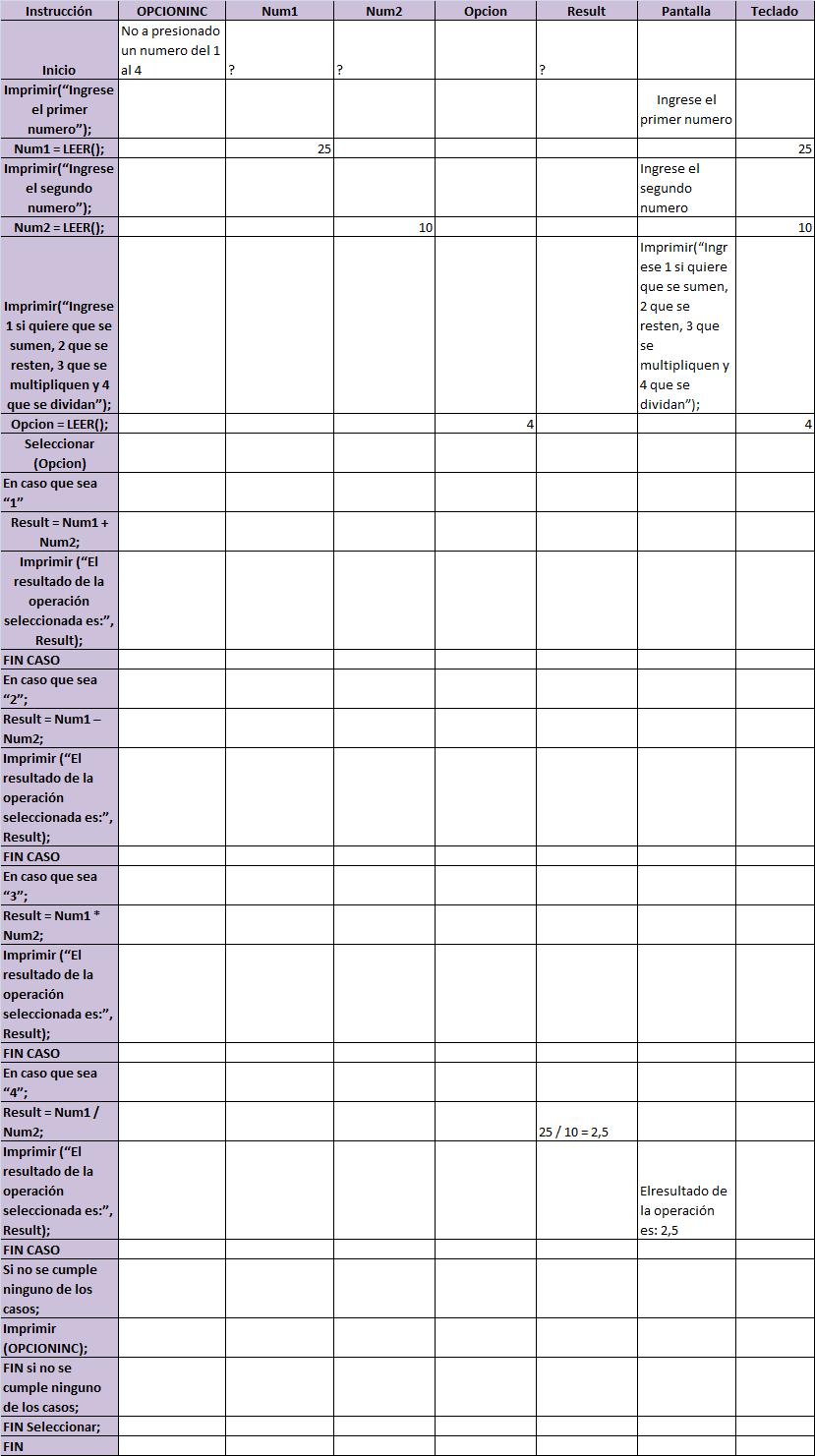
FIN si no se cumple ninguno de los casos;

FIN Seleccionar;

