

INSTITUTO POLITÉCNICO NACIONAL
SECRETARIA ACADÉMICA
DIRECCIÓN DE EDUCACION MEDIA SUPERIOR
CENTRO DE ESTUDIOS CIENTÍFICOS Y TECNOLÓGICOS No. 13
"RICARDO FLORES MAGÓN"

GUÍA

**de estudio para
presentar ETS de la
UNIDAD DE APRENDIZAJE
Herramientas de Programación
TURNO Vespertino
Semestre 26-2**

Presidente de academia: Carlos Francisco Arteaga Rodríguez

Fecha de Elaboración: JUNIO 2026

Área:	Nombre de la Unidad de Aprendizaje:	Nivel/semestre:
Tecnológica	Herramientas de programación	Tercer Semestre

Instrucciones generales de la guía:

Contesta la guía y apoyate en las bibliografía correspondiente

Presentación:

La escritura de programas de computadora o simplemente programación implica que la persona que decida involucrarse en esta actividad debería estar capacitado para realizar lo siguiente:

Diseño de algoritmos para la solución de problemas computacionales a través de las metodologías, procedimientos y herramientas de programación.

Las habilidades y actitudes concretas que debería desarrollar serían:

- Capacidad de análisis
- Dominio de las herramientas de programación
- Creativo e innovador
- Puntualidad y Responsabilidad
- Liderazgo
- Hábitos personales.
- Interés por el estudio
- Capacidad de generar y transmitir el conocimiento.
- Capacidad para motivar y propiciar el trabajo colaborativo

Objetivos

El propósito principal es preparar al estudiante para que desarrolle competencias en el manejo de las Herramientas de Programación que le permita solucionar problemas computacionales, ofreciéndole conocimientos básicos que favorecen su formación académica sirviendo como base para las unidades de aprendizaje relacionadas con el desarrollo de software y apoyando su formación como Técnico en Informática; y su posterior incorporación en estudios de Nivel Superior y/o al campo laboral.

Justificación

Las competencias profesionales (general y particulares) implican como principales objetos de conocimiento el diseño de algoritmos para solucionar problemas computacionales con el uso de las metodologías, procedimientos y herramientas de programación, mismos que podrá vincular con su entorno socioeconómico y laboral. Asimismo, en la particularidad el estudiante:

- Comprueba los criterios de solución y las heurísticas básicas con base en el planteamiento de problemas de diversa naturaleza.
- Diseña algoritmos para la solución de problemas con base en criterios estructurados.
- Diseña algoritmos a través de herramientas de programación.
- Programa aplicaciones sencillas a través de un entorno de programación

estructurada. Las habilidades y actitudes concretas que desarrolla el estudiante:

- Capacidad de análisis
- Dominio de las herramientas de programación
- Creativo e innovador
- Puntualidad y Responsabilidad
- Liderazgo
- Hábitos personales.
- Interés por el estudio
- Capacidad de generar y transmitir el conocimiento.
- Capacidad para motivar y propiciar el trabajo colaborativo

Asimismo, los principales objetos de conocimiento que se adquirirán y serán cuerpo de las acciones o desempeños a realizar son:

- Diseñar algoritmos computacionales para la solución de problemas con base a criterios estructurados
- Diseñar algoritmos con base en las herramientas de programación

Programará aplicaciones básicas a través de un entorno de programación estructurada

Estructura y contenidos

Unidad 1

Fases para la resolución de problemas: análisis, diseño, desarrollo, evaluación.

Métodos deductivo, inductivo, matemático y constructivo.

Métodos heurísticos:

descomposición, reducción, búsqueda local, brinca y atrapa, subir la cuesta, marcha atrás.

Definición de algoritmo. Estructura del algoritmo. Clasificación de los algoritmos.

Características de los algoritmos cualitativos y cuantitativos.

Técnicas de elaboración de algoritmos cualitativos y cuantitativos con enfoque en la programación orientada a objetos.

Tipos de datos

Conceptos: palabras reservadas, variable, constante, contador, acumulador, operadores aritméticos-lógicos y relacionales.

Estructuras de control: secuencial, selectiva, repetitiva.

Unidad 2

Simbología de los diagramas flujo. Reglas para crear diagramas de flujo. Estructuras de control.

Simbología de los diagramas flujo. Reglas para crear diagramas de flujo. Estructuras de control.

Unidad 3

Lenguajes de programación:

Componentes, Características, Traductores. Programa:

Fuente, Objeto, Ejecutable. Taxonomía.

Paradigmas:

Estructurado, Modular, Funcional, Declarativo y Orientado a objetos

Entorno del lenguaje de programación: Edición, compilación, ejecución, configuración.

Introducción al lenguaje de programación orientado a objetos.

Estructura de un programa:

Clases y objetos, tipos de datos. Expresiones:

Operadores aritméticos, lógicos y relacionales, jerarquía de operadores.

Constantes. Variables:

Contador, acumulador.

Entorno del lenguaje de programación: Edición, compilación, ejecución, configuración.

Introducción al lenguaje de programación orientado a objetos.

Estructura de un programa:

Clases y objetos, tipos de datos. Expresiones:

Operadores aritméticos, lógicos y relacionales, jerarquía de operadores.

Constantes. Variables:

Contador, acumulador.

Evaluación

La guía no tiene valor sobre el examen.

Materiales para la elaboración de la guía

Libreta u hojas de reciclado o blancas

Fuentes de información sugerida al final de la guía

Actividades de estudio

Búsqueda y análisis de la información
Elaboración de diagramas de flujo
Elaboración de Pseudocódigo
Elaboración de programas

Información adicional

Que desarrolles los conocimientos básicos sobre las características de la ciencia y sus métodos de estudio para las herramientas de programación.

Bibliografía básica

Vera, F. (2011). Diagramas de flujo. Trillas.

Joyanes, L. (2020). Fundamentos de programación. McGraw-Hill

Domínguez Vera, E. D., Flores Guerrero, M. D., & Rangel Aguilar, O. (2017). Algoritmos y diagramas de flujo con Raptor. Alfaomega.

Jiménez, J., Jiménez, E. M., & Alvarado, L. N. (2014). FUNDAMENTOS DE PROGRAMACIÓN. ALFAOMEGA GRUPO EDITOR.

Novara, P. (2003). PSeInt. (versión 20210906) [software]. GPL. <http://pseint.sourceforge.net/>

Carlisle, M., Wilson, T., Humphries, J., Moore, J. (2014). Raptor. (versión 2019) [software].
<https://raptor.martincarlisle.com/>

Ottogalli, K. A., & Martínez, A. A., & León, L. G. (2011). NASPOO: una notación algorítmica estándar para Programación Orientada a Objetos. Télématique, 10(1), 81-102. [fecha de Consulta 23 de agosto de 2021]. ISSN: 1856-4194. Disponible en: <https://www.redalyc.org/articulo.oa?id=78419688005>

Cairó, O. (2005). Metodología de la programación - Algoritmos, diagramas de flujo y programas 3ª Edición. México: Alfaomega.

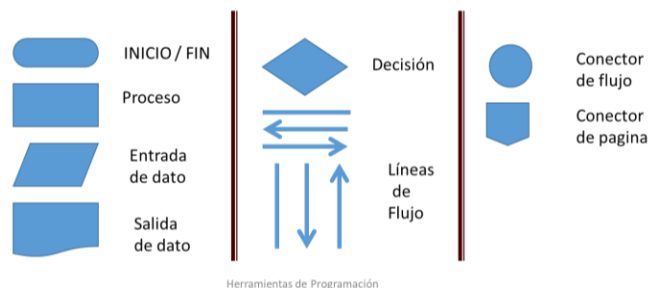
Integrantes de la academia

Carlos Francisco Arteaga Rodríguez

A. Conocimiento de terminología básica de cada una de las unidades del programa.

1. Algoritmo. Conjunto de pasos, procedimientos o acciones que nos permiten resolver un problema o alcanzar un resultado.
2. Características de un algoritmo:
 - Finito
 - Preciso
 - Efectivo
 - Cero o más entradas
 - 1 o más salidas
3. Diagramas de flujo
 - Representación gráfica de un algoritmo.
 - Muestra el flujo de los pasos del algoritmo usando un símbolo para representar cada operación o paso del algoritmo.
4. Simbología de un Diagrama de Flujo

✓ Los símbolos típicamente utilizados en los diagramas de flujo para programación son los siguientes:



5. Reglas para uso de un Diagrama de Flujo

- El sentido de un diagrama de flujo generalmente es de arriba hacia abajo y de izquierda a derecha.
- Los símbolos se unen con líneas, las cuales tienen en la punta una flecha que indica la dirección de flujo.
- Las líneas de flujo deben de entrar a un símbolo por la parte superior y salir de él por la parte inferior.

- Se deben de utilizar solamente líneas de flujo horizontales o verticales (nunca diagonales o de otro tipo).
- De un símbolo excepto el de decisión, solo puede salir una línea de flujo.
- En un símbolo solo puede entrar una flecha de flujo si varias líneas se dirigen al mismo símbolo, se deben unir en una sola flecha.

6. Seudocódigo con el software PseInt

- Es una imitación abreviada de instrucciones reales para las computadoras.
- Permite definir las estructuras de datos, las operaciones que se aplicarán a los datos y la lógica que tendrá el programa de computadora para solucionar un determinado problema.
- Similar al lenguaje natural
- Respeta las directrices y los elementos de los lenguajes de programación.

Tipo de Instrucciones	Instrucción a utilizar
Inicio y fin del programa	Inicio Termina
Instrucción de entrada	Leer
Instrucción de salida	Escribe
Operadores aritméticos	+, -, *, /, mod (cálculo de residuo), div (división entera), sqrt (raíz cuadrada), ** (cálculo de potencias)
Operadores lógicos	Y O No
Operadores relacionales	<, >, <=, >=, =
Asignación	←
Concatenación	+

- Estructura general de un algoritmo representado con

Pseudocódigo ALGORITMO NombreAlgoritmo

Declaración de variables

Sentencias del algoritmo

FINALGORITMO

7. Tipos de datos.

- Representación simbólica, atributo o característica de una entidad.
Por ejemplo
 - Rojo
 - 25 años
 - Café
 - Delgado
 - 9.8
- En programación es el elemento sobre el cual opera un algoritmo

Los tipos básicos de datos dependen de los lenguajes de programación y varían de uno a otro



8. Constantes. Es un dato cuyo valor se mantiene fijo mientras se realiza un algoritmo. Se le asocia el valor del símbolo que representa.

9. Variables.

- Son datos en los cuales el valor puede cambiar al realizar el algoritmo
- Una variable se representa con un símbolo alfanumérico
- En matemáticas se le conoce como literal
- Se acostumbra a representar con una letra, las más común es x
- En programación se le llaman identificadores
- Se le asocia un nombre descriptivo que indique para que se usa el dato variable
- Cuando se “crea” una variable en un programa, la computadora reserva espacio en la memoria de la computadora para almacenar el valor que pueda tomar la variable.

10. Expresiones aritméticas y lógicas

Una expresión lineal se evalúa de acuerdo con las siguientes reglas.

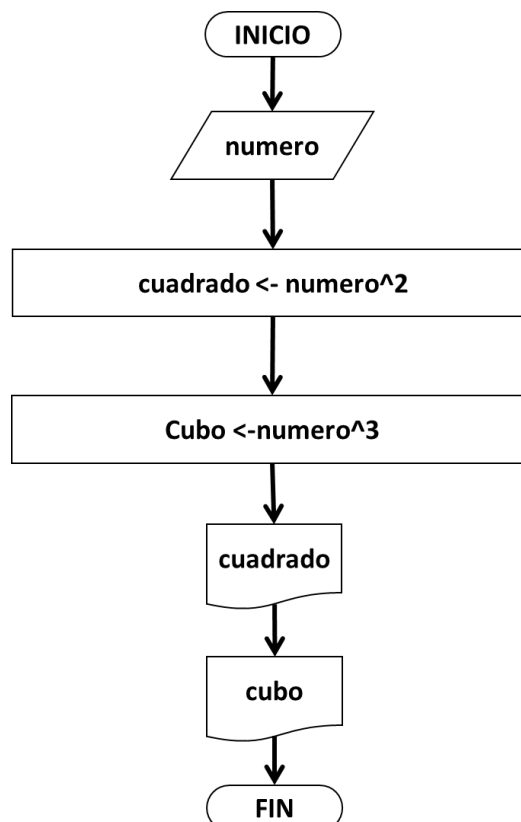
1. Realizar primero las operaciones de mayor jerarquía. Ver tabla de jerarquías
2. Si hay varios operadores de la misma jerarquía hacerlas de izquierda a derecha
3. Si existen paréntesis se procede de acuerdo con lo siguiente.
4. Si hay más de un nivel de paréntesis se comienza a resolver desde el nivel más profundo hacia arriba, aplicando en cada nivel las reglas 1 y 2.
5. Si los paréntesis están al mismo nivel se procede según lo indicado en la segunda regla y se resuelve las operaciones de acuerdo con la regla 1 y 2.
6. Si hay un solo nivel se aplican las reglas 1 y 2.

11. Jerarquía de operadores.

Operación	Símbolo	Jerarquía
Agrupación y especiales	()	Mayor
Potencia	$\wedge$	
Multiplicación, división, división entera, residuo	* / DIV MOD	
Suma, resta	+ -	
De comparación	=, <, >, <=, >=, <>	
Negación	No	
Conjunción	Y	
Disyunción	O	Menor

12. Prueba de escritorio. Las pruebas de escritorio son simulaciones del comportamiento de un algoritmo que permiten determinar la validez de este. Consisten en generar una tabla con tantas columnas como variables tenga el algoritmo y seguir las instrucciones poniendo los valores correspondientes

13. Representación de resolución de un problema en Diagrama de Flujo Diseñar un programa que calcule el cuadrado



y el cubo de un numero

14. Representación de resolución de un problema en PseInt

Diseñar un programa que calcule el cuadrado y el cubo de un numero

ALGORITMO Potencias

DEFINIR numero, cuadrado, cubo COMO REAL

LEER numero

cuadrado<- numero^2

cuadrado<- numero^3

ESCRIBIR cuadrado

ESCRIBIR cubo

FINALGORITMO

15. Representación de resolución de un problema en el lenguaje ANSI C Diseñar un programa que calcule el cuadrado y el cubo de un numero **Respuesta**

/*

Biblioteca de funciones

*/

#include<stdio.h>

#include<math.h>

/*

Cuando se llama el programa se realiza el código

incluido en la sección o función main

*/

int main() {

/*

Declaración de variables

**En ANSI C es necesario indicar el tipo de dato y el
identificador de la variable**

*/

float cuadrado;

float cubo;

float numero;

```
/*
```

Entrada de datos

**Se ponen mensajes informativos para que el usuario
tenga cierta idea de la que se espera que haga**

```
*/
```

```
printf("Introduzca un número\n",cubo);  
scanf("%f",&numero);
```

```
/*
```

Cálculos

En ANSI C no existe el operador ^

**por lo que se usa la función pow, esta función
esta en la biblioteca math.h**

```
*/
```

```
cuadrado = pow(numero,2);  
cuadrado = pow(numero,3);
```

```
/*
```

Salida de datos

**Se ponen mensajes informativos para que el usuario comprenda
lo que muestra el programa**

```
*/
```

```
printf("El cuadrado del número %f es %f\n",numero,cuadrado);  
printf("El cubo del número %f es %f\n",numero,cuadrado);  
return 0;
```

```
}
```

B. Para los siguientes problemas:

1. Realizar el análisis.
2. Diseñar su algoritmo en Diagrama de flujo, pseudocódigo (con la sintaxis de PseInt)
3. Probar su diseño haciendo la prueba de escritorio
4. Desarrollar su programa en lenguaje ANSI C.

