Pontificia Universidad Católica del Perú

Estudios Generales Ciencias

1INF01 - Fundamentos de Programación

Guía de laboratorio #3
Elaboración de programas con estructuras selectivas simples

6 de abril de 2019

Índice general

Historial de Revisiones

Revisión	Fecha	Autor(es)	Descripción
1.0	02.09.2018	A.Melgar	Versión inicial.
1.1	30.03.2019	A.Melgar	Se incrementó la cantidad de problemas propuestos, se añadió color al código en ANSI C y se completaron los casos de prueba de los problemas propuestos.

Siglas

ASCII	American Standard Code for Information Interchange
ANSI	American National Standards Institute
EEGGCC	Estudios Generales Ciencias
IDE	Entorno de Desarrollo Integrado
PUCP	Pontificia Universidad Católica del Perú
RAE	Real Academia Española

Capítulo 1

Guía de Laboratorio #3

1.1. Introducción

Esta guía ha sido diseñada para que sirva como una herramienta de aprendizaje y práctica para el curso de Fundamentos de Programación de los Estudios Generales Ciencias (EEGGCC) en la Pontificia Universidad Católica del Perú (PUCP). En particular se focaliza en el tema "Elaboración de programas con estructuras selectivas simples".

Se busca que el alumno resuelva paso a paso las indicaciones dadas en esta guía contribuyendo de esta manera a los objetivos de aprendizaje del curso, en particular en el diseño de programas con estructuras selectivas simples usando el paradigma imperativo. Al finalizar el desarrollo de esta guía y complementando lo que se realizará en el correspondiente laboratorio, se espera que el alumno:

- Comprenda el control de flujo de un programa en el paradigma imperativo y en particular la estructura selectiva simple.
- Diseñe algoritmos expresados en diagramas de flujo y pseudocódigos que controlen el flujo usando una estructura selectiva simple.
- Implemente programas que utilicen la estructura selectiva simple en un lenguaje de programación imperativo.
- Implemente programas que utilicen expresiones complejas que combinen operadores relacionales y lógicos.

1.2. Materiales y métodos

Como herramienta para el diseño de pseudocódigos y diagramas de flujo se utilizará PSeInt[1]. El PSeInt deberá estar configurado usando el perfil PUCP definido por los profesores del curso. Como lenguaje de programación imperativo se utilizará el lenguaje ANSI C. Como Entorno de Desarrollo Integrado (IDE) para el lenguaje ANSI C se utilizará Dev C++[2]. No obstante, es posible utilizar otros IDEs como Netbeans y Eclipse.

1.3. Estructura Selectiva Simple

Los programas escritos en el paradigma imperativo se basan en el cambio de estado de las variables definidas en los programas. Todo programa sigue un flujo el cual puede ser modificado a partir de estructuras de control

[1] http://pseint.sourceforge.net/
[2] http://sourceforge.net/projects/orwelldevcpp

de flujo. Las estructuras de control de flujo pueden ser de dos tipos: estructuras algoritmicas selectivas y estructuras algoritmicas iterativas.

Las estructuras selectivas, permiten que los programas ejecuten un conjunto de instrucciones si es que cumplen determinada condición. Es decir, el conjunto de instrucciones se ejecuta una sola vez si cumple la condición. Por otra parte, las estructuras iterativas permiten que los programas ejecuten un conjunto de instrucciones tantas veces como sea necesario, dependiendo de determinada condición. Las estructuras algorítmicas selectivas por su lado se pueden clasificar en estructuras selectivas simples y estructuras selectivas dobles. En esta guía se estudiará la estructura selectiva simple.

La estructura selectiva simple permite ejecutar un conjunto de instrucciones si y solo si se cumple determinada condición. Si la condición no se cumple, el conjunto de instrucciones no se ejecuta.

La estructura selectiva simple es muy usada en la programación imperativa, gracias a ella se pueden verificar situaciones antes de realizar determinado procesamiento. Muchas situaciones requieren que se verifiquen condiciones iniciales antes de realizar el procesamiento de datos, por ejemplo, para calcular la raíces de una ecuación cuadrática de una variable, si se desea obtener una solución real, el discriminante debe ser mayor o igual a cero. Si se desea calcular el área de un triángulo dado sus 3 lados, antes se debe verificar si efectivamente dichos lados forman un triángulo.

1.3.1. Representación de la Estructura Selectiva Simple

A continuación se revisará como se representa la estructura selectiva simple tanto en pseudocódigo como en diagrama de flujo, así como su implementación en lenguaje ANSI C.