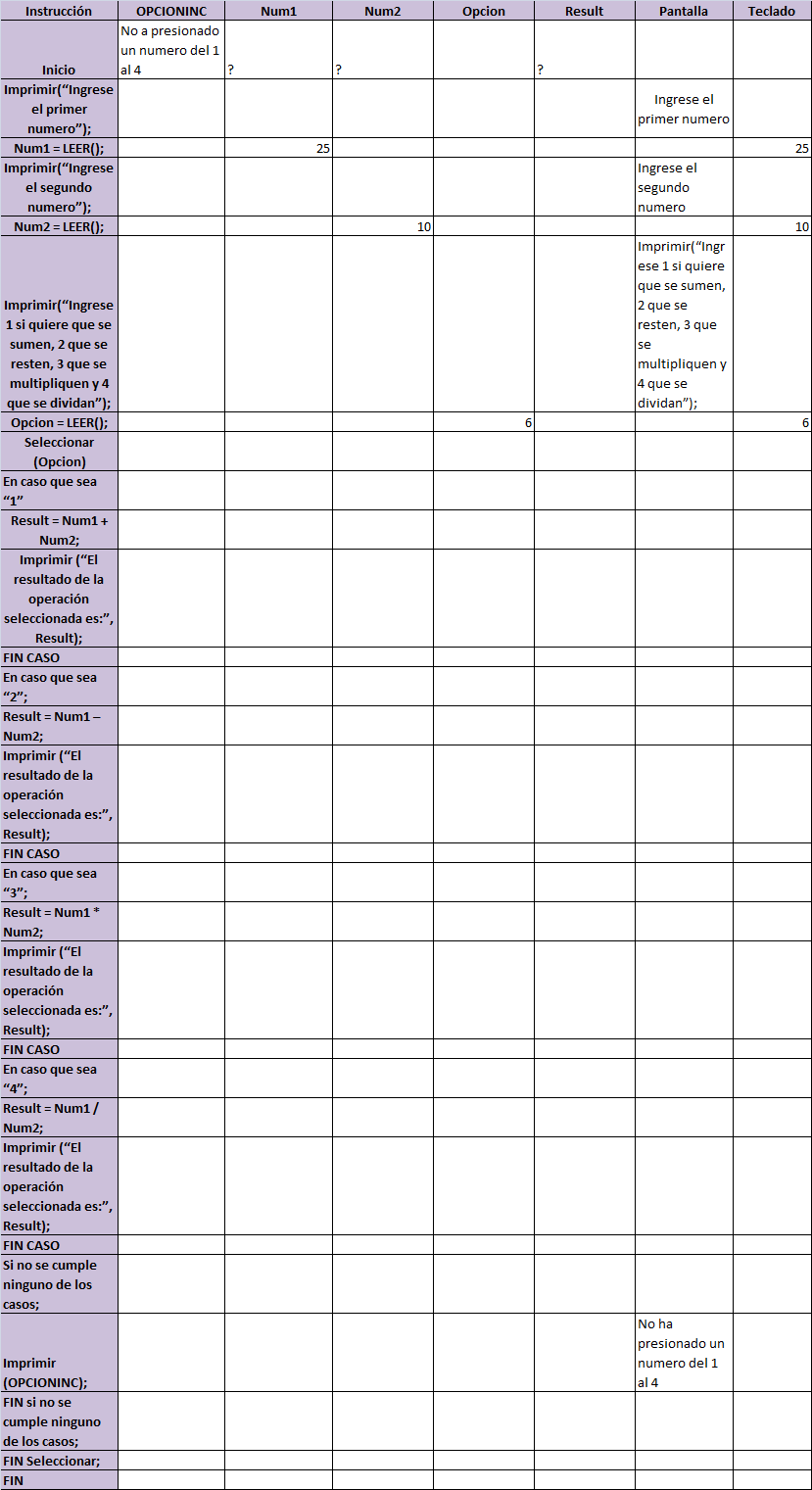
Imprimir (“El resultado de la operación seleccionada es:”, Result);

FIN

Prueba de escritorio para el caso en el que la opción elegida es la 4.



Prueba de escritorio para el caso en el que la opción ingresada no es un numero del 1 al 4.