1. Escriba un programa que calcule el valor del I.V.A. de una cantidad dada. El programa deberá desplegar en pantalla la cantidad original, la cantidad de I.V.A y la cantidad total.

Respuesta

Análisis

Datos de Salida

Cantidad
Cantidad de IVA
Cantidad total

Datos de Entrada.

Cantidad

Datos auxiliares

El IVA el 16% por de la cantidad

Condiciones.

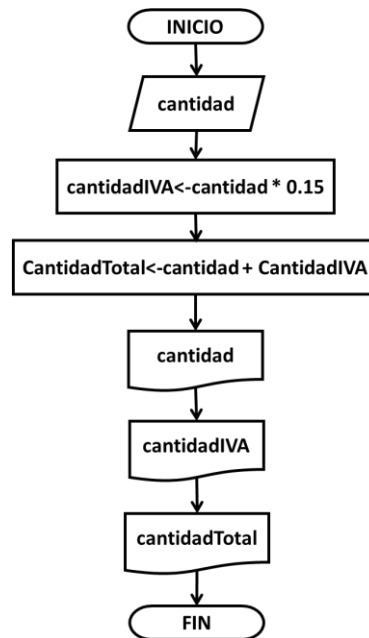
Cantidad mayor que cero

Algoritmo en lenguaje natural o cualitativo.

- | | | |
|--------|----|--|
| Diseño | 1. | Conocer la cantidad |
| | 2. | Realizar la siguiente operación para conocer la cantidad de IVA
$\text{Cantidad IVA} = \text{Cantidad} * 0.16$ |
| | 3. | Para conocer la cantidad total hacer lo siguiente
$\text{Cantidad total} = \text{cantidad} + \text{cantidad IVA}$ |
| | | |

Diagrama

Seudocódigo



Algoritmo IVA

leer cantidad

cantidadIVA <- cantidad * 0.16

CantidadTotal <- cantidad + cantidadIVA

Escribir cantidad

Escribir cantidadIVA

Escribir CantidadTotal

FinAlgoritmo

Prueba de escritorio

Entrada	Procesos		Salidas		
Cantidad	$\text{cantidadIVA} \leftarrow \text{cantidad} * 0.16$	$\text{CantidadTotal} \leftarrow \text{cantidad} + \text{cantidadIVA}$	Cantida d	CantidadIV A	CantidadTo tal
100	$\text{cantidadIVA} \leftarrow 100 * 0.16$ $\text{cantidadIVA} \leftarrow 16$	$\text{CantidadTotal} \leftarrow 100 + 16$	100	16	116

Programa

```
#include<stdio.h>

int main() {
    float cantidad;
    float cantidadiva;
    float cantidadtotal;
    printf("Introducir cantidad: ");
    scanf("%f",&cantidad);
    cantidadiva = cantidad*0.16;
    cantidadtotal = cantidad+cantidadiva;
    printf("Cantidad: %f\n",cantidad);
    printf("Monto IVA: %f\n",cantidadiva);
    printf("TOTAL (Cantidad + IVA): %f\n",cantidadtotal);
    return (0);
}
```

2. Haga un programa para calcular el perímetro, área y volumen de un cilindro. Respuesta

Análisis

Diseño

Poner valor de PI en 3.1416

Datos de Salida

Área y
volumen
de un
cilindro

- El área se calcula con la fórmula $A = 2\pi r(h+r)$
- El volumen se calcula con la fórmula $V = \pi r^2 h$

Condiciones.

Un cilindro no tiene perímetro.

Datos de Entrada.

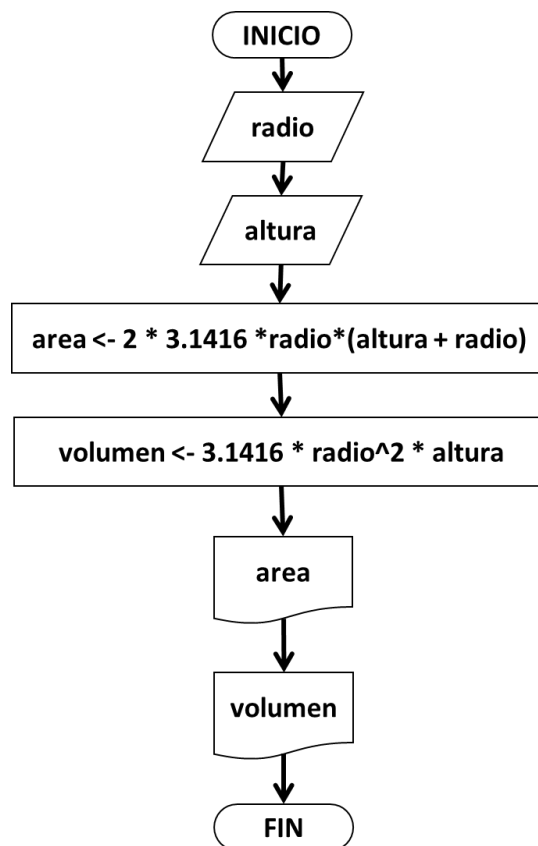
Radio y
altura

Algoritmo en lenguaje natural o cualitativo.

1. Obtener los valores del radio y la altura
2. Calcular el área
con $A = 2\pi r(h+r)$
3. Calcular el volumen
con $V = \pi r^2 h$
4. Mostrar el área y el volumen calculados

Datos auxiliares

Diagrama



Seudocódigo

Algoritmo cilindro

Definir area,volumen,radio,altura Como Real

leer radio

leer altura

area <- $2 * 3.1416 * \text{radio} * (\text{radio} + \text{altura})$

volumen <- $3.1416 * \text{radio}^2 * \text{altura}$

Escribir area

Escribir volumen

FinAlgoritmo

Prueba de escritorio

Entrada		Procesos		Salidas	
radio	altura	area <- $2 * 3.1416 * \text{radio} * (\text{radio} + \text{altura})$	volumen <- $3.1416 * \text{radio}^2 * \text{altura}$	area	volumen
1	1	area <- $2 * 3.1416 * 1 * (1 + 1)$ area <- $2 * 3.1416 * 1 * (2)$ area <- $2 * 3.1416 * 1 * (2)$ area <- 12.5664	volumen <- $3.1416 * 1^2 * 1$ volumen <- $3.1416 * 1 * 1$ volumen <- 3.1416	12.5664	3.1416

Programa

```
#include<stdio.h>
```

```
#include<math.h>
```

```
int main() {
```

```
    float altura;
```

```
    float area;
```

```
    float radio;
```

```
    float volumen;
```

```
    printf("Introducir radio de la base del cilindro");
```

```
    scanf("%f",&radio);
```

```
    printf("Introducir altura del cilindro");
```

```
    scanf("%f",&altura);
```

```
    area =  $2 * 3.1416 * \text{radio} * (\text{radio} + \text{altura})$ ;
```

```
    volumen =  $3.1416 * \text{pow}(\text{radio}, 2) * \text{altura}$ ;
```

```
    printf("El area del cilindro es %f\n",area);
```

```
    printf("La altura del cilindro es %f\n",volumen);
```

```
    return 0;
```

```
}
```

3. Proporcionar el precio promedio de un producto, calculado a partir del precio de este en tres establecimientos distintos. Los precios para el cálculo son introducidos por el usuario.

Respuesta

Análisis

Diseño

Datos de Salida

Precio promedio

Datos de Entrada.

3 precios

Datos auxiliares

Sin datos iniciales

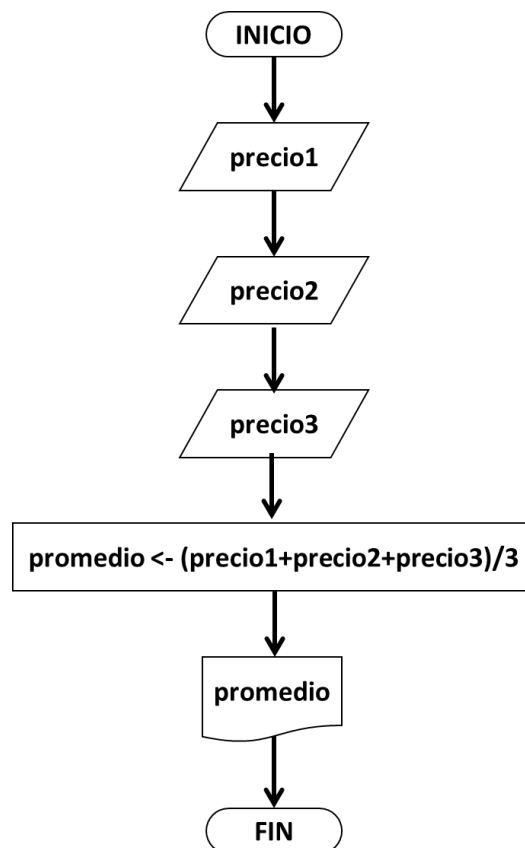
Condiciones.

Los precios son del mismo producto de diferentes establecimientos

Algoritmo en lenguaje natural o cualitativo.

1. Introducir precio 1, precio 2 y precio 3
2. Calcular el promedio de la siguiente manera. $\text{Promedio} = (\text{precio 1} + \text{precio 2} + \text{precio 3}) / 3$
3. Mostrar el promedio

Diagrama



Seudocódigo

ALGORITMO Promedio_precios

DEFINIR precio1,precio2,precio3,promedio COMO REAL

LEER precio1

LEER precio2

LEER precio3

$\text{promedio} \leftarrow (\text{precio1} + \text{precio2} + \text{precio3}) / 3$

ESCRIBIR promedio

FINALGORITMO

Prueba de escritorio

Entrada			Procesos	Salidas
Precio1	Precio2	Precio3	$\text{promedio} \leftarrow (\text{precio1} + \text{precio2} + \text{precio3}) / 3$	promedio
100	129	125	$\text{promedio} \leftarrow (100 + 129 + 125) / 3$	118

Programa

```
#include <stdio.h>
#include <stdlib.h>
// Ejercicios resueltos aprenderaprogramar.com
int main() {
    double precio1, precio2, precio3, promedio;
    // Datos
    printf ("Introduzca el precio en establecimiento 1, en pesos: ");
    scanf ("%f", &precio1);
    printf ("Introduzca el precio en establecimiento 2, en pesos: ");
    scanf ("%f", &precio2);
    printf ("Introduzca el precio en establecimiento 3, en pesos: ");
    scanf ("%f", &precio3);
    //Cálculo
    promedio = (precio1 + precio2 + precio3) / 3;
    printf ("El precio medio del producto es de %f pesos", media);
    return 0;
}
```

4. Escriba el pseudocódigo para calcular la distancia entre dos puntos dados P1 y P2. Respuesta

Análisis

Diseño

Datos de Salida

x_1, y_1, x_2, y_2

Datos de Entrada.

Distancia entre dos puntos

Datos auxiliares

Para calcular la distancia entre dos puntos en usar la formula $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

Condiciones.

x_1 y x_2 diferentes a x_2 y y_2

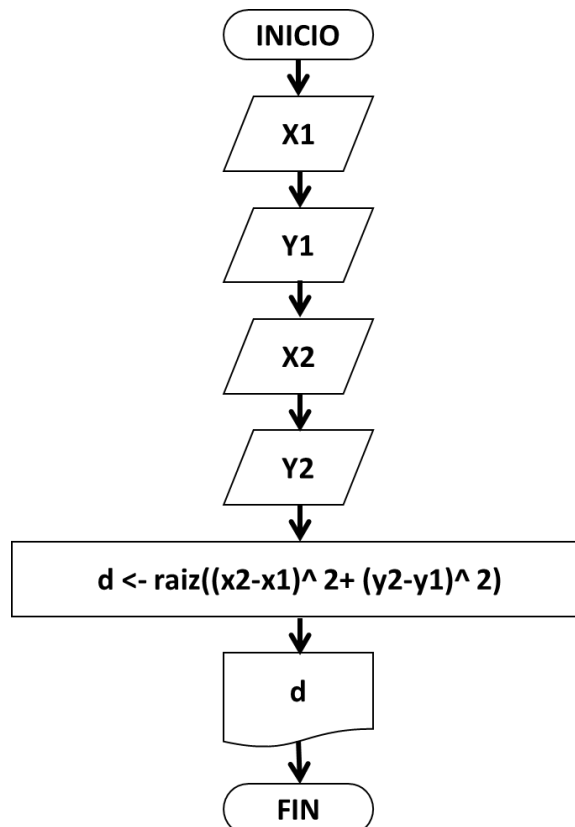
Algoritmo en lenguaje natural o cualitativo.

5. Obtener los valores de x_1 y y_1
6. Obtener los valores de x_2 y y_2
7. Sustituir en la siguiente expresión

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

8. Resolver la expresión del paso 3
9. Indicar cual es el valor de d

Diagrama



Seudocódigo

Algoritmo Distancia

 leer x1

 leer y1

 leer x2

 leer y2

$d \leftarrow \text{raiz}((x2-x1)^2 + (y2-y1)^2)$

 Escribir d

FinAlgoritmo

Prueba de escritorio

Entrada				Procesos	Salidas
X1	Y1	X2	Y2	$d \leftarrow \text{raiz}((x2-x1)^2 + (y2-y1)^2)$	d
1	1	2	2	$d \leftarrow \text{raiz}((2-1)^2 + (2-1)^2)$ $d \leftarrow \text{raiz}((1)^2 + (1)^2)$ $2) d \leftarrow \text{raiz}(1+1)$ $d \leftarrow \text{raiz}(2)$ $d \leftarrow 1.4142135624$	1.4142135624

Programa

```
#include<stdio.h>
#include<math.h>

int main() {
    float d;
    float x1;
    float x2;
    float y1;
    float y2;
    printf("Valor x del punto 1\n");
    scanf("%f",&x1);
    printf("Valor y del punto 1\n");
    scanf("%f",&y1);
    printf("Valor x del punto 2\n");
    scanf("%f",&x2);
    printf("Valor y del punto 2\n");
    scanf("%f",&y2);
    d = sqrt(pow((x2-x1),2)+pow((y2-y1),2));
    printf("La distancia entre P1(%f,%f) y P2(%f,%f) es %f\n",x1,y1,x2,y2,d);
    return (0);
}
```

5. Dado como dato el sueldo de un trabajador, considere un aumento del 15% si su sueldo es inferior a \$1000 y de un 12% en caso contrario. Imprima el sueldo con el aumento incorporado.