1.3.2. Representación en pseudocódigo

En la figura 1.1 se puede apreciar la representación de la estructura selectiva simple en pseudocódigo. La estructura inicia con el identificador `Si` y finaliza con el identificador `Fin Si`. La `condición` es una expresión lógica como por ejemplo: $i < 100$, $nota \leq 10$, $i \leq max$ y $i <> 0$. Luego de la `condición` se coloca el identificador `Entonces`. En el `conjunto de instrucciones` se pueden colocar asignaciones ($i \leftarrow 0$), expresiones matemáticas ($suma \leftarrow suma + termino$) e inclusive otras estructuras selectivas (`Si i=0 Entonces`).

Si *condicion* **Entonces**
| conjunto de instrucciones;
Fin Si

Figura 1.1: Pseudocódigo: Estructura selectiva simple

1.3.3. Representación en diagrama de flujo

En la figura 1.2 se puede apreciar la representación de la estructura selectiva simple en diagrama de flujo. La estructura inicia con el bloque `condición` y si la condición es verdadera, se ejecuta el `conjunto de instrucciones`. En el diagrama, el control del flujo se gestiona a través de las líneas que conectan los bloques. Como se aprecia, inmediatamente después de ejecutar el bloque de instrucciones, el control se dirige hacia abajo. Si la `condición` no se satisface, el flujo se dirige hacia abajo sin procesar el `conjunto de instrucciones`.

1.3.4. Implementación en ANSI C

En el lenguaje ANSI C la estructura selectiva simple se implementa a través de la instrucción `if`. La representación del `if` se puede apreciar en el programa 1.1. El funcionamiento de la instrucción `if` es muy similar al `Si` del pseudocódigo y diagrama de flujo, pero hay que tener ciertas consideraciones para su uso.

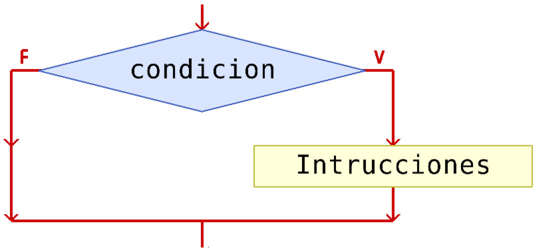

Figura 1.2: Diagrama de Flujo: Selectiva simple

Como se sabe, el ANSI C no implementa nativamente el tipo de dato lógico (`bool` o `boolean`). Entonces ¿Cómo hace el ANSI C para evaluar la condición de la instrucción `if`? ANSI C asume que todo valor igual a 0 falla una condición. El comportamiento es muy similar al *falso* de una expresión lógica. Por lo contrario, una valor diferente de 0 hará que se cumpla la condición. Comportamiento similar al *verdadero* de una expresión lógica. Por ejemplo: si se tienen las siguientes definiciones `int suma=0, i=10;`, las siguientes expresiones serán consideradas *verdaderas*: $suma \leq 100$, $i == 10$, $suma < i$, i. Por otro lado, las siguientes expresiones serán consideradas *falsas*: $suma \geq 100$, $i == 20$, $suma > i$, $suma$.

Programa 1.1: ANSI C: Estructura selectiva simple

```
...
if (condicion){
        conjunto de instrucciones;
}
...
```

1.4. Verificación del discriminante en el cálculo de las raíces ecuaciones cuadráticas con una variable

Una ecuación cuadrática con una variables es una ecuación que tiene la forma de $ax^2 + bx + c = 0$ siendo a, b y c números reales con la restricción que $a \neq 0$. Para encontrar la solución a la ecuación se puede utilizar la siguiente fórmula general $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$. La fórmula general genera una solución con dos raíces, la $x_1 = \frac{-b+\sqrt{b^2-4ac}}{2a}$ y la $x_2 = \frac{-b-\sqrt{b^2-4ac}}{2a}$ las cuales no son necesariamente diferentes.

Dada una ecuación cuadrática con una variable se solicita que elabore un algoritmo expresado en diagrama de flujo y pseudocódigo así como un programa en ANSI C que calcule las 2 raíces de la solución. En la guía #2 se resolvió este problema y se asumió que el discriminante siempre sería mayor o igual a 0 por lo que la solución daría un número real. En la realidad no se puede asumir esto pues el usuario puede ingresar los números que desee. ¿Qué se debe hacer entonces? Verificar antes de procesar si los datos ingresados permiten dicho procesamiento. En este caso en particular primero se debe hacer la lectura de los datos de la ecuación, posteriormente calcular el discriminante y finalmente verificar si este es mayor o igual a 0. Esta verificación se realiza con una estructura selectiva simple. Solo si se cumple la condición (≥ 0) se procede al cálculo de las raíces. Si no se cumple la condición, no se hace nada.

En la figura 1.3 se puede apreciar el diagrama de flujo que diseña la alternativa de solución propuesta. Note la inclusión de la estructura selectiva simple (bloque con la condición $discriminante \geq 0$).