**3.5.1.2 Escribir un programa que determine si un año es bisiesto,**

Un año es bisiesto si es múltiplo de 4 (por ejemplo 1884

Sin embargo los años múltiplos de 100 no son bisiesto por ejemplo 1900 salvo que sean múltiplos de 400 por ejemplo 2000

***Ayuda*** *:un número es divisible por 400 si el resto de dividir por 400 da cero*

*Ejemplo Numero% 400==0*

VARIABLES

int Año;

INICIO

Imprimir (“Ingrese el año que desea saber si fue bisiesto”);

Año = LEER();

SI (Año % 400 == 0)

Imprimir(“El año es bisiesto”);

SI NO

SI (Año % 100 == 0)

Imprimir(“El año no es bisiesto”);

SI NO

SI (Año % 4 == 0)

Imprimir(“El año es bisiesto”);

SI NO

Imprimir(“El año no es bisiesto”);

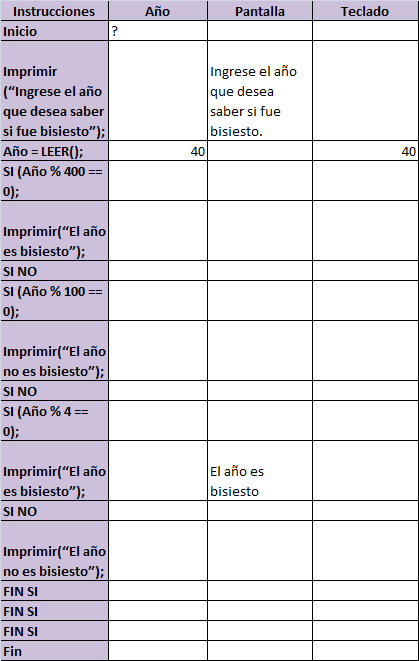
FIN SI

FIN SI

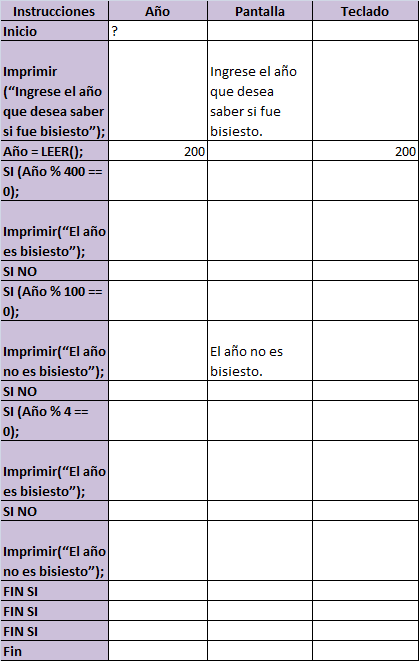
FIN SI

FIN

Prueba de escritorio en el que el año introducido es bisiesto.



Prueba de escritorio para el caso en el que el año introducido no es bisiesto.



## 4 Acciones cíclicas

**4.0.1 ¿Cuántas y cuáles son las acciones cíclicas?**

Las acciones cíclicas se utilizan cuando en un programa se realiza la misma acción x veces. Por ejemplo si pincho una rueda tengo que sacar es cuatro veces sacar una tuerca en algoritmos esta acción se la denomina iteración.

|  |  |
| --- | --- |
| ACCIONES ITERATIVAS | PARA |
| MIENTRAS |
| REPETIR MIENTRAS |

##### 4.1 Acción cíclica PARA

4.1.1 Definir la fórmula algorítmica de la acción cíclica PARA

La acción **PARA** repite las veces que se indique (N) una secuencia de acciones.

La fórmula algorítmica es:

**PARA (CONTADOR=1;CONTADOR <=N;CONTADOR++)**

**\*\*\*\***

**\*\*\*\***

**FIN PARA**

Contador es una variable numérica, en este caso se iniciara con el valor 1, luego se indica la condición a cumplir que en este caso es que si contador es menor a N se realizara el ciclo, y en el tercer lugar se indica que a la variable contador se le sumara 1 cada vez que se termine el ciclo. Ya que la variable contador cuenta la cantidad de veces que se realiza el ciclo.