Respuesta

Análisis

Datos de Salida

Sueldo final

Datos de Entrada.

Sueldo

Diseño Datos auxiliares

Sin datos auxiliares

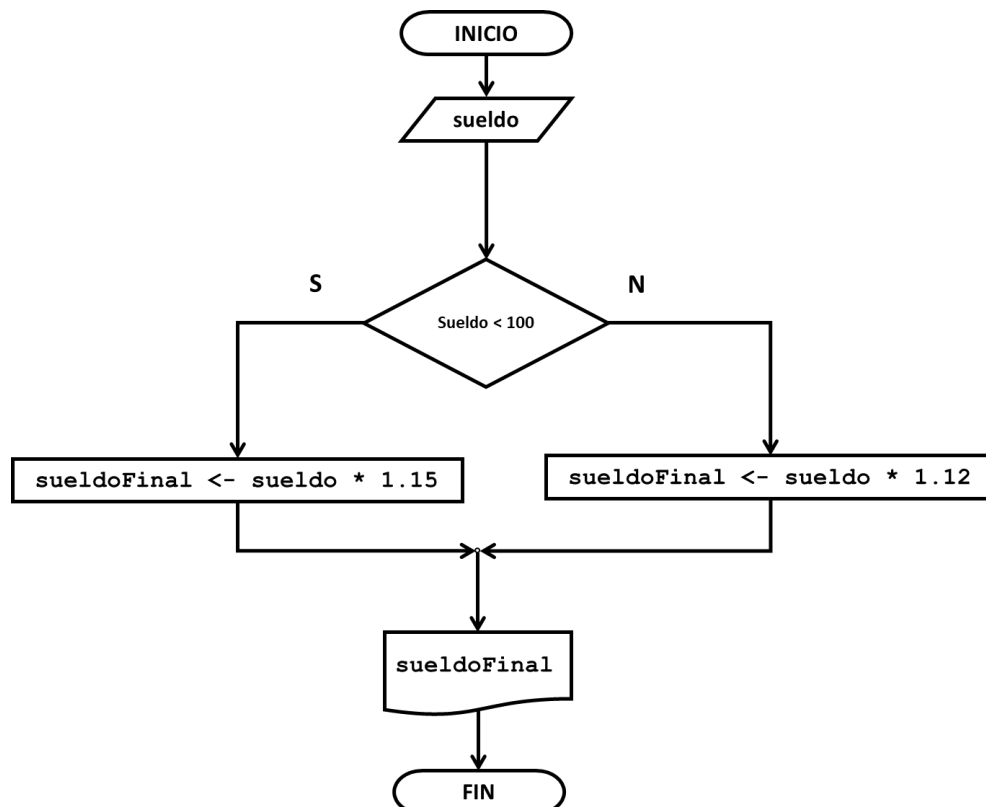
Condiciones.

Si el sueldo es menor que 1000 aumentar 15% en caso contrario aumentar 12%

Algoritmo en lenguaje natural o cualitativo.

1. Obtener el sueldo
2. Si el sueldo es menor a 1000 aumentarle el 15%, hacer la siguiente operación para conocer el sueldo final
 $\text{Sueldo final} \leftarrow \text{sueldo} * 1.15$
Si no aumentar el 12%, hacer la siguiente operación para conocer el sueldo final
 $\text{Sueldo final} \leftarrow \text{sueldo} * 1.12$
3. Mostrar el sueldo final

Diagrama



Seudocódigo

```
Algoritmo Aumento_de_sueldo
    DEFINIR sueldo, sueldofinal COMO REAL
    LEER sueldo
    SI sueldo < 1000 ENTONCES
        sueldofinal<-sueldo*1.15

    Sin
    o
        sueldofinal<-sueldo*1.12

    Fin
Si

    Escribir sueldofinal
FinAlgoritmo
```

Prueba de escritorio

Entrada	Procesos			Salidas
Sueldo	Sueldo <100	sueldofinal<-sueldo*1.15	sueldofinal<-sueldo*1.12	SueldoFinal
500	500 <100 Si	sueldofinal<-500*1.15 sueldofinal<-575		575
2000	2000 <100 Si	sueldofinal<-500*1.15 sueldofinal<-575	sueldofinal<-2000*1.12 sueldofinal<-2240	2240

Programa

```
#include<stdio.h>

int main() {
    float sueldo;
    float sueldofinal;
    scanf("%f",&sueldo);
    if (sueldo<1000) {
        sueldofinal = sueldo*1.15;

    }
    else {
        sueldofinal = sueldo*1.12;

    }
}
```

```
printf("%f\n",sueldofinal);  
return 0;
```

```
}
```

6. Escriba el pseudocódigo, que dados como datos la matrícula (o boleta) y 5 calificaciones de un alumno; imprima la matricula, el promedio y la palabra “aprobado” si el alumno tiene un promedio mayor o igual que 6, y la palabra “no aprobado” en caso contrario.

Respuesta

Análisis

Datos de Salida

Matricula

Promedio

Reprobado según se de la condición

Datos de Entrada.

Matricula

5 calificaciones

Datos auxiliares

Sin datos auxiliares

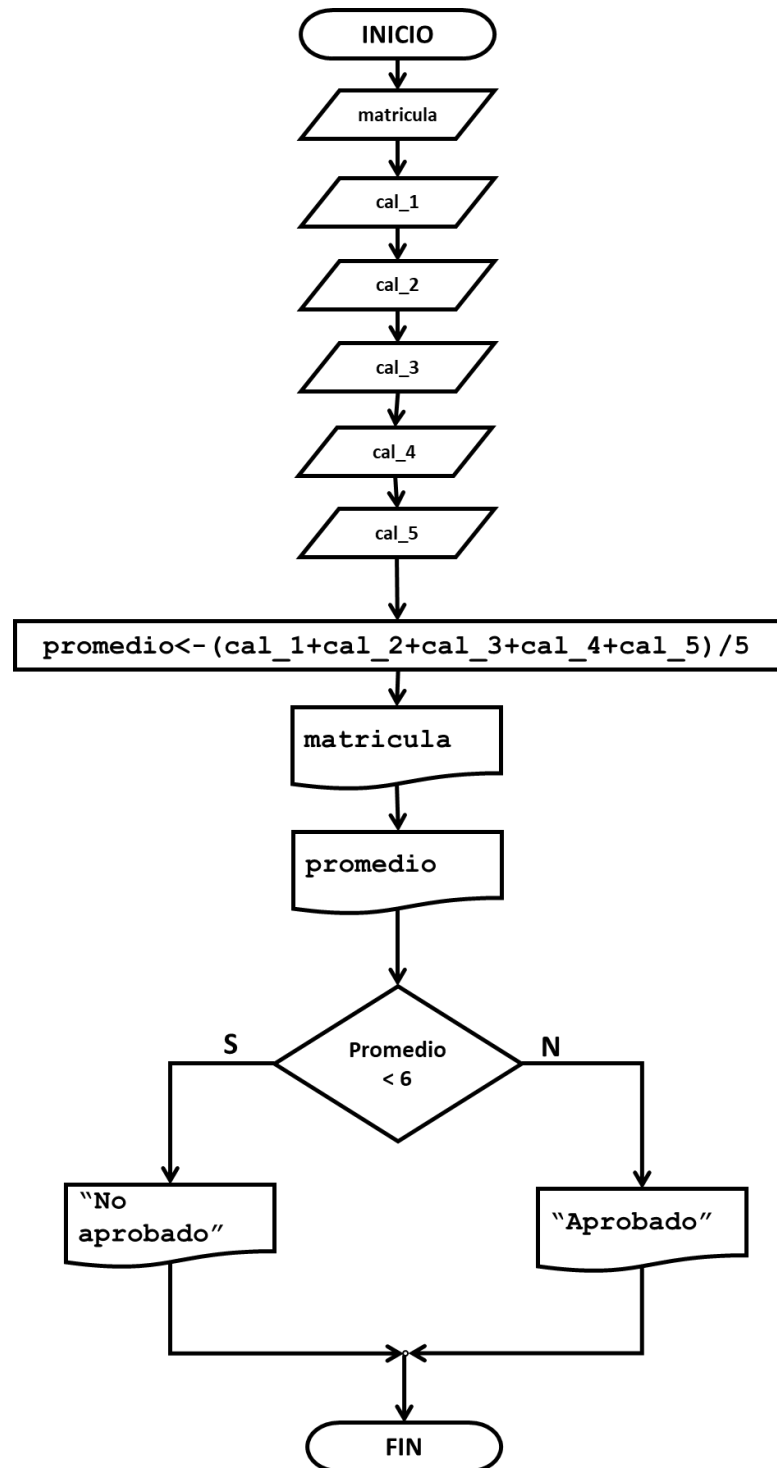
Condiciones.

Alumno reprobado si el promedio es menor a 6

Algoritmo en lenguaje natural o cualitativo.

1. Obtener la matricula
2. Obtener las 5 calificaciones
3. Calcular el promedio de la siguiente manera
$$\text{Promedio} = (\text{suma de las 5 calificaciones})/5$$
4. Mostrar la matricula
5. Mostrar el promedio
6. Si el promedio es menor a 6, Mostrar no aprobado, si es igual a 6 o mayor mostrar aprobado

Diseño
Diagrama



Seudocódigo

ALGORITMO Calificaciones

DEFINIR matricula,promedio,cal_1,cal_2, cal_3, cal_4,cal_5 COMO REAL
LEER matricula

LEER cal_1

LEER cal_2

LEER cal_3

LEER cal_4

LEER cal_5

$\text{promedio} \leftarrow (\text{cal}_1 + \text{cal}_2 + \text{cal}_3 + \text{cal}_4 + \text{cal}_5) / 5$

ESCRIBIR matricula

ESCRIBIR promedio

SI promedio < 6 ENTONCES

 ESCRIBIR "No aprobado"

SiNo

 ESCRIBIR "Aprobado"

FINSI

FINALGORITMO

Prueba de escritorio

Entrada						Procesos		Salidas			
Matricula	Cal_1	Cal_2	Cal_3	Cal_4	Cal_5	$\text{promedio} \leftarrow (\text{cal}_1 + \text{cal}_2 + \text{cal}_3 + \text{cal}_4 + \text{cal}_5) / 5$	$\text{promedio} < 6$	matricula	promedio	No aprobado	Aprobado
10001	5	6	6	5	5	$\text{promedio} \leftarrow (5 + 6 + 6 + 5 + 5) / 5$ $\text{promedio} \leftarrow (27) / 5$ $\text{promedio} \leftarrow 5.4$	$5.4 < 6$ S	10001	5.4	No aprobado	
10002	7	7	8	8	8	$\text{promedio} \leftarrow (7 + 7 + 8 + 8 + 8) / 5$ $\text{promedio} \leftarrow (38) / 5$ $\text{promedio} \leftarrow 7.6$	$7.6 < 6$ N	10002	7.6		Aprobado

Programa

```
#include<stdio.h> int main() {
    float cal_1; float cal_2; float cal_3;
    float cal_4; float cal_5; float
    matricula; float promedio;
    scanf("%f",&matricula); scanf("%f",&cal_1);
    scanf("%f",&cal_2);
    scanf("%f",&cal_3);

        scanf("%f",&cal_4);
        scanf("%f",&cal_5);
        promedio = (cal_1+cal_2+cal_3+cal_4+cal_5)/5;
        printf("%f\n",matricula);
        printf("%f\n",promedio);
        if (promedio<6) {
            printf("No aprobado\n");
        } else {

        }

    }

    printf("Aprobado\n");

    return 0;

}
```

7. Construya un diagrama de flujo que le permita calcular la tangente de un ángulo, considerado que se conoce el valor del seno y del coseno de este.

Respuesta

Análisis

Diseño

Datos auxiliares

Datos de Salida

Seno y coseno del ángulo

Tangente ángulo = seno ángulo / coseno ángulo

Tangent
e de un
ángulo

Condiciones.

Para el caso de ángulos de 90° la tangente no está definida

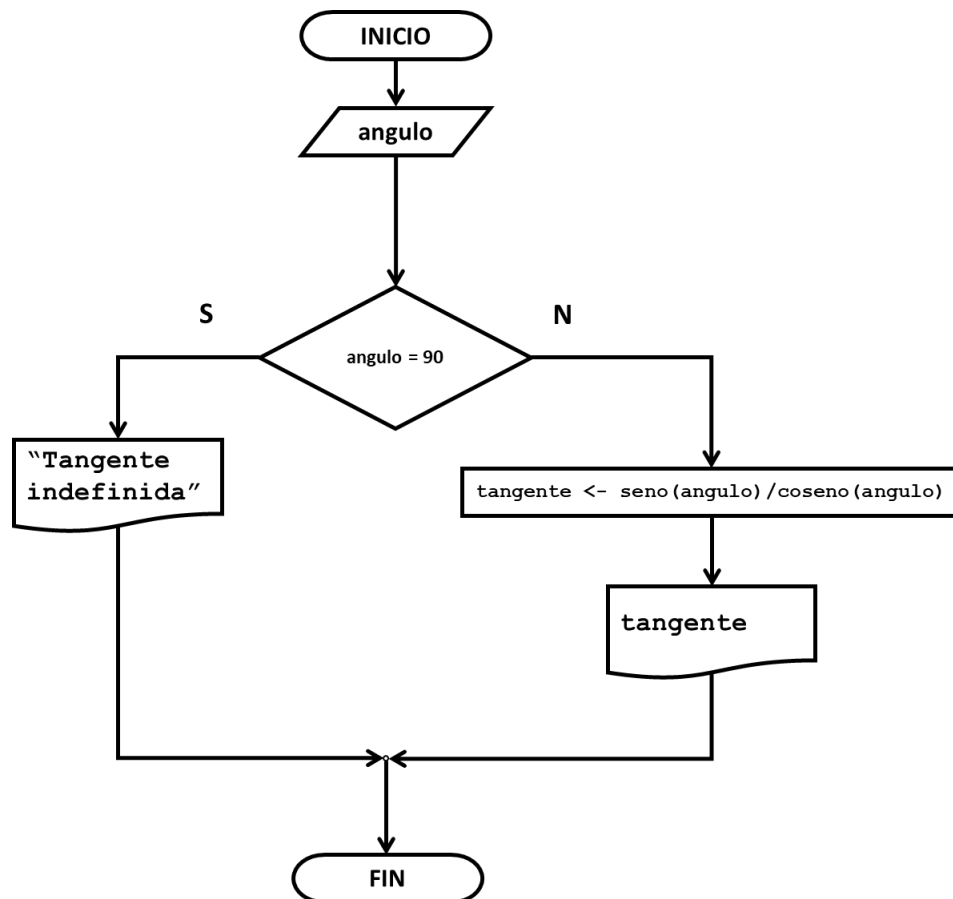
Datos de Entrada.