El problema con el diagrama de flujo que se visualiza en la figura 1.3 es que si es que si no hay solución real ($discriminante < 0$), el algoritmo no imprime nada. Para solucionar este problema se puede colocar otra estructura algorítmica selectiva simple[3] que permita evaluar este caso en particular, de forma tal que cuando no se pueda retornar una solución real, se presente el mensaje informativo adecuado. La alternativa de solución con la estructura algorítmica selectiva simple adicional se puede apreciar en la figura 1.4.

[3]La solución más natural para este problema sería utilizar una estructura algorítmica selectiva doble. Este tema se discutirá en la próxima guía.

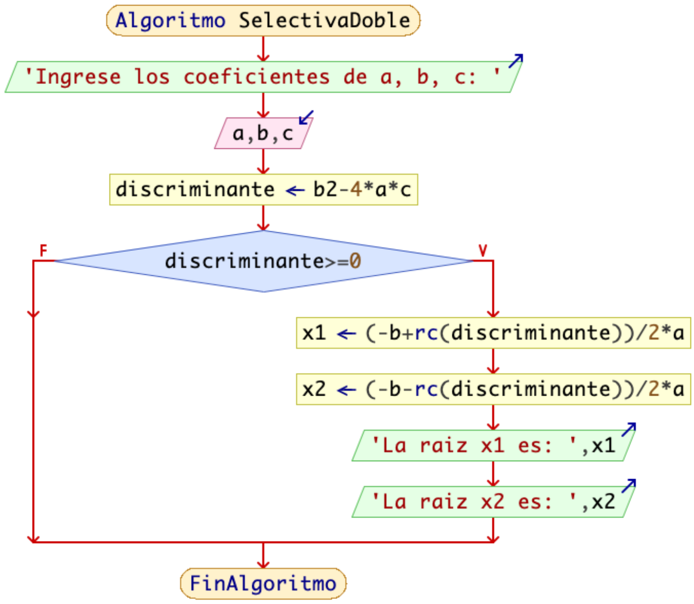

Figura 1.3: Diagrama de flujo: Ecuación cuadrática.

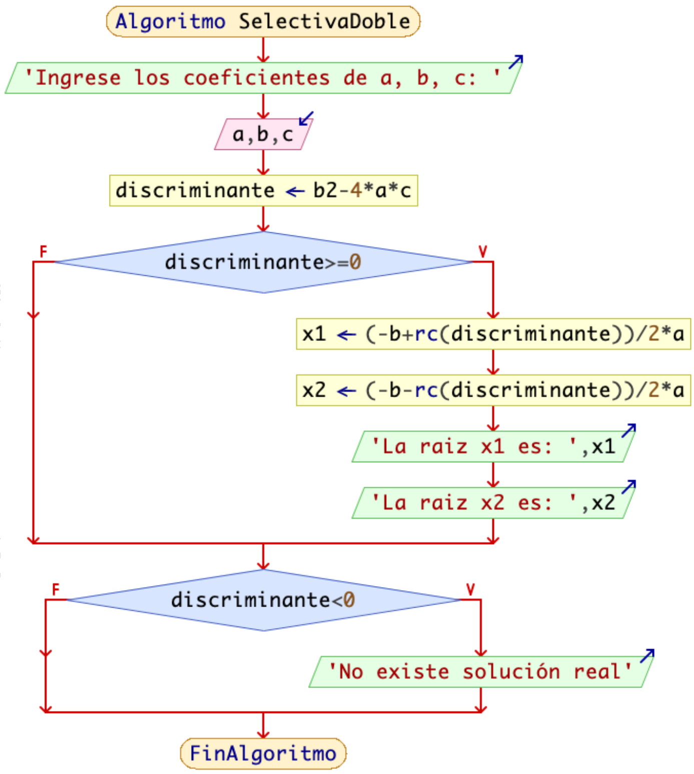

Figura 1.4: Diagrama de flujo: Ecuación cuadrática.

En la figura 1.5 se aprecia el pseudocódigo de la misma alternativa de solución. La estructura selectiva simple inicia en la línea 5 y finaliza en la línea 10. Por razones estéticas, el conjunto de instrucciones de la estructura selectiva simple se indenta, es decir se crea un sangrado. Todo el bloque se mueve unos caracteres a la derecha. Si bien es cierto a las herramientas que ejecuta código no les afecta la indentación de código, a los humanos sí. Es mucho más fácil entender un código bien indentado que un código sin indentación. Esto afecta mucho la etapa de corrección de errores y al mantenimiento de software.

```
Algoritmo SelectivaDoble
    Escribir 'Ingrese los coeficientes de a, b, c: '
    Leer a,b,c
    discriminante <- b2-4*a*c
    Si discriminante>=0 Entonces
        x1 <- (-b+rc(discriminante))/2*a
        x2 <- (-b-rc(discriminante))/2*a
        Escribir 'La raiz x1 es: ',x1
        Escribir 'La raiz x2 es: ',x2
    FinSi
    Si discriminante<0 Entonces
        Escribir 'No existe solución real'
    FinSi
FinAlgoritmo
```