4.1.2 Utilizando los pasos de resolución de un algoritmo resolver los siguientes problemas dada su definición

4.1.2.1 Escribir un programa que acepte 5 números de un usuario y devuelva el promedio

VARIABLES

double Suma, SumaTot, Contador, Prom;

INICIO

Suma = 0;

SumaTot = 0;

Contador = 0;

Prom = 0;

PARA (Contador = 1, Contador <= 5, Contador++);

Imprimir(“Ingrese un numero”);

Suma = LEER();

SumaTot = Suma + SumaTot;

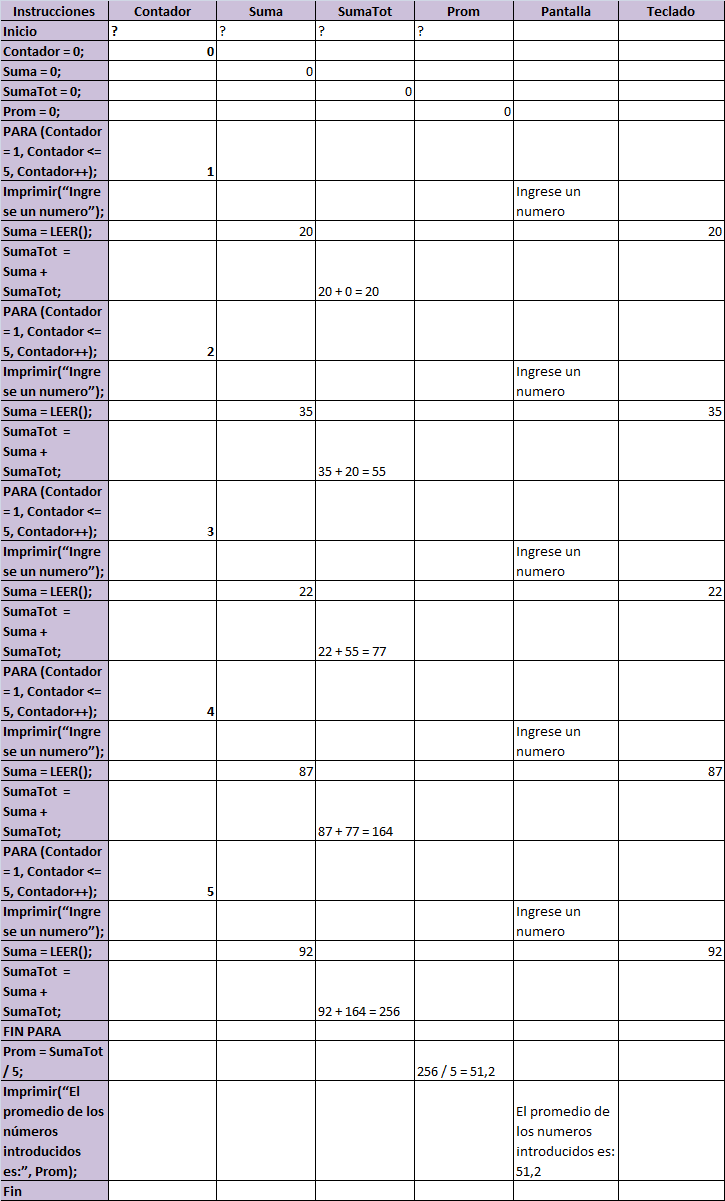
FIN PARA

Prom = SumaTot / 5;

Imprimir(“El promedio de los números introducidos es:”, Prom);

FIN

Prueba de escritorio.



4.1.2.2 Escribir un programa que visualice los números pares entre 1 y 10 incluido.

VARIABLES

int Contador;

INICIO

Contador = 0;

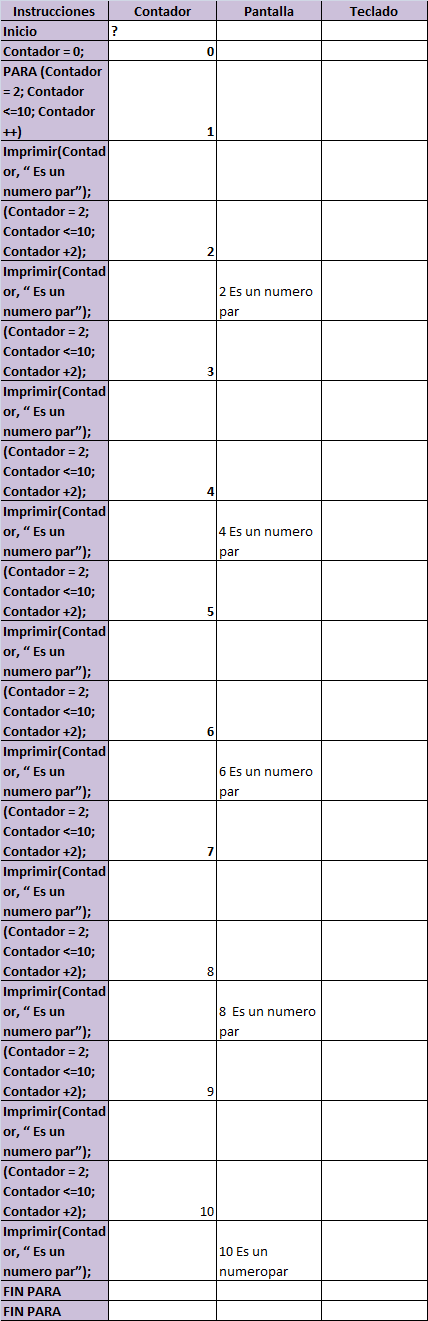
PARA (Contador = 2; Contador <=10; Contador ++)