A
n
g
u
l
o

Algoritmo en lenguaje natural o cualitativo.

1. Capturar valor del Angulo en grados
2. Si el Angulo es de 90 grados enviar un error y terminar si no ir al paso 3
3. La tangente se calcula de la siguiente forma $\text{Tangente} = \text{seno} / \text{coseno}$
4. Mostrar valor de la tangente

Diagrama



Seudocódigo

```
Algoritmo Calculatangente
    definir tangente,angulo Como Real
    leer angulo
    si angulo = 90 Entonces
        Escribir "tangente indefinida"

    sino
        tangente <- sen(angulo)/cos(angulo)
        Escribir tangente
    Fin
Si
FinAlgoritmo
o
```

Prueba de escritorio

Entrada	Procesos		Salidas	
angulo	Angulo = 90	Tangente <- seno(angulo)/coseno(angulo)	Tangente	"tangente indefinida"
90	90=90 S			"tangente indefinida"
45	45 = 90 N	Tangente <- seno(45)/coseno(45) Tangente <- 0.7/0.7 Tangente <-1	1	

Programa

```
#include<stdio.h>
#include<math.h>
int main() {
    float angulo;
    float pi=3.1416;
    float radianes;
    float tangente;
    scanf("%f",&angulo);

    //En lenguaje las funciones sin y cos usan ángulos en radianes por lo que hay que convertir
    //Grados a radianes
    radianes = (angulo*pi)/180;
    if (angulo==90) {
        printf("tangente indefinida\n");
    }
    else {
```

```
}
```

```
tangente = sin(radianes)/cos(radianes);  
printf("%f\n",tangente);
```

```
return 0;
```

```
}
```

8. Construya un diagrama de flujo tal que, dados como datos de entrada tres números enteros, determine si los mismos están en orden creciente.

Respuesta

Análisis

Datos de Salida

Indicar si los números están en orden creciente

Datos de Entrada.

3 números

Datos auxiliares

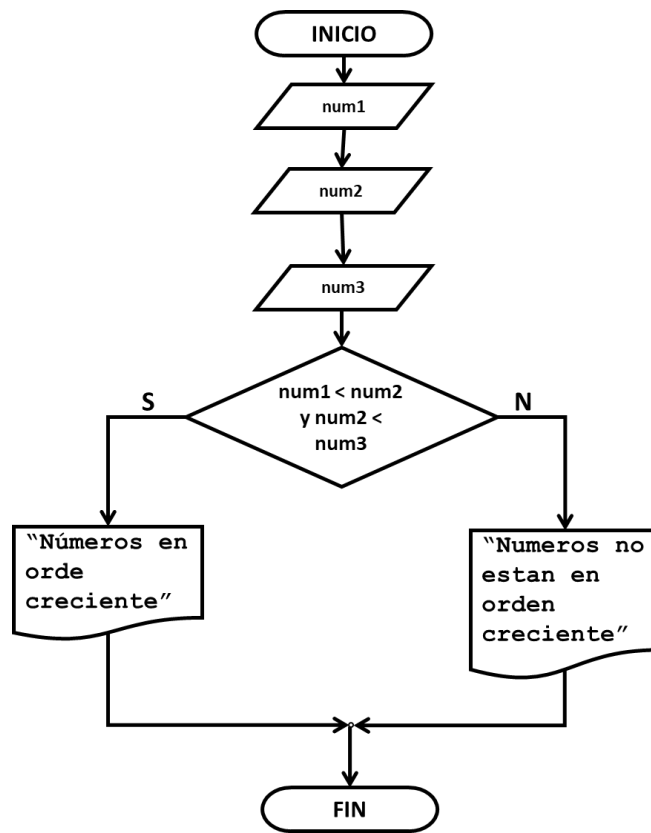
Sin datos auxiliares

Condiciones.

Los números están en orden creciente si $A < B < C$

Algoritmo en lenguaje natural o cualitativo.

- | | |
|-----------------|---|
| Diseño | <ol style="list-style-type: none">1. Introducir 3 números2. Si el primer número es menor al segundo y el segundo es menor al tercero están orden creciente, en caso contrario no lo están3. Mostrar mensaje |
| Diagrama | |



Seudocódigo

ALGORITMO Numeros_crece

DEFINIR num1,num2,num3 COMO REAL LEER num1

LEER num2

LEER num3

si num1<num2 y num2<num3 entonces

 Escribir "Numeros en orden creciente"

SiNo

FinSi Escribir "Números no están en orden creciente"

 ESCRIBIR promedio

FINALGORITMO

Prueba de escritorio

Entrada			Procesos	Salidas	
Num1	Num2	Num3	num1<num2 y num2<num3	"Números en orden creciente"	"Números no están en orden creciente"
1	2	3	1 < 2 y 2 < 3 S	Números en orden creciente	
3	1	2	3 < 1 y 1 < 2 N		Números no están en orden creciente

Programa

```
#include<stdio.h>
```

```
int main() {
```

```
    float num1;
```

```
    float num2;
```

```
    float num3;
```

```
    printf("Introducir un numero\n");
```

```
    scanf("%f",&num1);
```

```
    printf("Introducir un segundo numero\n");
```

```
    scanf("%f",&num2);
```

```
    printf("Introducir un tercer numero\n");
```

```
    scanf("%f",&num3);
```

```
    if (num1<num2 && num2<num3) {
```

```
        printf("Números en orden creciente\n");
```

```

        } else {

        }

printf("Números no están en orden creciente\n");
return 0;

}

```

9. En una tienda efectúan un descuento a los clientes dependiendo del monto de la compra. El descuento se efectúa en base al siguiente criterio:

Si el monto es menor que \$500, no hay descuento.

Si el monto está comprendido entre \$500 y hasta \$1,000, 5% de descuento.

Si el monto está comprendido entre \$1,000 y hasta \$7,000, 11% de descuento.

Si el monto está comprendido entre \$7,000 y hasta \$15,000, 18% de descuento.

Si el monto es mayor a \$15,000, 25% de descuento.

Desarrolle un programa que dado el monto de un cliente determine lo que debe de pagar el cliente.

Respuesta

Análisis

Datos de Salida

Pago del cliente

Datos de Entrada.

Monto

Datos auxiliares

Sin datos auxiliares

Condiciones.

Si el monto es menor que \$500, no hay descuento.

Si el monto está comprendido entre \$500 y hasta \$1,000, 5% de descuento.

Si el monto está comprendido entre \$1,000 y hasta \$7,000, 11% de descuento.

Si el monto está comprendido entre \$7,000 y hasta \$15,000, 18% de descuento.

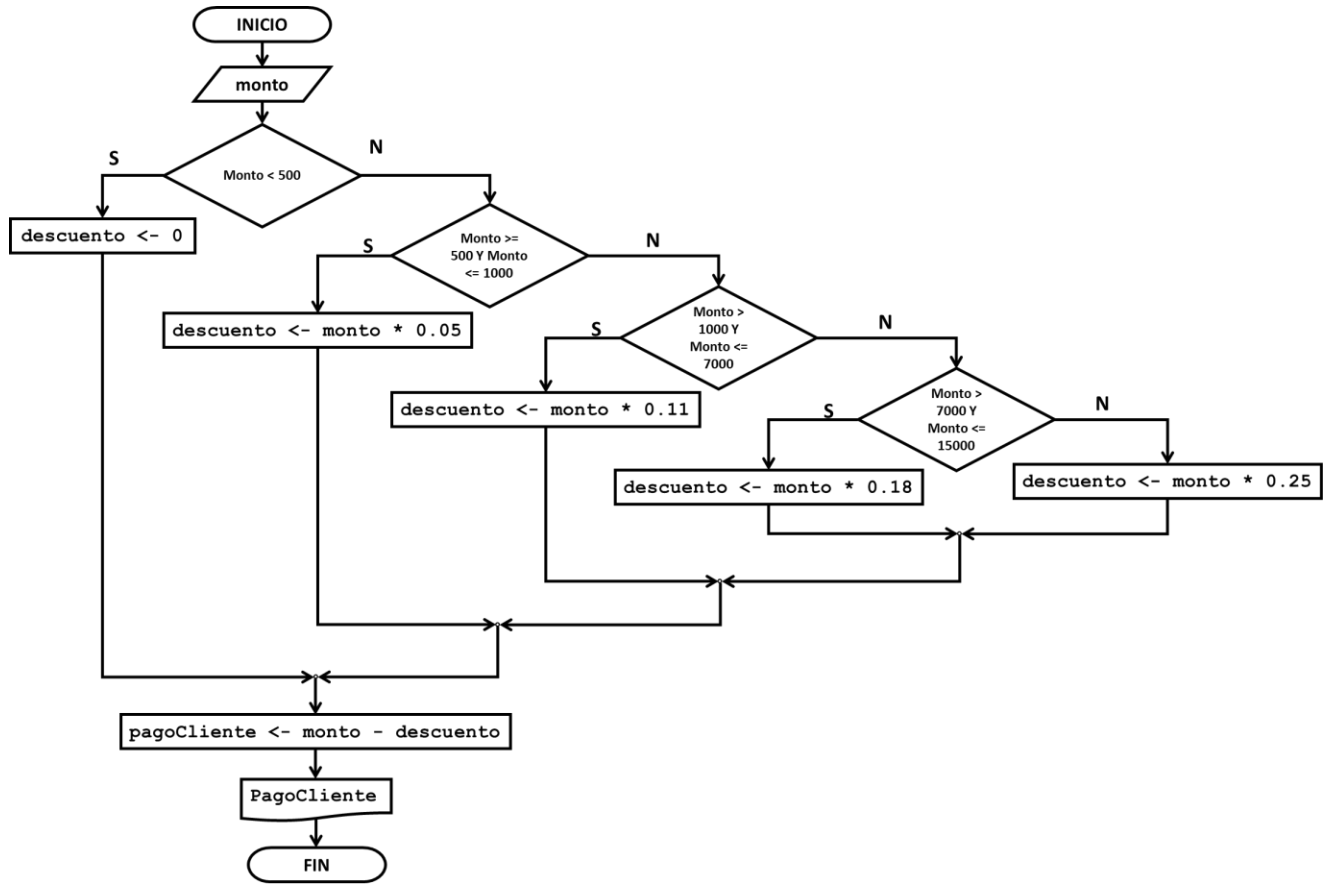
Si el monto es mayor a \$15,000, 25% de descuento.

Si el monto es mayor a \$15,000, 25% de descuento.

Algoritmo en lenguaje natural o cualitativo.

1. Obtener monto
2. Calcular el descuento según lo siguiente:
 - Si el monto es menor que 500, descuento $\leftarrow 0$
 - Si el monto está entre 500 y 1,000, calcular el descuento con descuento $\leftarrow \text{monto} * 0.05$.
 - Si el monto está entre 1,000 y 7,000, calcular el descuento con descuento $\leftarrow \text{monto} * 0.11$
 - Si el monto está entre \$7,000 y 15,000, calcular el descuento con descuento $\leftarrow \text{monto} * 0.18$
 - Si el monto es mayor 15,000, calcular el descuento con descuento $\leftarrow \text{monto} * 0.25$
3. Calcular el pago de cliente con Pago del cliente $\leftarrow \text{monto} - \text{descuento}$
4. Mostrar el pago del cliente

Diseño
Diagrama



Seudocódigo

```

Algoritmo Descuento_precio
    DEFINIR monto, descuento, pagoCliente COMO REAL
    LEER monto
    SI monto < 500 ENTONCES
        descuento<-0
    Sino
        SI monto > 500 y monto <= 1000 ENTONCES
            descuento<-monto * 0.05

        SiNo
            SI monto > 1000 y monto <= 7000 ENTONCES
                descuento<-monto * 0.11

            SiNo
                SI monto > 7000 y monto <= 15000 ENTONCES
                    descuento<-monto * 0.18

                SiN
                    o descuento<-monto * 0.25

            Fin
        fin
    FinSi
    Fin
    Si
    si

    pagoCliente<-monto-descuento
    Escribir pagoCliente
FinAlgoritmo
    
```

Prueba de escritorio

Entrada	Procesos										Salidas
Monto	monto < 500	des c uen t o<- 0	monto > 500 y monto <= 1000	descuento <- monto * 0.05	monto > 1000 y monto <= 7000	descu e nto<- mont o * 0.11	monto > 7000 y monto <= 15000	des c uen t o<- mo n to * 0.18	descu e nto<- mont o * 0.25	pagoClie n te<- monto- descuent o	PagoCliente

[illegible]

Programa

```
#include<stdio.h>

int main() {
    float descuento;
    float monto;
    float pagocliente;
    scanf("Introduzca monto de compra %f",&monto);
    if (monto<500) {
        descuento = 0;
    } else {
        if (monto>500 && monto<=1000) {
            descuento = monto*0.05;
        } else {
            if (monto>1000 && monto<=7000) {
                descuento = monto*0.11;
            } else {
                if (monto>7000 && monto<=15000) {
                    descuento = monto*0.18;
                }
                else {
                    descuento = monto*0.25;
                }
            }
        }
    }

    pagocliente = monto-descuento;
    printf("Total a pagar por el cliente %f\n",pagocliente);
    return 0;
}
```

10. Construya un diagrama de flujo tal que, dados como datos, la matrícula de un alumno, la carrera en la que está inscrito su semestre y su promedio; determine si el mismo es apto para pertenecer a alguna de las facultades menores que tiene la universidad. Si el alumno es aceptado teniendo en cuenta las especificaciones que se listan abajo, se debe imprimir su matrícula, carrera y la palabra aceptado:

Especificaciones para pertenecer a las facultades menores:

Economía: Desde sexto semestre y promedio mayor o igual a 8.8

Computación: Desde sexto semestre y promedio mayor a 8.5

Administración: Desde quinto semestre y promedio mayor a 8.5

Contabilidad: Desde quinto semestre y promedio mayor a 8.0

Respuesta

Análisis

Datos de Salida

Facultad

Datos de Entrada.

Matricula, carrera, semestre, promedio

Datos auxiliares

Sin datos auxiliares

Condiciones.