Figura 1.5: Pseudocódigo: Ecuación cuadrática

En el programa 1.2 se puede apreciar la alternativa de solución en ANSI C. La estructura selectiva simple inicia en la línea 12 y finaliza en la línea 17. Al igual que en el pseudocódigo, el conjunto de instrucciones de la selectiva simple se encuentra indentado. Un aspecto importante de la sintaxis del ANSI C es que la expresión que representa a la condición en la estructura selectiva simple siempre va entre paréntesis. Los paréntesis no son necesarios para el caso del diagrama de flujo y pseudocódigo.

Programa 1.2: Verificación de discriminante

```
1  #include <stdio.h>
2  #include <math.h>
3
4  int main() {
5      int a, b, c;
6      double discriminante, x1, x2;
7
8      printf("Ingrese coeficientes a, b y c de la ecuacion: ");
9      scanf(" %d %d %d", &a, &b, &c);
10
11     discriminante = pow(b, 2) - 4 * a*c;
12     if (discriminante >= 0) {
13         x1 = (-b + sqrt(discriminante)) / 2 * a;
14         x2 = (-b - sqrt(discriminante)) / 2 * a;
15         printf("La raiz x1 es %lf\n", x1);
16         printf("La raiz x2 es %lf\n", x2);
17     }
18     if (discriminante <0)
19         printf("No existe solucion real");
20     return 0;
21 }
```

Para poner en práctica

¿Cuáles de las siguientes ecuaciones tiene solución real?

- $x^2 + x + 1 = 0$
- $-2x^2 + 3x + 4 = 0$
- $x^2 + 3x + 6 = 0$
- $-x^2 - 2x + 4 = 0$
- $2x^2 + 3x + 4 = 0$

1.5. Verificación de datos de entrada para cambio de billetes

Un cajero electrónico posee billetes de las siguientes denominaciones 50, 20 y 10. Dada una cantidad x de dinero, se desea conocer la menor cantidad de billetes que se requiere para obtener la cantidad x de dinero. En caso no se consiga la cantidad exacta, deberá indicarse además el monto faltante para llegar a x.

El problema en cuestión fue resuelto en la guía #2, pero en dicha solución se asumió que el usuario siempre ingresaría una cantidad $x > 0$. Nuevamente, el computador no tiene control sobre el usuario y por esto se hace necesario la realización de una verificación antes del procesamiento de los datos.

En el programa 1.3 se puede observar una alternativa de solución al problema incluyendo la verificación del monto ingresado por el usuario. La verificación se puede apreciar en la línea 10. Si el usuario no ingresa un monto "válido", no ejecutará el procesamiento.

Programa 1.3: Verificación del monto de las monedas

```
1  #include <stdio.h>
2  #include <math.h>
3
4  int main() {
5      int x, cant50, cant20, cant10, resto;
6
7      printf("Ingrese el monto: ");
8      scanf(" %d", &x);
9
10     if (x > 0) {
11         cant50 = x / 50;
12         x %= 50;
13         cant20 = x / 20;
14         x %= 20;
15         cant10 = x / 10;
16         resto = x % 10;
17         printf("Se requieren:\n");
18         printf("\t %d billete de 50\n", cant50);
19         printf("\t %d billete de 20\n", cant20);
20         printf("\t %d billete de 10\n", cant10);
21         printf("\t %d es el monto faltante\n", resto);
22     }
23     return 0;
24 }
```

Para poner en práctica

- Pruebe el programa introduciendo valores negativos o iguales a cero y vea qué sucede.
- ¿Qué debería agregar en el programa 1.3 para que se imprima `Ingrese cantidad >0` cuando la cantidad ingresada sea menor o igual a 0?
- Diseñe el algoritmo expresado en diagrama de flujo que corresponde al programa 1.3.
- Diseñe el algoritmo expresado en pseudocódigo que corresponde al programa 1.3.

1.6. Verificación de la base y operando en el cálculo de logaritmos

Sea b un número real positivo no nulo distinto de 1, y x otro número positivo no nulo. Se denomina logaritmo del número x en la base b, al exponente l al que debe elevarse la base b para obtener dicho número x. El logaritmo se expresa de la siguiente manera $\log_b x$ y si $\log_b x = l \leftrightarrow b^l = x$. Por ejemplo $\log_5 625 = 4$ ya que $625 = 5^4 = 5 \times 5 \times 5 \times 5$. El logaritmo es la operación inversa a la exponenciación.