Si(contador % 2 == 0)

Imprimir(Contador, “ Es un numero par”);

FIN PARA

FIN



#### 4.2 Acción cíclica MIENTRAS y HACER MIENTRAS

4.2.0 Definir la fórmula algorítmica de la acción cíclica MIENTRAS

La fórmula algorítmica es:

**MIENTRAS(CONDICION)**

**\*\*\*\*\*\*\***

**\*\*\*\*\*\*\***

**FIN MIENTRAS**

La CONDICION es una variable o una expresión que devuelve un valor booleano y realiza el ciclo si su valor es VERDADERO.

4.2.1. Definir la fórmula algorítmica de la acción cíclica de HACER MIENTRAS.

REPETIR MIENTRAS entra por lo menos una vez en el ciclo y sale cuando la condición del MIENTRAS es FALSO

**La fórmula algorítmica es:**

**REPETIR**

**\*\*\*\***

**\*\*\*\***

**MIENTRAS(CONDICION)**

4.2.2 Utilizando los pasos de resolución de un algoritmo resolver los siguientes problemas dada su definición.

4.2.2.1 Construir un programa que permita leer números positivos sumarlos y contar cuantos números se han introducido, En este programa se establecerá un valor CENTINELA que detecte el final de entrada de datos y que naturalmente no puede ser un dato válido, en este caso como los números son positivos el CENTINELA puede se un número negativo.

VARIABLES

double SumaTot, Contador;

int CantNum;

INICIO

SumaTot = 0;

CantNum = 0;

Imprimir(“Ingrese un numero, si el numero es negativo se saldrá del programa”);

Contador = LEER();

MIENTRAS (Contador > 0);

CantNum = ++;

Sumatot = Contador + SumaTot;

Imprimir(“Ingrese un numero”);

Contador = LEER();