Las especificaciones para pertenecer a una facultad menor son:

Para Economía: Desde sexto semestre y promedio mayor o igual a 8.8

Para Computación: Desde sexto semestre y promedio mayor a 8.5

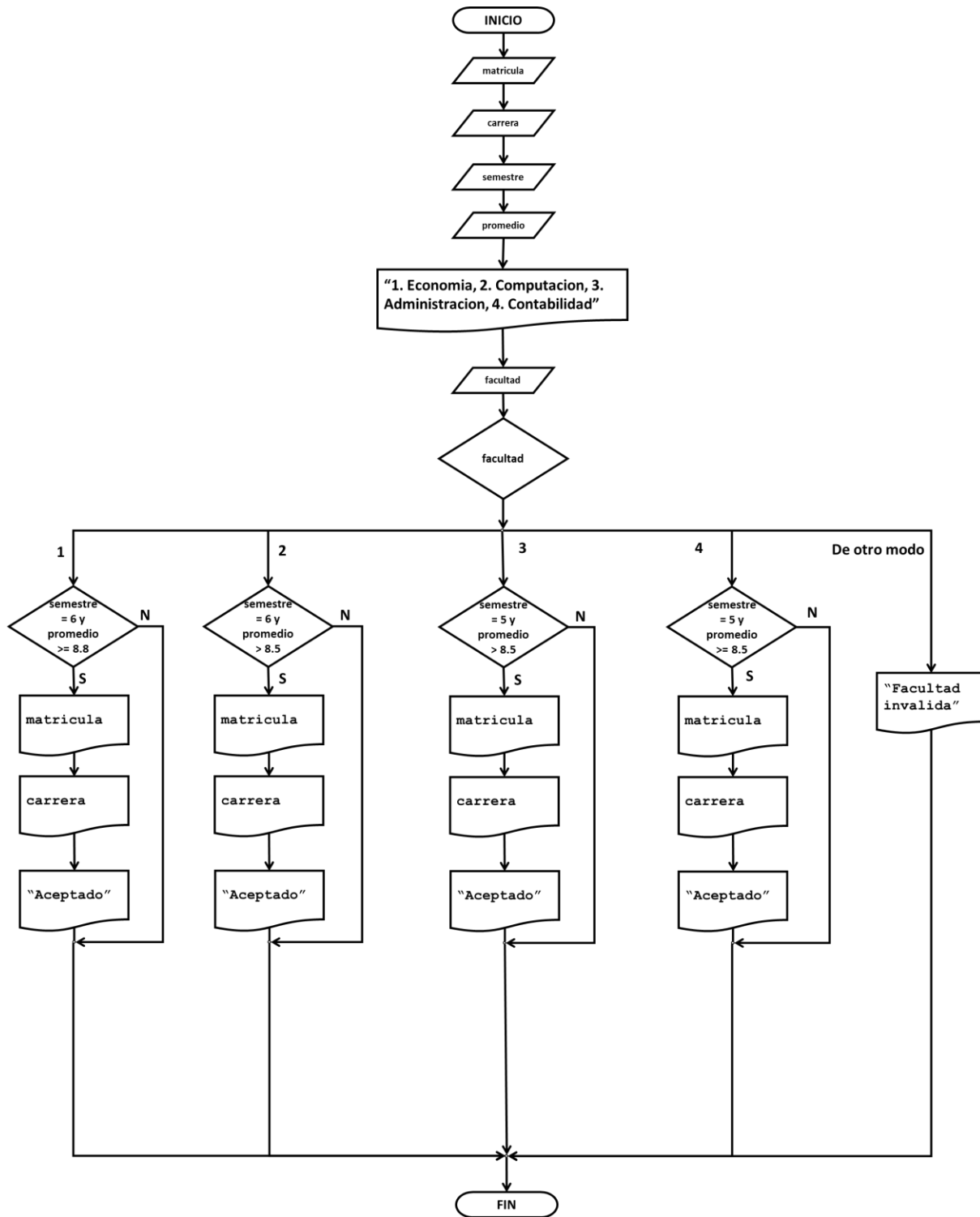
Para Administración: Desde quinto semestre y promedio mayor a 8.5

Para Contabilidad: Desde quinto semestre y promedio mayor a 8.5

Algoritmo en lenguaje natural o cualitativo.

1. Introducir matricula, carrera, semestre, promedio
2. Preguntar que facultad desea
3. Para Economía debe cumplir que promedio ≥ 8.8 y semestre es sexto
4. Para Computación debe cumplir que promedio > 8.5 y semestres es sexto
5. Para Administración debe cumplir que Si promedio > 8.5 y semestre es quinto
6. Para Contabilidad debe cumplir que Si promedio > 8.5 y semestre es quinto.

Diseño
Diagrama



Seudocódigo

Algoritmo Elegir_facultad

Definir matricula, promedio Como Real

Definir semestre Como Entero

Definir carrera Como Cadena

Definir facultad Como Entero

Leer matricula

Leer carrera

Leer semestre

Leer promedio

Escribir "1. Economía, 2. Computación, 3. Administración, 4. Contabilidad"

Leer facultad

Segun facultad Hacer

1:

si semestre = 6 y promedio $\geq$ 8.8 Entonces

Escribir matricula

Escribir carrera

Escribir "Aceptado"

FinSi

2:

si semestre = 6 y promedio $>$ 8.5 Entonces

Escribir matricula

Escribir carrera

Escribir "Aceptado"

FinSi

3:

si semestre = 5 y promedio $>$ 8.5 Entonces

Escribir matricula

Escribir carrera

Escribir "Aceptado"

FinSi

4:

si semestre = 5 y promedio > 8.5 Entonces

Escribir matricula

Escribir carrera

Escribir "Aceptado"

FinSi

De Otro Modo:

Escribir "Facultad invalida"

Fin Segun

FinAlgoritmo

Prueba de escritorio

Entrada					Procesos				Salidas		
matricula	carrera	semestre	Promedio	facultad	si semestre = 6 y promedio >= 8.8	si semestre = 6 y promedio > 8.5	si semestre = 5 y promedio > 8.5	si semestre = 6 y promedio >= 8.5	matri c ula	car r era	“ac e pta d o”
1000	dere c ho	sexto	8.5	2					1000	der ec h o	Ac e pta do

Programa

```
#include<stdio.h>
```

```
int main() {
```

```
    char carrera[256]; // forma para declarar datos tipo cadena en ANSI C
    int facultad;
```

```
    float matricula;
    float promedio;
    int semestre;
```

```
    printf("Introducir matricula\t");
    scanf("%f",&matricula);
    printf("Introducir carrera\t");
    scanf("%s",carrera);
    printf("Introducir semestre\t");
    scanf("%d",&semestre);
    printf("Introducir promedio\t");
    scanf("%f",&promedio);
```

```
    printf("Elige una facultad\n1. Economía\n2. Computación\n3. Administración\n4.
```

```
Contabilidad\n");
```

```
    scanf("%d",&facultad);
    switch (facultad) {
```

```
    case 1:
```

```
        if (semestre==6 && promedio>=8.8) {
            printf("Matricula: %f\n",matricula);
            printf("Carrera: %s\n",carrera);
            printf("Aceptado\n");
        }
```

```
    }
```

```
        break;
```

```
    case 2:
```

```
        if (semestre==6 && promedio>8.5) {
            printf("Matricula: %f\n",matricula);
            printf("Carrera: %s\n",carrera);
            printf("Aceptado\n");

        }

        break;

case 3: if (semestre==5 && promedio>8.5) {
        printf("Matricula: %f\n",matricula);
        printf("Carrera: %s\n",carrera);
        printf("Aceptado\n");
```

```
    }

    if (semestre==5 && promedio>8.5) {
r        printf("Matricula: %f\n",matricula);
e        printf("Carrera: %s\n",carrera);
a        printf("Aceptado\n");
k
    ;

case 4:

    }

    break;

default:

}        printf("Facultad invalida\n");
```

```
        return 0;  
    }  
}
```

11. Hacer un Diagrama de Flujo que genere e imprima los números del 100 al 0 en forma decreciente. Respuesta

Análisis

Datos de Salida

Lista de números del 100 al 0

Datos de Entrada.

Sin datos de entrada

Datos auxiliares

Sin datos auxiliares

Condiciones.

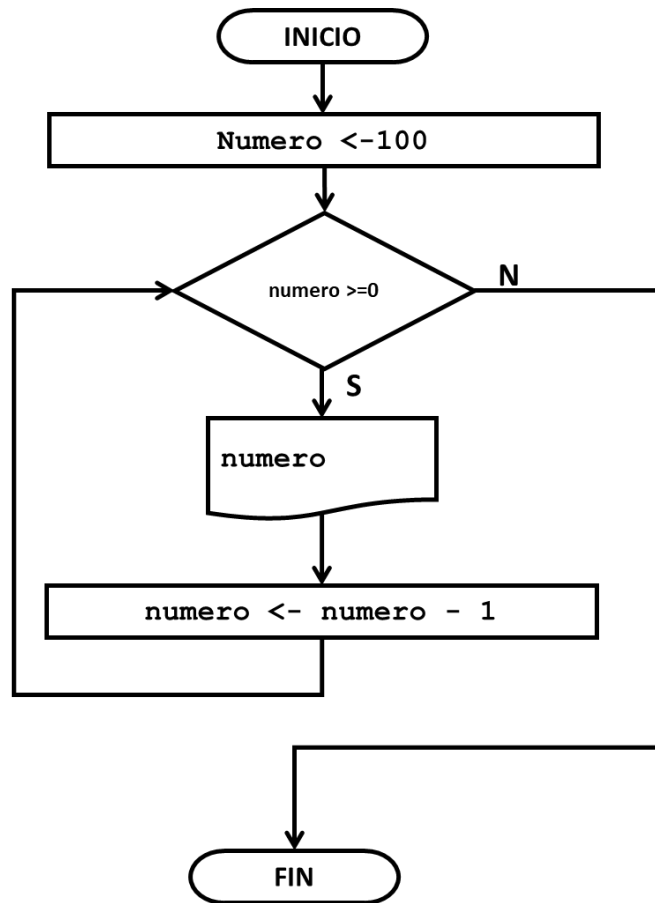
Terminar programa cuando se llegue a 0

Algoritmo en lenguaje natural o cualitativo.

Diseño

1. Número igual a 100
2. Mostrar el valor del numero
3. A numero restarle 1
4. Repetir desde el paso 2 hasta que numero sea igual a 0

Diagrama



Seudocódigo

Algoritmo lista

DEFINIR numero COMO Entero

numero <- 100

mientras numero >=0

 Escribir numero

 numero <- numero -1

finmientras

FinAlgoritmo

Prueba de escritorio

En el caso de programas con ciclos se puede omitir la prueba de escritorio.

Programa

```
#include<stdio.h>
```

```
int main() {
```

```
    int numero;
```

```
    numero = 100;
```

```
    while (numero>=0) {
```

```
        printf("%i\n",numero);
```

```
        numero = numero-1;
```

```
    }
```

```
    return 0;
```

```
}
```

12. Hacer un Diagrama de Flujo que lea un número N y que genere e imprima los números impares desde el número 1 hasta el número leído, así como cuantos son.

Respuesta

Análisis

Datos de Salida

Números impares entre 1 y n
Cantidad de impares

Datos de Entrada.

Numero

Datos auxiliares

Se sabe si un numero es impar cuando su residuo entre 2 es 1.

Condiciones.

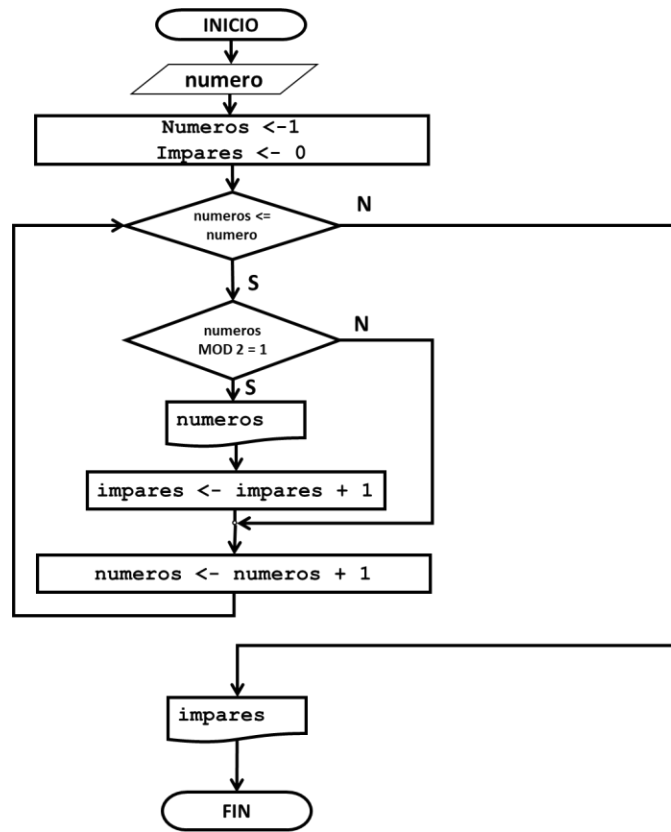
Usar números enteros

Algoritmo en lenguaje natural o cualitativo.

Diseño

1. Introducir el numero
2. Impares es cero
3. Números es 1
4. Si números modulo 2 es 1 mostrar el valor de números Impares crece en 1
5. Números crece en 1
6. Repetir desde el paso 4 hasta que números sea mayor a numero
7. Mostrar impares

Diagrama



Seudocódigo

Algoritmo Numeros_impares

definir numero, numeros, impares Como entero

leer numero

numeros<-1

impares<-0

Mientras numeros <= numero

 si numeros mod 2 = 1 Entonces

 Escribir numeros

 impares <- impares+1

 FinSi

 numeros <- numeros+1

FinMientras

Escribir impares

FinAlgoritmo

Prueba de escritorio

Entrada	Procesos						Salidas	
numero	numeros <-1	impares <-0	numeros <= numero	numeros mod 2 = 1	impares <- impares+1	numeros <- numeros+1	numeros	impares
5	numero s<-1	impar e s<-0	5 <=1 S	1 MOD 2 = 1 s	Impares <-1+1 Impares <-2	numeros <-1+1 numeros <- 2	1	
			5 <=2 S	2 MOD 2 = 1 N		numeros <-2+1 numeros <- 3		
			5 <=3 S	3 MOD 2 = 1 S	Impares <-1+1 Impares <-2	numeros <-3+1 numeros <- 4	3	
			5 <=4 S	4 MOD 2 = 1 n		numeros <-4+1 numeros <- 5		
			5 <=5 S	5 MOD 2 = 1 s	Impares <-2+1 Impares <-3	numeros <-5+1 numeros <- 6	5	
			6 <=5 N					3

Programa

```
#include<stdio.h>

int main() {

    int impares;
    int numero;
    int numeros;

    printf("Introduce un numero: ");

    scanf("%i",&numero);
    numeros = 1;

    impares = 0;

    while (numeros<=numero) {
        if (numeros%2==1) {

            printf("%i es impar\n",numeros);
            impares = impares+1;

        }

        numeros = numeros+1;

    }

    printf("Se encontraron %i impares\n",impares);
    return 0;

}
```

13. Hacer un Diagrama de Flujo que lea N matrículas, para cada matrícula se leerán 2 calificaciones, que calcule el promedio de las 2 calificaciones y que imprima la matrícula, las dos calificaciones y el promedio de las dos calificaciones para cada una de las N matrículas, además deberá imprimir cuantos alumnos aprobaron y cuántos alumnos reprobaron. El diagrama terminará cuando se lea una matrícula igual a 9999 que no se procesará.