Existen diversas propiedades de los logaritmos pero en esta sección centraremos la atención en el teorema de cambio de base. Según este teorema $\log_b x = \frac{\log_a x}{\log_a b}$. Esto significa que se puede basar el cálculo del logaritmo en cualquier base con el logaritmo de otra base. Este teorema es muy importante pues algunos lenguajes de programación solamente ofrecen la función logaritmo en una base determinada, típicamente el logaritmo natural cuya base es el número e. Aplicando el teorema de cambio de base y fijando la base en el número e, se tiene que $\log_b x = \frac{\ln x}{\ln b}$.

Para poder realizar el cálculo del logaritmo de un número x en una base b debe cumplirse que $b > 0 \wedge b \neq 1$ y $x > 0$. ¿Cómo se expresa dicha condición en ANSI C? En la guía #1 se estudiaron los operadores en ANSI C, la condición anteriormente descrita se puede expresar como `b>0 && b!=1 && x>0`. En el programa 1.4 se puede apreciar la validación antes de realizar el cálculo del logaritmo. Es bastante común que la expresión que representa a la condición en una estructura selectiva simple (y en realidad en todas las estructuras de control de flujo) contenga expresiones complejas que combinen tanto operadores relacionales como operadores lógicos.

Programa 1.4: Verificación de la base y número de un logaritmo

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

int main() {
    double base, numero, logaritmo;

    printf("Ingrese el numero y la base: ");
    scanf(" %lf %lf", &numero, &base);

    if (base > 0 && base != 1 && numero > 0) {
        logaritmo = log(numero) / log(base);
        printf("El logaritmo es %lf\n", logaritmo);
    }
    return 0;
}
```

Para poner en práctica

- ¿Qué debería agregar en el programa 1.4 para que se imprima `Dados inválidos` cuando los datos ingresados no cumplen las precondiciones del problema (i.e., $b > 0 \wedge b \neq 1 \wedge x > 0$)?
- Diseñe el algoritmo expresado en diagrama de flujo que corresponde al programa 1.4.
- Diseñe el algoritmo expresado en pseudocódigo que corresponde al programa 1.4.

1.7. ¿3 lados forman un triángulo?

Según el teorema de la desigualdad del triángulo *"La suma de las longitudes de cualesquiera de los lados de un triángulo es mayor que la longitud del tercer lado"*.

Basándose en el teorema de la desigualdad del triángulo, para verificar si 3 lados a, b y c forman en realidad un triángulo, deben cumplirse 3 condiciones: i) $a + b > c$, ii) $b + c > a$ y iii) $c + a > b$. ¿Cómo expresar estas 3 condiciones en ANSI C? Esta expresión se representará como una conjunción, es decir utilizando el operador `&&`. Para una mejor lectura de la condición $x + y > z$ agrupamos la suma usando paréntesis `(x+y)>z`. El programa que verifica si 3 lados pueden formar un triángulo se puede apreciar en el programa 1.5.

Programa 1.5: Verificación de los lados de un triángulo

```c
#include <stdio.h>

int main() {
    int a, b, c, lados_forman_triangulo;

    printf("Ingrese los lados de un triangulo: ");
    scanf(" %d %d %d", &a, &b, &c);

    lados_forman_triangulo = (a+b)>c && (a+c)>b && (b+c)>a;
    if (lados_forman_triangulo) {
```

```c
        printf("Los lados forman un triangulo.\n");
    }
    if (!lados_forman_triangulo) {
        printf("Los lados no forman un triangulo.\n");
    }
    return 0;
}
```

Para poner en práctica

- Si cambia la condición de la selectiva por $a + b > c$ && $a + c > b$ && $b + c > a$ ¿El programa sigue funcionando?.
- Diseñe el algoritmo expresado en diagrama de flujo que corresponde al programa 1.5.
- Diseñe el algoritmo expresado en pseudocódigo que corresponde al programa 1.5.

1.8. ¿Es el año bisiesto?

Si queremos entender por qué existen los años bisiestos debemos fijarnos en el movimiento de la Tierra alrededor del Sol: nuestro planeta rota $365,24219$ veces durante una órbita completa alrededor del astro, por tanto un año dura 365 días, 5 horas, 48 minutos y 56 segundos, no 365.

Al emperador Julio César se le ocurrió crear el año bisiesto. Si cada año nosotros contamos esos 365 días, perdemos esas 5 horas que deberemos recuperar. Durante tres años contamos esos 365 y al cuarto recuperamos el día que falta, los 29 días que tiene febrero, el año bisiesto.

El año bisiesto tiene una buena explicación. Si no añadiéramos un día completo cada cuatro años, las estaciones acabarían descompasadas del calendario, de tal manera que después de unos 700 años, en el hemisferio norte la Navidad caería en mitad del verano. Al revés, en el hemisferio sur[4].

¿Cómo se puede determinar si un año es bisiesto? Un año bisiesto si el número que lo representa es divisible entre 4, salvo que sea año secular -último de cada siglo, terminado en 00-, en cuyo caso también ha de ser divisible entre 400.

Si tenemos las siguientes preposiciones:

- p: El número que representa al año es divisible entre 4
- q: El número que representa al año es divisible entre 100
- r: El número que representa al año es divisible entre 400

La expresión lógica que permite determinar si un año es bisiesto es $p \wedge (\neg q \vee r)$. En el programa 1.6 se puede apreciar la implementación de esta expresión lógica en ANSI C.

Programa 1.6: Verificación de un año bisiesto