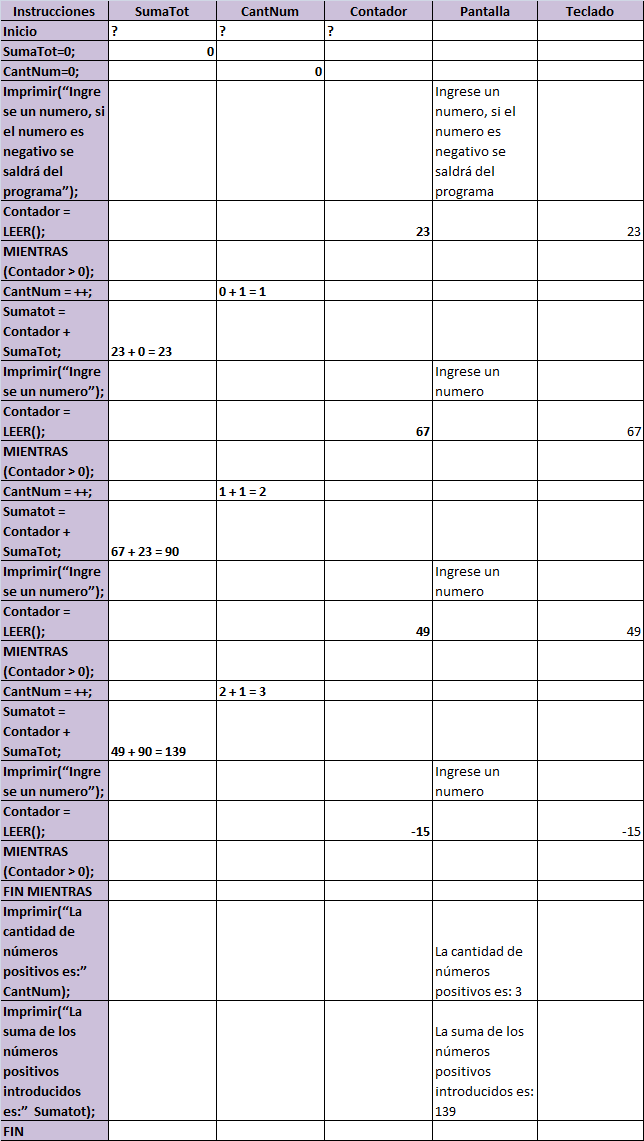
FIN MIENTRAS

Imprimir(“La cantidad de números positivos es:” CantNum);

Imprimir(“La suma de los números positivos introducidos es:” Sumatot);

FIN

Prueba de escritorio.



4.2.3 Utilizando los pasos de resolución de un algoritmo sin hacer prueba escritorio resolver el siguiente problema dada su definición.

4.2.3.1 El matemático Italiano Leonardo Fibonacci propuso el siguiente problema: Suponiendo que una pareja de conejos paren una pareja cada mes que se hace fértil al mes siguiente y no muere ningún conejo durante un año, programar el algoritmo que calcule cuantas parejas habrá al final del año.

VARIABLES

int Mes, MesAnt, MesAntAnt, MesAct;

INICIO

Mes = 1;

MesAnt = 1;

MesAntAnt = 0;

MIENTRAS ( Mes <=12);

MesAct = MesAnt + MesAntAnt;

MesAntAnt = MesAnt;

MesAnt = MesAct;

Mes ++;

FIN MIENTRAS

Imprimir(“La cantidad de parejas al final del año será de:” MesAct);

FIN

4.2.3.2 Determinar un algoritmo que dada una pareja de conejos, calcule la cantidad de meses que se necesita para obtener N parejas sin que se muera ningún conejo.

VARIABLES

int ParejNecesarias, ParAnt,ParAntAnt, Mes, ParAct;

INICIO

Imprimir(“Ingrese la cantidad de parejas que desea consultar”);

ParejNecesarias = LEER();

ParAnt = 1;

ParAntAnt = 0;

Mes = 1;

ParAct = 1

MIENTRAS (ParAct < ParejNecesarias);

ParAct = ParAnt + ParAntAnt;

ParAntAnt = ParAnt;

ParAnt = ParAct;

Mes ++;

FIN MIENTRAS

Imprimir(“Los meses que se necesitaran para alcanzar las parejas deseadas son:”, Mes);

FIN

### 5 Problemas de Máximos y Mínimos

5.1 En que se basan los métodos para hallar un máximo o un mínimo de los datos ingresados en forma iterativa.

.

El método será tener una variable X que en cada ciclo compare con el valor ingresado y si este es mayor o menor, según lo que se busque, se le asigna el valor.

5.2 ¿En qué consiste el problema principal para asignar un máximo o un

Mínimo ?.

El problema radica en que a las variables que se utilizan para comparar ya sea Máximo o Mínimo, se les debe asignar un valor inicial. Y este debe ser menor que los valores a leer en el caso del Máximo y mayor en el caso del Mínimo.

5.3 ¿Cuales son las dos posibilidades para asignar inicialmente un máximo o un mínimo?

A\_ Que en el caso de Máximo tenga un valor menor a los valores que se van a leer, si los valores a leer son todos positivos se le puede asignar un valor negativo, para asegurarse que el máximo siempre será uno de los valores introducidos.

B\_ Asignarle a la variable el valor de la primera lectura y luego continúe el ciclo.

5.3 Si el problema consiste en ingresar edades de personas, ¿cuál de de las dos formas elijo para calcular la edad de la más vieja?

El valor Máximo.