Respuesta

Análisis

Datos de Salida

Para todas las matrículas capturadas
Matricula
2 calificaciones
Promedio
Al terminar de capturar las matriculas
Total aprobados
Total reprobados

Datos de Entrada.

Matricula
2 calificaciones

Datos auxiliares

Sin datos auxiliares

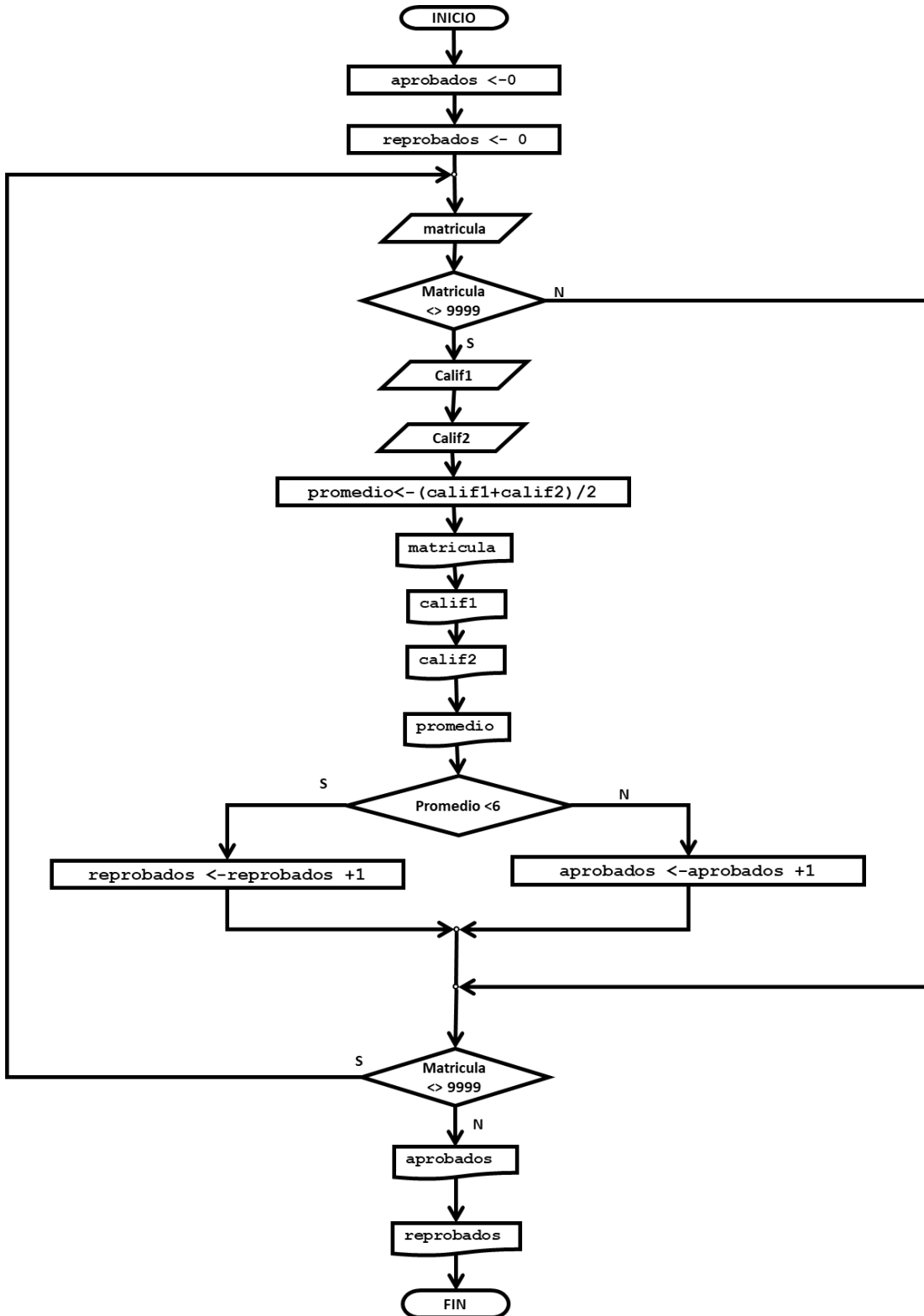
Condiciones.

El alumno reprueba si promedio es menor a 6 y aprueba si es mayor o igual a 6
Termina captura si matricula es 9999

Algoritmo en lenguaje natural o cualitativo.

1. Introducir matricula
2. Si matricula es diferente de 9999 ir al paso 3 sino terminar
3. Introducir 2 calificaciones
4. Calcular el promedio
5. Mostrar la matricula
6. Mostrar las calificaciones
7. Mostrar el promedio
8. Si el promedio es menor a 6 contar como reprobado sino contar como aprobado
9. Repetir desde el paso 1
10. Mostrar cuantos reprobados y cuantos aprobados
11. Terminar

Diseño
Diagrama



Seudocódigo

Algoritmo Calificaciones

Definir promedio, matricula Como Real

Definir calif1, calif2, aprobados, reprobados Como Real

aprobados <- 0

reprobados <- 0

Repetir

 Leer matricula

 Si matricula <> 9999 entonces

 Leer calif1

 Leer calif2

 promedio <- (calif1+calif2)/2

 Escribir matricula

 Escribir calif1

 Escribir calif2

 Escribir promedio

 si promedio < 6 Entonces

 reprobados <- reprobados + 1

 SiNo

 FinSi aprobados <- aprobados + 1

FinSi

Hasta Que matricula == 9999

Escribir aprobados

Escribir reprobados

FinAlgoritmo

Prueba de escritorio

[illegible]

Programa

```
#include<stdio.h>

int main() {
    float aprobados;
    float calif1;
    float calif2;
    float matricula;
    float promedio;
    float reprobados;
    aprobados = 0;
    reprobados = 0;
    do {
        printf("Introducir matricula (teclear 9999 para terminar)\n");
        scanf("%f",&matricula);
        if (matricula!=9999) {
            printf("Introducir calificación 1\n");
            scanf("%f",&calif1);
            printf("Introducir calificación 2\n");
            scanf("%f",&calif2);
            promedio = (calif1+calif2)/2;
            printf("Matricula : %f\n",matricula);
            printf("Calificación 1: %f\n",calif1);
            printf("Calificación 2: %f\n",calif2);
            printf("%f\n",promedio);
            if (promedio<6) {
                reprobados = reprobados+1;
            }
            else {
                aprobados = aprobados+1;
            }
        }
    }

    } while (matricula!=9999);
    printf("Total aprobados%f\n",aprobados);
    printf("Total reprobados%f\n",reprobados);
    return 0;
}
```

14. Hacer un Diagrama de Flujo que calcule la suma de los números impares y la suma de los números pares que hay entre el 300 y el 450.

Respuesta

Análisis

Datos de Salida

Suma de impares y de pares

Datos de Entrada.

No hay

Datos auxiliares

Sin datos auxiliares

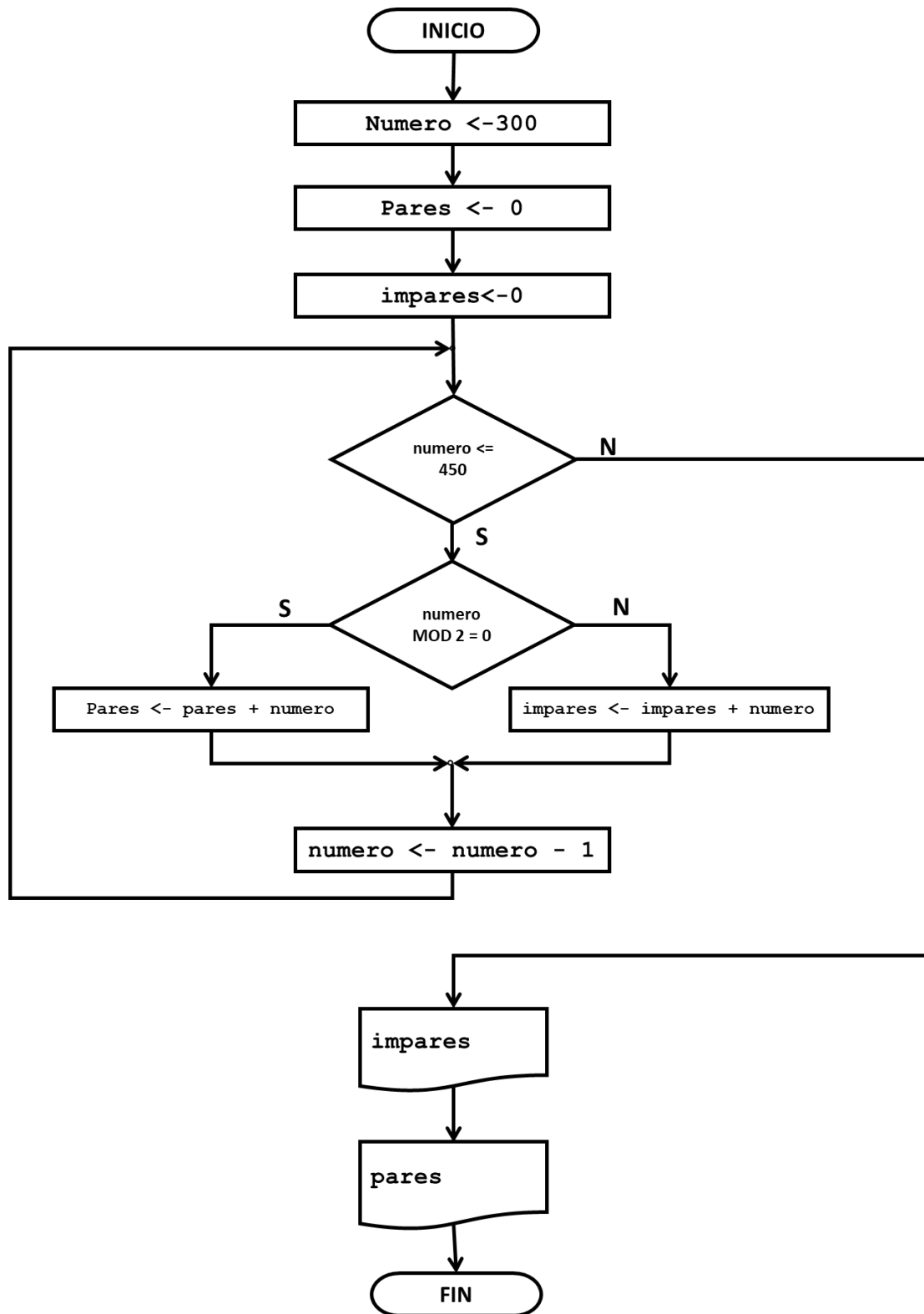
Condiciones.

Un numero es para si el residuo entre dos es cero e impar si es 1

Algoritmo en lenguaje natural o cualitativo.

1. Numero es 300
2. Par = 0 e impar = 0
3. Si el residuo de numero entre dos es cero a par sumarle el numero sino a impar sumarle el numero
4. A numero sumarle 1
5. Repetir desde el paso 2 mientras numero sea menor o igual a 450

Diseño
Diagrama



Seudocódigo

```
Algoritmo pares_impares
    Definir numero, pares, impares Como Entero
    numero <- 300
    pares <- 0
    impares <- 0
    Mientras numero <= 450 hacer
        si numero MOD 2 = 0 entonces
            pares <- pares + numero

        sin
        o
            impares <- impares + numero

        Fin
        Si

        numero <- numero + 1
    FinMientras
    Escribir pares
    Escribir impares
FinAlgoritmo
```

Prueba de escritorio

Entrada	Procesos								Salidas	
	numero <- 300	pares <- 0	impares< -0	numero <=450	numero MOD 2 = 0	pares <- pares + numero	impares <- impares + numero	numero< - numero +1	Pares	Impar e s
	numero <- 300	pares< - 0	impares< - 0	300 <=450 S	300 MOD 2 = 0 S	pares <- 0 + 300 pares <-300		numero<-300 +1 numero <-301		
				301 <=450 S	301 MOD 2 = 0 n	pares <- 0 + 300 pares <-300	pares <- 0 + 301 pares <-301	numero<-301 +1 numero <-301		
				Estos pasos se continúan repitiendo por 148 veces más hasta que numero sea 451 en ese momento termina el ciclo						
									2850 0	28125

Programa

```
#include<stdio.h>

int main() {
    int impares;
    int numero;
    int pares;
    numero = 300;
    pares = 0;
    impares = 0;
    printf("Programa que calcula la suma de pares e impares entre 300 y 450\n");
    while (numero<=450) {
        if (numero%2==0) {
            pares = pares+numero;

        }
        else {
            impares = impares+numero;

        }

        numero = numero+1;

    }

    printf("La suma de pares entre 300 y 450 es %d\n",pares);
    printf("La suma de impares entre 300 y 450 es %d\n",impares);
    return 0;
}
```

15. Hacer un Diagrama de Flujo para imprimir los primeros 30 términos de la secuencia 1, 1/2, 1, 1/4, 1, 1/8, 1, 1/16....

Respuesta

Análisis

Datos de Salida

Los 30 elementos de la serie 1, 1/2, 1, 1/4, 1, 1/8, 1, 1/16...

Datos de Entrada.