```c
#include <stdio.h>

int main() {
    int anho, p, q, r, es_bisiesto;

    printf("Ingrese el anho: ");
    scanf(" %d", &anho);

    p = (anho % 4) == 0;
    q = (anho % 100) == 0;
    r = (anho % 400) == 0;
    es_bisiesto = p && (!q || r);
```

[4] Texto tomado del a URL https://elpais.com/elpais/2016/02/29/actualidad/1456703053_016861.html

```

    if (es_bisiesto) {
        printf("El anho es bisiesto.\n");
    }
    if (!es_bisiesto) {
        printf("El anho no es bisiesto.\n");
    }
    return 0;
}
```

1.9. Ejercicios propuestos

En la pregunta 1.9.1 se presenta una serie de programas sintácticamente correctos pero que contienen errores lógicos comunes en el uso de operadores y la estructura algorítmica selectiva en el lenguaje ANSI C. Para cada uno de estos programa se le solicita que los analice y reflexione sobre el resultado que debería obtener. Luego ejecute el programa usando un IDE de ANSI C para verificar el resultado.

Para los demás ejercicios propuestos, se solicita que elabore el correspondiente algoritmo representado tanto en diagrama de flujo como en pseudocódigo así como la implementación de un programa en ANSI C conforme a los temas revisados en las guías #1 y #2 del curso Fundamentos de Programación.

1.9.1. Análisis de programas

Programa 1.7: Análisis de programa: Operador de asignación

```
1 #include <stdio.h>
2
3 int main() {
4     int a=10;
5     if (a=5){
6         printf("El valor de a= %d\n", a);
7     }
8     return 0;
9 }
```

Para analizar

- ¿El valor que se imprime es el valor esperado?
- ¿Qué hace la línea 5?
- Cambie la línea 5 por `if(a==5)`. ¿Qué cambia en el programa?

Programa 1.8: Análisis de programa: Operador relacional de comparación

```
#include <stdio.h>

int main() {
    int a=10, b=10, c=10;
    if (a==b==c){
        printf("Los 3 numeros son iguales\n");
    }
    return 0;
}
```

Para analizar

- ¿El valor que se imprime es el valor esperado?
- ¿El ANSI C permite la comparación múltiple?
- Si quisiéramos verificar que los 3 números sean iguales, ¿Cómo debería ser la condición la estructura selectiva `if` en la línea 5?

Programa 1.9: Análisis de programa: Operador relacional de comparación

```c
#include <stdio.h>

int main() {
    int a=1, b=1, c=1;
    if (a==b==c){
        printf("Los 3 numeros son iguales\n");
    }
    return 0;
}
```

Para analizar

- ¿Por qué en este programa pareciera que el lenguaje ANSI C sí implementa la comparación múltiple?

Programa 1.10: Análisis de programa: Operador aritmético de suma

```c
#include <stdio.h>

int main() {
    int a=5, b=-5;
    if (a+b){
        printf("La suma es cero\n");
    }
    return 0;
}
```

Para analizar

- ¿Cuál es el error lógico en este programa?
- Corrija el error lógico usando el operador de comparación (==).
- Corrija el error lógico usando el operador de negación (`!`).

Programa 1.11: Análisis de programa: Operador aritmético de suma