5.4 Codificar los cuatro ejemplos de la teoría para el caso del Mínimo.

EJEMPLO 1:(Utilizando PARA)

CONSTANTE

const int CANTIDADALEER=10;

VARIABLES

int Minimo, Contador,Valor;

INICIO

PARA(Contador=0;Contador<CANTIDADALEER;Contador ++);

IMPRIMIR("INGRESE UN VALOR");

Valor=LEER();

SI(Contador==0);

Minimo=Valor;

SI NO

SI(Minimo>Valor);

Minimo=Valor;

FIN SI

FIN SI

FIN PARA

IMPRIMIR("El Mínimo leído es:",Minimo);

FIN

EJEMPLO 2:(Utilizando MIENTRAS)

CONSTANTE

const int CANTIDADALEER=10;

VARIABLES

int Minimo,Contador,Valor;

INICIO

Contador=0;

MIENTRAS(Contador<CANTIDADALEER);

IMPRIMIR("INGRESE UN VALOR");

Valor=LEER();

SI(Contador==0);

Minimo=Valor;

SI NO

SI(Minimo>Valor);

Minimo=Valor;

FIN SI

FIN SI

Contador++;

FIN MIENTRAS

IMPRIMIR("El Mínimo leído es:",Minimo);

FIN

EJEMPLO 3:(Utilizando MIENTRAS lectura previa)

CONSTANTE

const int CANTIDADALEER=10;

VARIABLES

int Minimo , Contador ,Valor

INICIO

Contador=1;

IMPRIMIR("INGRESE UN VALOR");

Valor=LEER();

Minimo=Valor;

MIENTRAS(Contador<CANTIDADALEER);

IMPRIMIR("INGRESE UN VALOR");

Valor=LEER();

SI(Minimo>Valor);

Minimo=Valor;

FIN SI

Contador++;

FIN MIENTRAS

IMPRIMIR("El Mínimo leído es:",Minimo);

FIN

EJEMPLO 4:(Utilizando REPETIR MIENTRAS )

CONSTANTE

const int CANTIDADALEER=10;

VARIABLES

int Minimo,Contador,Valor;

INICIO

Contador=0;

REPETIR

IMPRIMIR("INGRESE UN VALOR");

Valor=LEER();

SI(Contador==0);

Minimo=Valor;

SI NO

SI(Minimo>Valor);

Minimo=Valor;

FIN SI

FIN SI

Contador++;

MIENTRAS(Contador<CANTIDADALEER);

IMPRIMIR("El Mínimo leído es:",Minimo);

FIN

5.5 Utilizando los pasos de resolución de un algoritmo resolver los siguientes problemas dada su definición.

5.5.1 Una estación meteorológica toma diariamente las lluvias caídas y hace un reporte semanal de cual fué el día de máxima precipitación, y cual el de mínima , indicando cuántos milímetros cayeron en cada caso, construir un programa que realice ese trabajo.

CONSTANTES

Contante int CANTIDADALEER = 7;

VARIABLES

Int Contador, DiaMin, DiaMax;

double Valor, Maximo, Minimo;

INICIO

Contador = 1;

Mientras ( Contador < CANTIDADALEER);

Imprimir(“Ingrese el valor de precipitación del dia”, Contador);

Valor = LEER();

SI(Contador ==1)

Minimo = Valor;

DiaMin = Contador;

Maximo = Valor;

DiaMax = Contador;

SI NO

SI (Minimo > Valor)

Minimo = Valor;

DiaMin = Contador;

FIN SI

SI (Maximo < Valor)

Maximo = Valor;

DiaMax = Contador;

FIN SI

FIN SI

Contador ++;

FIN MIENTRAS

Imprimir(“El día con menos precipitaciones es el día:” DiaMin);

Imprimir (“Las precipitaciones mínimas fueron de:”, Minimo);

Imprimir(“El día con mas precipitaciones es el día:” DiaMax);

Imprimir(“Las precipitaciones máximas fueron de:”, Maximo);

FIN

5.5.2 Modificar el programa del ejercicio anterior indicando el total de la lluvia caída durante la semana y el promedio diario de precipitación .

VARIABLES

int Contador;

double Valor, PrecTot, Promedio;

INICIO

PrecTot = 0;

PARA(Contador = 1; Contador <=7; Contador ++);

Imprimir(“Ingrese el valor de precipitación del día” Contador);

Valor = LEER();

PrecTot = Valor + PrecTot;

FIN PARA

PromedioDia = PrecTot / 7;

Imprimir (“Las precipitaciones totales de la semana son:”, PrecTot);

Imprimir (“El promedio diario de precipitaciones es:” PromedioDia);

FIN