Sin datos de entrada

Datos auxiliares

Sin datos auxiliares

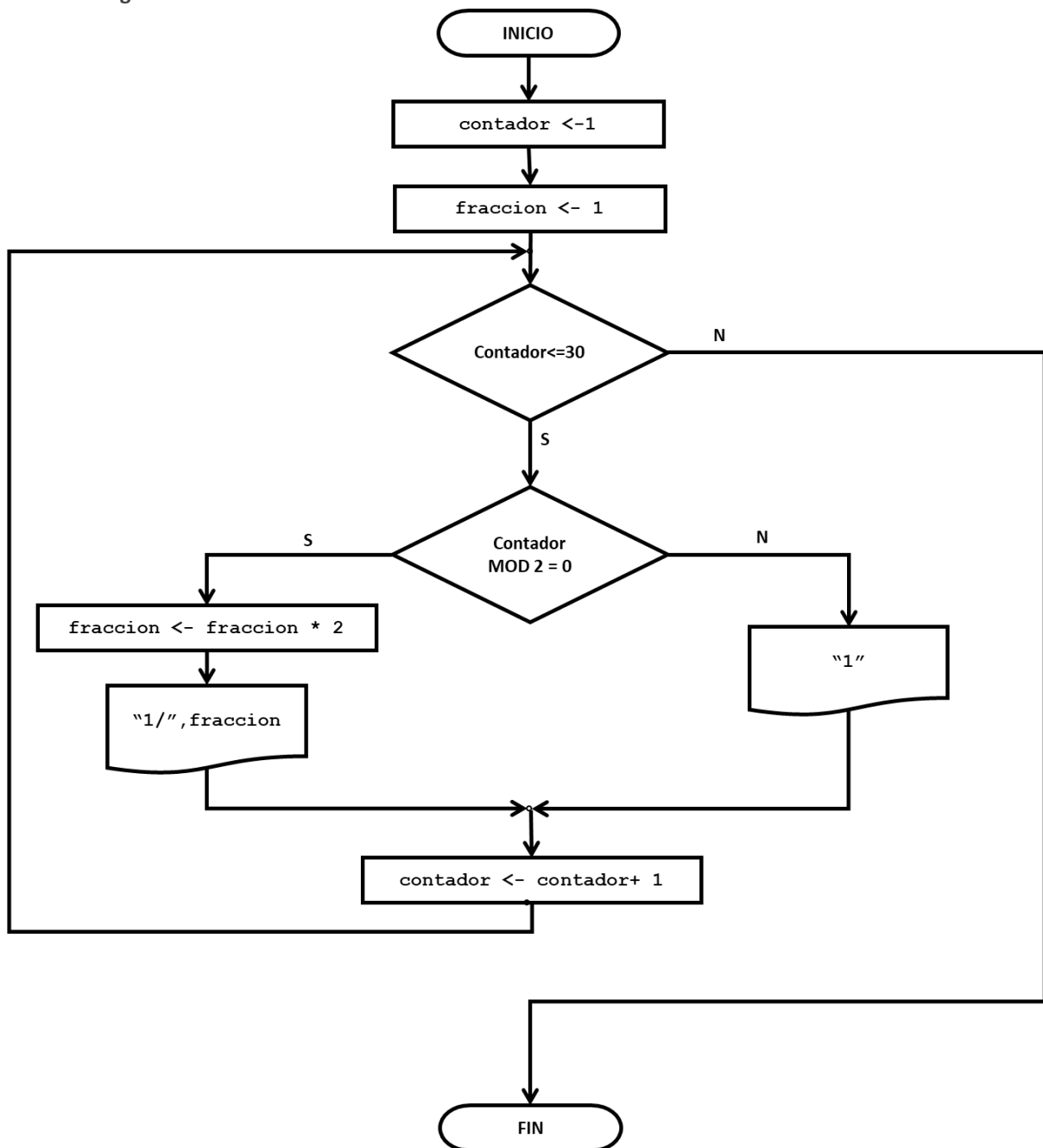
Condiciones.

Solo mostrar 30 elementos

Algoritmo en lenguaje natural o cualitativo.

1. Contador = 1
2. Fracción = 1
3. Si contador MOD 2 fracción = fracción*2 y mostrar 1/fracción
4. Sino mostrar 1
5. Al contador sumarle 1
6. Repetir desde el paso 3 mientras contador <=30

Diseño
Diagrama



Seudocódigo

Algoritmo serie

Definir contador,fraccion Como Entero

contador<-1

fraccion<-1

mientras contador <=30 Hacer

si contador MOD 2=0 Entonces

fraccion=fraccion*2

Escribir "1/",fraccion

sin

o

Escribir "1"

Fin

Si

contador<-contador+1

FinMientras

FinAlgoritmo

Prueba de escritorio

Entra da	Procesos						Salidas	
	contador<-1	fraccion<-1	contador <=30	contador MOD 2=0	fraccion=fracci o n*2	contador<- contador+1	1/fraccion	1
	contador<-1	fraccion<-1	1<=30 s	1 MOD 2 = 0 N		Contador <-1 +1 Contador<-2		1
			2<=30 s	2 MOD 2 = 0 S	Fracion <-1*2 Fraccion<-2	Contador <-2 +1 Contador<-3	1/2	
			3<=30 s	3 MOD 2 = 0 N		Contador <-3 +1 Contador<-4		1
			4<=30 s	4 MOD 2 = 0 S	Fracion <-2*2 Fraccion<-4	Contador <-4 +1 Contador<-5	1/4	
			5<=30 s	5 MOD 2 = 0 N		Contador <-5 +1 Contador<-6		1
			6<=30 s	6 MOD 2 = 0 S	Fracion <-4*2 Fraccion<-8	Contador <-6 +1 Contador<-7	1/8	
			7<=30 s	7 MOD 2 = 0 N		Contador <-7 +1 Contador<-8		1
			8<=30 s	8 MOD 2 = 0 S	Fracion <-8*2 Fraccion<-16	Contador <-8+1 Contador<-9	1/16	
			9<=30 s	9 MOD 2 = 0 N		Contador <-9+1 Contador<-10		1
			10<=30 s	10 MOD 2 = 0 S	Fracion <- 16*2	Contador <-10 + 1 Contador<- 11	1/32	

					Fraccion<-32			
			11<=30 s	11 MOD 2 = 0 N		Contador <-11 +1 Contador<-12		1
			12<=30 s	12 MOD 2 = 0 S	Fracion <- 32*2 Fraccion<-64	Contador <-12 + 1 Contador<- 13	1/64	
			13<=30 s	13 MOD 2 = 0 N		Contador <-13 +1 Contador<-14		1
			14<=30 s	14 MOD 2 = 0 S	Fracion <- 64*2 Fraccion<- 128	Contador <-14 + 1 Contador<- 15	1/128	
			15<=30	15 MOD 2 = 0		Contador <-15 + 1		1

			s	N		Contador<- 16		
			16<=30 s	16 MOD 2 = 0 S	Fracion <- 128*2 Fraccion<- 256	Contador <-16 + 1 Contador<- 17	1/256	
			17<=30 s	17 MOD 2 = 0 N		Contador <-17 + 1 Contador<- 18		1
			18<=30 s	18 MOD 2 = 0 S	Fracion <- 256*2 Fraccion<- 512	Contador <-18 + 1 Contador<- 19	1/512	
			19<=30 s	19 MOD 2 = 0 S		Contador <-19+1 Contador<-20		1
			20<=30 s	20 MOD 2 = 0 S	Fracion <- 512*2 Fraccion<- 1024	Contador <-20 +1 Contador<-21	1/1024	
			21<=30 s	21 MOD 2 = 0 N		Contador <-21 +1 Contador<-22		1
			22<=30 s	22 MOD 2 = 0 S	Fracion <- 1024*2 Fraccion<- 2048	Contador <-22 +1 Contador<-23	1/2048	
			23<=30 s	23 MOD 2 = 0 N		Contador <-23 +1 Contador<-24		1
			24<=30 s	24 MOD 2 = 0 S	Fracion <- 2048*2 Fraccion<- 4096	Contador <-24 +1 Contador<-25	1/4096	
			25<=30 S	25 MOD 2 = 0 N		Contador <-25 +1 Contador<-26		1
			26<=30 s	26 MOD 2 = 0 S	Fracion <- 4096*2 Fraccion<- 8192	Contador < 26 +1 Contador<-27	1/8192	
			27<=30 s	27 MOD 2 = 0 N		Contador <-27 +1 Contador<-28		1
			28<=30 s	28 MOD 2 = 0 S	Fracion <- 8192*2 Fraccion<- 16384	Contador <-28 + 1 Contador<- 29	1/16384	
			29<=30 s	29 MOD 2 = 0 N		Contador <-29 +1 Contador<-30		1
			30<=30 s	30 MOD 2 = 0 S	Fracion <- 16384*2 Fraccion<- 32768	Contador <- 30+1 Contador<-31	1/32768	
			31<=30 N					

Programa

```
#include<stdio.h>

int main() {

    int contador;
    int fraccion;
    contador = 1;

    fraccion = 1;

    printf("Muestra 30 elementos de la serie 1,1/2,1,1/4..\n");
    while (contador<=30) {

        if (contador%2==0) {

            fraccion = fraccion*2;
            printf("1/%d,",fraccion);

        } else {

            printf("1, ");

        }

        contador = contador+1;

    }

    return 0;

}
```

16. Durante las siguientes 3 semanas usted trabajará en la cenaduría de José, este pagará 1 dólar el primer día, 2 dólares el segundo día, 4 dólares el tercer día, y así sucesivamente, cada día pues, usted recibirá un salario doble que el recibido el día anterior. Hacer un Diagrama de Flujo para calcular cuál será su salario al término fijado.

Respuesta

Análisis

Datos de Salida

Salario total

1. Datos de
Entrada.

Sin datos de entrada

Datos auxiliares

3 semanas de trabajo

Salario primer día 1 dólar

Condiciones.

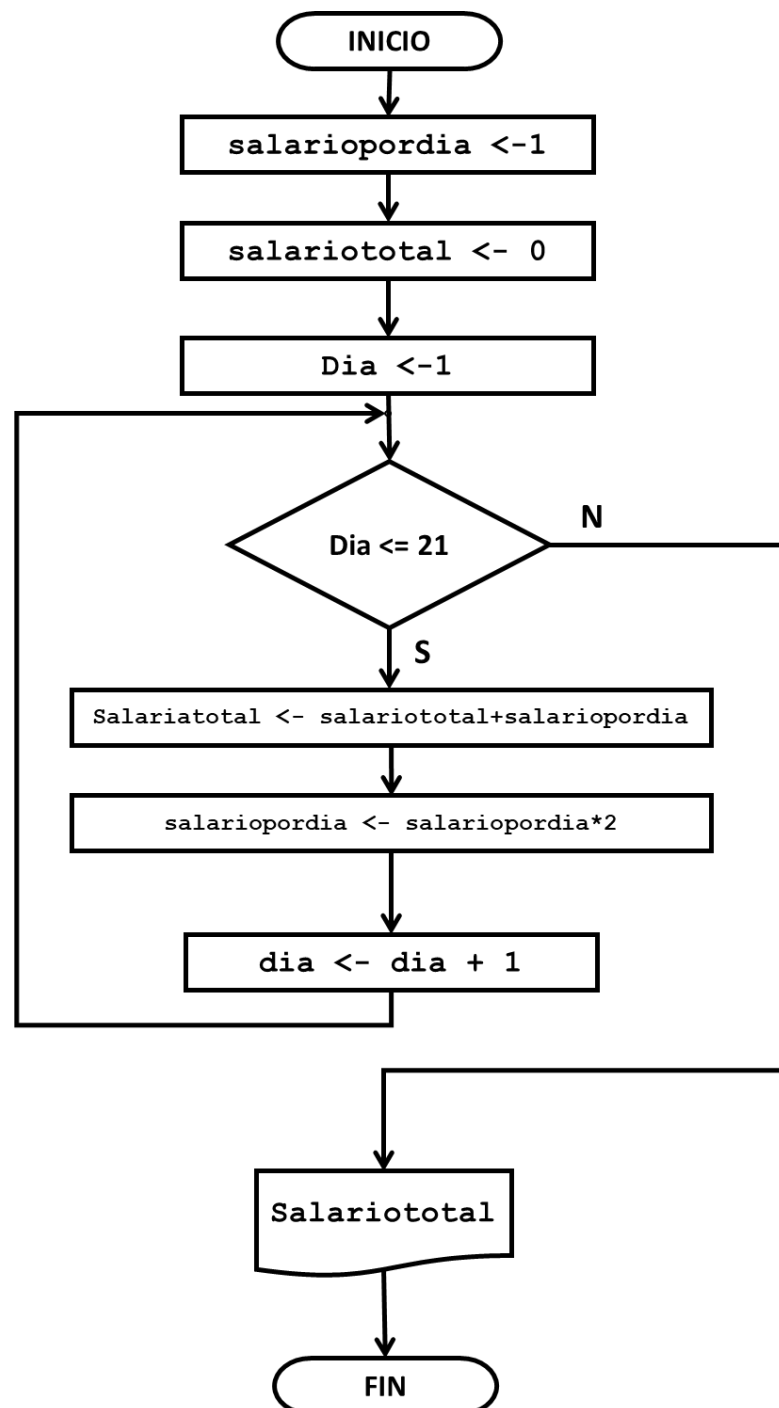
Establecer la semana de 7 días, en total son 21 días

El salario del siguiente día es el doble del día anterior

Algoritmo en lenguaje natural o cualitativo.

2. Día = 1
3. Salario por día = 1
4. Salario total = 0
5. Al salario total sumarle el salario por día
6. Al salario por día multiplicarlo por 2
7. Incrementar día en 1
8. Repetir desde el paso 2 hasta que día sea igual a 21
9. Mostrar el salario total

Diseño
Diagrama



Seudocódigo

Algoritmo salario

Definir dia, salariopordia, salariototal Como Entero

salariopordia<-1

salariototal<-0

dia<-1

Mientras dia<=21 hacer

 salariototal<-salariototal+salariopordia

 salariopordia<-salariopordia*2

 dia<-dia+1

FinMientras

Escribir salariototal

FinAlgoritmo

Prueba de escritorio

Entrada	Procesos							Salidas
	salariopordia<-1	salariototal<-0	dia<-1	dia<=21	salariototal<-salariototal+salariopordia	salariopordia<-salariopordia*2	dia<-dia+1	salariototal
	salariopordia<-1	salariototal<-0	dia<-1	1<=21 S	salariototal<-0 + 1 salariototal<-1	salariopordia<-1*2 salariopordia<-2	dia<-1+1 dia<-2	
			dia<-2	1<=21 S	salariototal<-1 + 2 salariototal<-3	salariopordia<-2*2 salariopordia<-4	dia<-2+1 dia<-3	
			dia<-3	1<=21 S	salariototal<-3 + 4 salariototal<-7	salariopordia<-4*2 salariopordia<-8	dia<-3+1 dia<-4	
			dia<-4	1<=21 S	salariototal<-7 + 8 salariototal<-15	salariopordia<-8*2 salariopordia<-16	dia<-4+1 dia<-5	
			dia<-5	1<=21 S	salariototal<-15 + 6 salariototal<-31	salariopordia<-16*2 salariopordia<-32	dia<-5+1 dia<-6	
			dia<-6	1<=21 S	salariototal<-31 + 32 salariototal<-63	salariopordia<-32*2 salariopordia<-64	dia<-6+1 dia<-7	