```c
#include <stdio.h>

int main() {
    int a=-3, b=-5;
    if (a+b){
        printf("La suma es diferente de cero\n");
    }
    return 0;
}
```

Para analizar

- ¿Este programa tiene algún error lógico?
- ¿Es necesario cambiar la condición de la línea 5 por `(a+b)!=0`?
- Si a la variable `b` se le hubiera asignado el valor de 3. ¿Qué se imprime?

1.9.2. El valor absoluto

Dado determinado número real, se le pide que retorne el valor absoluto de dicho número.

Recordar que:

El valor absoluto de un número real x, es la magnitud numérica de dicho número sin importar su signo.

$$\mid x \mid = \begin{cases} x & \text{si } x \geq 0 \\ -x & \text{si } x < 0 \end{cases}$$

Restricciones

- Para la implementación en ANSI C no podrá usar la función `fabs`.
- Para la implementación en PseInt no podrá usar la función `abs`.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $x = -1.6$, entonces se debe imprimir 1.6.
- Si $x = 2.5$, entonces se debe imprimir 2.5.
- Si $x = -3.9$, entonces se debe imprimir 3.9.

1.9.3. La función techo

Dado determinado número real, se le pide que retorne el techo de dicho número.

Recordar que:

La función techo es una función que recibe como parámetro a un número real x ($x \in \mathbb{R}$) y retorna el mínimo entero y ($y \in \mathbb{Z}$) más próximo que es mayor o igual a x.

$$techo(x) = \lceil x \rceil = min\{k \in \mathbb{Z} | x \leq k\}$$

Sugerencia

Realice el siguiente procedimiento:

- Lea el número real x en una variable.
- Obtenga la parte entera del número real x usando la función `trunc`. Almacene este valor en una variable. En ANSI C la declaración de la función `trunc` se encuentra en el archivo de cabecera `math`.
- Obtenga la parte fraccionaria del número real x. Para esto reste a x la parte entera hallada en el paso anterior. Almacene este valor en una variable.
- Si la parte fraccionaria del número real x es mayor que cero, incremente en uno el valor de la parte entera.
- El techo de x se encontrará en la variable en donde se almacenó la parte entera.

Restricciones

- Para la implementación en ANSI C no podrá usar la función `ceil`.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $x = -3.4$, entonces se debe imprimir -3.
- Si $x = -0.7$, entonces se debe imprimir 0.
- Si $x = 0$, entonces se debe imprimir 0.
- Si $x = 1.4$, entonces se debe imprimir 2.
- Si $x = 3.8$, entonces se debe imprimir 4.
- Si $x = 4.1$, entonces se debe imprimir 5.

1.9.4. La función piso

Dado determinado número real, se le pide que retorne el piso de dicho número.

Recordar que:

La función piso es una función que recibe como parámetro a un número real x ($x \in \mathbb{R}$) y retorna el máximo entero y ($y \in \mathbb{Z}$) más próximo que es menor o igual a x.

$$piso(x) = \lfloor x \rfloor = max\{k \in \mathbb{Z} | k \leq x\}$$

Sugerencia

Realice el siguiente procedimiento:

- Lea el número real x en una variable.
- Si el número real x es mayor o igual a cero, la función piso se obtiene aplicando la función `trunc`. En ANSI C la declaración de la función `trunc` se encuentra en el archivo de cabecera `math`.
- Si el número real x es menor que cero:
  - Obtenga la parte entera del número real x usando la función `trunc`. Almacene este valor en una variable.
  - Obtenga la parte fraccionaria del número real x. Para esto reste a x la parte entera hallada en el paso anterior. Almacene este valor en una variable.
  - Si la parte fraccionaria del número real x es diferente que cero, decremente en uno el valor de la parte entera.
  - El piso de x se encontrará en la variable en donde se almacenó la parte entera.

Restricciones

- Para la implementación en ANSI C no podrá usar la función `floor`.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $x = -3.4$, entonces se debe imprimir -4.
- Si $x = -0.7$, entonces se debe imprimir -1.
- Si $x = 0$, entonces se debe imprimir 0.
- Si $x = 1.4$, entonces se debe imprimir 1.
- Si $x = 3.8$, entonces se debe imprimir 3.
- Si $x = 4.1$, entonces se debe imprimir 4.

1.9.5. ¿Cuántos días tiene determinado año?

Dado determinado número entero que representa un año, imprima la cantidad de días que posee dicho año. Recuerde que los años bisiestos tienen un día adicional, el 29 de febrero.

Recordar que:

Un año es bisiesto si el número que lo representa es divisible entre 4, salvo que sea año secular –último de cada siglo, terminado en 00–, en cuyo caso también ha de ser divisible entre 400.

Dadas las siguientes preposiciones:

- p: El número que representa al año es divisible entre 4.
- q: El número que representa al año es divisible entre 100.
- r: El número que representa al año es divisible entre 400.

La expresión lógica que permite determinar si un año es bisiesto es $p \wedge (\neg q \vee r)$.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $año = 2016$, entonces se debe imprimir 366.