					tal<-63	rdia<-64		
			dia<-7	1<=21 S	salariot o tal<- 63 + 64 salariot o tal<- 127	salariop o rdia<- 64*2 salariop o rdia<- 128	dia<- 7+1 dia<-8	
			dia<-8	1<=21 S	salariot o tal<- 127 + 128 salariot o tal<- 255	salariop o rdia<- 128*2 salariopo rdia<- 256	dia<- 8+1 dia<-9	
			dia<-9	1<=21 S	salariot o tal<- 255 + 256 salariot o tal<- 511	salariop o rdia<- 256*2 salariop o rdia<- 512	dia<- 9+1 dia<-10	
			dia<-10	1<=21 S	salariot o tal<- 511 + 512 salariot o tal<- 1023	salariop o rdia<- 512*2 salariop o rdia<- 1024	dia<- 10+1 dia<-11	
			dia<-11	1<=21 S	salariot o tal<- 1023 + 1024 salariot o tal<- 2047	salariop o rdia<- 1024*2 salariop o rdia<- 2048	dia<- 11+1 dia<-12	
			dia<-12	1<=21 S	salariot o tal<- 2047 + 2048 salarioto tal<- 4095	salariop o rdia<- 2048*2 salariop o rdia<- 4096	dia<- 12+1 dia<-13	
			dia<-13	1<=21 S	salariot o tal<- 4095 + 4096 salarioto tal<- 8191	salariop o rdia<- 4096*2 salariop o rdia<- 8192	dia<- 13+1 dia<-14	
			dia<-14	1<=21 S	salariot o tal<- 8191 +8192 salariot o tal<- 16383	salariop o rdia<- 8192*2 salariop o rdia<- 16384	dia<- 14+1 dia<-15	

			dia<-15	1<=21 S	salariot o tal<- 16383 + 16384	salariop o rdia<- 16384*2	dia<- 15+1 dia<-16	
--	--	--	---------	------------	--	---------------------------------	--------------------------	--

					salariot o tal<- 32767	salariop o rdia<- 32786		
			dia<-16	1<=21 S	salariot o tal<- 32767 + 32768 salariot o tal<- 65535	salariop o rdia<- 32768*2 salariop o rdia<- 65536	dia<- 16+1 dia<-17	
			dia<-17	1<=21 S	salariot o tal<- 65535 + 65536 salarioto tal<- 13107 1	salariop o rdia<- 65536*2 salariop o rdia<- 131072	dia<- 17+1 dia<-18	
			dia<-18	1<=21 S	salariot o tal<- 131071 + 13107 2 salarioto tal<- 126214 3	salariop o rdia<- 131072* 2 salariop o rdia<- 126214 4	dia<- 18+1 dia<-19	
			dia<-19	1<=21 S	salariot o tal<- 126214 3 + 126214 4 salariot o tal<- 1524287	salariop o rdia<- 126214 4 *2 salariop o rdia<- 152428 8	dia<- 19+1 dia<-20	
			dia<-20	1<=21 S	salariot o tal<- 152428 7 + 152428 8 salariot o tal<- 1048575	salariop o rdia<- 152428 8 *2 salariop o rdia<- 104857 6	dia<- 20+1 dia<-21	

			dia<-21	1<=21 S	salariot o tal<- 104857 5 + 104857 6 salariot o tal<- 2097151	salariop o rdia<- 104857 6 *2 salariop o rdia<- 209715 2	dia<- 21+1 dia<-22	
			dia<-22	22<=21 N				2097151

Programa

```
#include<stdio.h>
```

```
int main() {
```

```
    int dia;
```

```
    int salariopordia;
```

```
    int salariototal;
```

```
    salariopordia = 1;
```

```
    salariototal = 0;
```

```
    dia = 1;
```

```
    printf("Calcula salario por trabajar 3 semanas en la cenaduria de Jose\n");
```

```
    while (dia<=21) {
```

```
        salariototal = salariototal+salariopordia;
```

```
        salariopordia = salariopordia*2;
```

```
        dia = dia+1;
```

```
    }
```

```
    printf("El salario total es: %d\n",salariototal);
```

```
    return 0;
```

```
}
```

17. Hacer un diagrama de flujo que calcule el área o el perímetro, según lo decida el usuario, de las siguientes figuras:

- A. Cuadrado**
- B. Rectángulo**
- C. Triangulo**
- D. Circulo**

Después de realizar el cálculo el programa preguntara al usuario si desea realizar otra operación, si la respuesta es positiva repita el procedimiento, si es negativa finalice el programa enviando un mensaje de despedida.

Respuesta

Análisis

Datos de Salida

Área o perímetro de figura seleccionada

Datos de Entrada.

Datos de la figura seleccionada

Datos auxiliares

Pi igual a 3.1416

Condiciones.

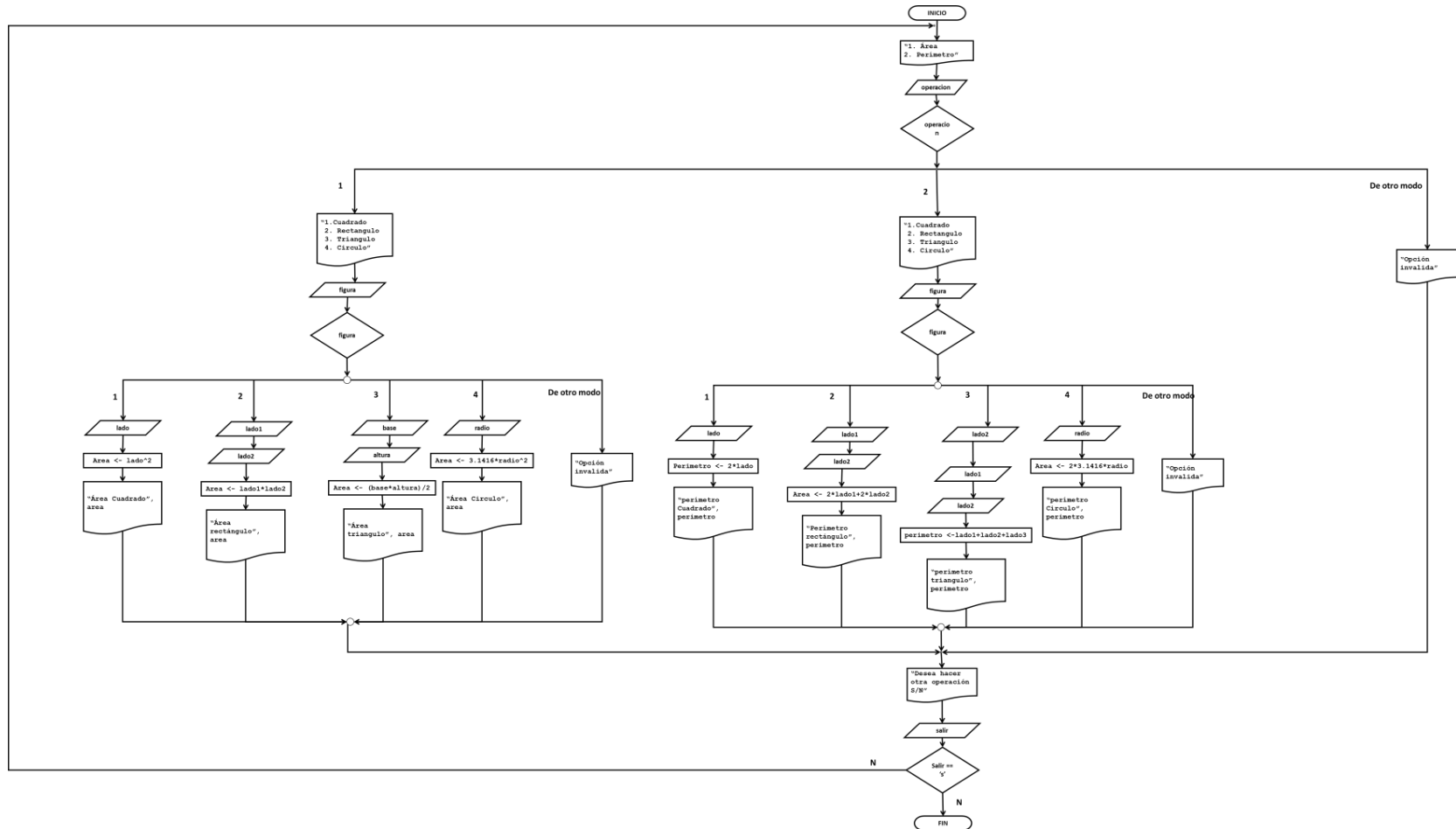
Dimensiones mayores a cero

Algoritmo en lenguaje natural o cualitativo.

1. Seleccionar área o perímetro
2. Si selecciona área
3. Seleccionar figura
4. Si selecciona cuadrado pedir longitud de un lado
5. El área se calcula con $\text{área} = \text{lado} * \text{lado}$
6. Si selecciona rectángulo pedir longitud de lado1 y de lado2
7. El área se calcula con $\text{área} = \text{lado1} * \text{lado2}$
8. Si selecciona triangulo pedir longitud de base y altura
9. El área se calcula con $\text{área} = (\text{base} * \text{altura}) / 2$
10. Si selecciona circulo pedir longitud del radio
11. El área se calcula con $\text{área} = \pi * \text{radio} * \text{radio}$
12. Si selecciona perímetro
13. Seleccionar figura
14. Si selecciona cuadrado pedir longitud de un lado
15. El perímetro se calcula con $\text{perímetro} = 4 * \text{lado}$
16. Si selecciona rectángulo pedir longitud de lado1 y de lado2
17. El área se calcula con $\text{área} = 2 * \text{lado1} + 2 * \text{lado2}$
18. Si selecciona triangulo pedir longitud de los lados
19. El perímetro se calcula con $\text{perímetro} = \text{lado1} + \text{lado2} + \text{lado3}$

20. Si selecciona circulo pedir longitud del radio
21. El perímetro se calcula con $\text{perímetro} = 2 * \pi * \text{radio}$
22. Para hacer otro calculo repetir desde el paso 1 si no desea repetir
23. Mostrar gracias por usar la aplicación

Diseño Diagrama



Seudocódigo

Algoritmo figuras

Definir area,perimetro,lado1,lado2,lado3,base,altura,radio Como Real

Definir operacion,figura Como Entero

Definir salir Como Caracter

Repetir

 Escribir "1. Área"

 Escribir "2. Perímetro"

 leer operacion

 Segun operacion Hacer

 1:

 Escribir "1. Cuadrado"

 Escribir "2. Rectángulo"

 Escribir "3. Triangulo"

 Escribir "4. Circulo"

 Leer figura

 Segun figura Hacer

 1:

 Leer lado1

$area \leftarrow lado1^2$

 Escribir "El area del cuadrado es ",area

 2:

 Leer lado1

 Leer lado2

$area \leftarrow lado1 * lado2$

 Escribir "El area del rectangulo es ",area

 3:

 Leer base

 Leer altura

$area \leftarrow (base * altura) / 2$

 Escribir "El area del triangulo es ",area

 4:

 Leer radio

$area \leftarrow 3.1416 * radio^2$

 Escribir "El area del circulo es ",area

 De Otro Modo:

 Escribir "Opción invalida"

 Fin Segun

2:

 Escribir "1. Cuadrado"

 Escribir "2. Rectángulo"

 Escribir "3. Triangulo"

 Escribir "4. Circulo"

 Leer figura

 Segun figura Hacer

 1:

Leer lado1

perimetro<-4*lado1

Escribir "El perimetro del cuadrado es ",perimetro

2:

Leer lado1

Leer lado2

$\text{perimetro} \leftarrow 2 * \text{lado1} + 2 * \text{lado2}$

Escribir "El perimetro del rectangulo es ",perimetro

3:

Leer lado1

Leer lado2

Leer lado3

$\text{perimetro} \leftarrow \text{lado1} + \text{lado2} + \text{lado3}$

Escribir "El perimetro del triangulo es ",perimetro

4:

Leer radio

$\text{perimetro} \leftarrow 2 * 3.1416 * \text{radio}$

Escribir "El perimetro del circulo es ",perimetro

De Otro Modo:

Escribir "Opción invalida"

Fin Segun

De Otro Modo:

Escribir "Opción invalida"

Fin Segun

Escribir "Desea realizar otra operacion S/N"

Leer salir

Hasta Que salir $\neq$'s'

Escribir "Gracias por usar la aplicación"

FinAlgoritmo

Prueba de escritorio

[illegible]

Programa

```
#include<stdio.h>
#include<math.h>
#include<ctype.h>
int main() {
    float altura;
    float area;
    float base;
    int figura;
    float lado1;
    float lado2;
    float lado3;
    int operacion;
    float perimetro;
    float radio;
    char salir;
    do {
        printf("1. Área\n");
        printf("2. Perímetro\n");
        scanf("%i",&operacion);
        switch (operacion) {
            case 1:
                printf("1. Cuadrado\n");
                printf("2. Rectángulo\n");
                printf("3. Triangulo\n");
                printf("4. Circulo\n");
                scanf("%i",&figura);
                switch (figura) {
                    case 1:
                        printf("Introducir longitud de lado: ");
                        scanf("%f",&lado1);
                        area = pow(lado1,2);
                        printf("El area del cuadrado es %f\n",area);
                        break;
                    case 2:
                        printf("Introducir longitud de lado: ");
                        scanf("%f",&lado1);
                        printf("Introducir longitud de lado: ");
                        scanf("%f",&lado2);
                        area = lado1*lado2;
                        printf("El area del rectángulo es %f\n",area);
                        break;
                    case 3:
                        printf("Introducir longitud de la base: ");
                        scanf("%f",&base);
```

```
printf("Introducir longitud de la altura: ");  
scanf("%f",&altura);  
area = (base*altura)/2;  
printf("El área del triángulo es %f\n",area);  
break;
```

```

case
4:      printf("Introducir longitud del radio: ");
        scanf("%f",&radio);
        arrea = 3.1416*pon(radio,2);
        printf("El área del circulo es %f\n",area);
        break;
default
t:

case 2:
    }

    break;

    printf("1. Cuadrado\n");
    printf("2. Rectángulo\n");
    printf("3. Triangulo\n");
    printf("4. Circulo\n");
    scanf("%i",&figura);
    switch (figura) {
case 1:

        printf("Introducir longitud de lado: ");
        scanf("%f",&lado1);
        perimetro = 4*lado1;
        printf("El perímetro del cuadrado es %f\n",perimetro);
        break;

case 2:
        printf("Introducir longitud de lado: ");
        scanf("%f",&lado1);
        printf("Introducir longitud de lado: ");
        scanf("%f",&lado2);
        perimetro = 2*lado1+2*lado2;
        printf("El perímetro del rectángulo es %f\n",perimetro);
        break;

case 3:
        printf("Introducir longitud de lado: ");
        scanf("%f",&lado1);
        printf("Introducir longitud de lado: ");
        scanf("%f",&lado2);
        printf("Introducir longitud de lado: ");
        scanf("%f",&lado3);
        perimetro = lado1+lado2+lado3;
        printf("El perímetro del triángulo es %f\n",perimetro);
        break;

```

```
case 4: printf("Introducir longitud del radio: ");
        scanf("%f",&radio);
        perimetro = 2*3.1416*radio;
        printf("El perímetro del círculo es %f\n",perimetro);
        break;
```

```
default: printf("Opción inválida\n");
t:
```

```
default: }
```

```
break;
```

```
printf("Opción inválida\n");
```

```
}
```

```
fflush(stdin);  
printf("Desea realizar otra operación S/N\n"); scanf("%c",&salir);  
} while (salir==tolower('s')); printf("Gracias por  
usar la aplicación\n"); return 0;
```