- Si $año = 2019$, entonces se debe imprimir 365.
- Si $año = 2020$, entonces se debe imprimir 366.
- Si $año = 2013$, entonces se debe imprimir 365.
- Si $año = 2024$, entonces se debe imprimir 366.

1.9.6. ¿Cuántos paquetes utilizar?

Se requiere colocar cierta cantidad l de lapiceros en paquetes de capacidad c. ¿Cuántos paquetes se deberán utilizar para este fin?

Restricciones

- Para la implementación en ANSI C no podrá usar la función `ceil`.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $l = 12$ y $c = 15$, deberá retornar `Se requiere 1 paquete(s)`.
- Si $l = 20$ y $c = 6$, deberá retornar `Se requiere 4 paquete(s)`.
- Si $l = 25$ y $c = 5$, deberá retornar `Se requiere 5 paquete(s)`.
- Si $l = 30$ y $c = 4$, deberá retornar `Se requiere 8 paquete(s)`.

1.9.7. ¿Cuánto cuesta la matrícula?

En determinada universidad la matricula del semestre tiene un costo de c soles. A los alumnos que se encuentran en facultad se les realiza un descuento del d %. Se le solicita que dado el costo c de la matrícula, el descuento d y el número n del ciclo en que se encuentra determinado alumno, calcule y presente el monto que debe pagar dicho alumno. Asuma que los alumnos se encuentran en facultad si es que el ciclo en que se encuentra es mayor o igual a 5.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $c = 1500$, $d = 15\,\%$ y $n = 2$, deberá imprimir `Deberá pagar 1500 soles`.
- Si $c = 1500$, $d = 15\,\%$ y $n = 5$, deberá imprimir `Deberá pagar 1275 soles`.
- Si $c = 2000$, $d = 25\,\%$ y $n = 4$, deberá imprimir `Deberá pagar 2000 soles`.
- Si $c = 2000$, $d = 25\,\%$ y $n = 8$, deberá imprimir `Deberá pagar 1500 soles`.

1.9.8. Longitud de onda

La nota musical `LA` tiene una frecuencia, por convenio internacional, de 440 Hz. Esta nota se propaga con una velocidad de 340 m/s en el aire y con una velocidad de 1400 m/s en el agua. Se le pide que lea un caracter que represente al medio en donde se propaga la nota musical (`A` para el aire y `G` para el agua) e imprima la longitud de la onda en dicho medio.

Recordar que:

$$\lambda = \frac{v}{f}$$

donde:

- λ es la longitud de onda.
- v es la velocidad de propagación.
- f es la frecuencia de la onda.

Además: $1\ Hz = \frac{1}{s}$

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si el medio ingresado es `A`, entonces deberá retornar $\lambda_{agua} \approx 3.181818182$.
- Si el medio ingresado es `G`, entonces deberá retornar $\lambda_{aire} \approx 0.772727273$.

1.9.9. Identidades trigonométricas

Se desea probar algunas identidades trigonométricas a través del computador. Para esto deberá de leer un ángulo en sexagesimal y luego calcular el valor de la identidad con el ángulo ingresado, el programa deberá imprimir `Se cumple identidad` en caso se cumpla la identidad trigonométrica y `No se cumple identidad` en caso contrario. Las identidades a probar serán las siguientes:

- $sen(2\theta) = 2sen(\theta)cos(\theta)$
- $cos(2\theta) = cos^2(\theta) - sen^2(\theta)$
- $cos(2\theta) = 1 - 2sen^2(\theta)$

Recordar que:

$360° = 2\pi$ radianes

Sugerencia para PSeInt

- Utilice la función `sen` para obtener el valor del seno un ángulo en radianes.
- Utilice la función `cos` para obtener el valor del coseno un ángulo en radianes.

Sugerencia para ANSI C

- Utilice la función `sin` cuyo prototipo se encuentra en el archivo de cabecera `math` para obtener el valor del seno un ángulo en radianes.
- Utilice la función `cos` cuyo prototipo se encuentra en el archivo de cabecera `math` para obtener el valor del coseno un ángulo en radianes.

Casos de prueba

Deberá retornar una salida similar a la siguiente:

```
Se cumple la identidad sen(2x)=2sen(x)cos(x)
Se cumple la identidad cos(2x)=cos^2(x)-sen^2(x)
Se cumple la identidad cos(2x)=1-2sen^2(x)
```

1.9.10. La serie de Gregory

Una de las formas de calcular el número π es mediante la serie de Gregory, según esta serie:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots = \sum_{n=1}^{\infty} (-1)^{(n+1)} \frac{1}{2n-1}$$

Se le pide que dado un número n que representa el número del término de la serie, retorne dicho término.

Restricciones

- Para la implementación en ANSI C no podrá usar la función `pow`.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $n = 1$, se deberá retornar 1
- Si $n = 3$, se deberá retornar 0.2
- Si $n = 4$, se deberá retornar ≈ -0.142857143

1.9.11. Suma de los cuadrados de los números en un rango

Se desea calcular la suma de los cuadrados de los números naturales que existen en un rango $[a..b]$ en donde tanto a como b son números naturales mayores que 0 y se debe cumplir además que $a < b$. Por ejemplo si $a = 5$ y $b = 10$, se deberá retornar $sumatoria = 5^2 + 6^2 + 7^2 + 8^2 + 9^2 + 10^2 = 25 + 36 + 49 + 64 + 81 + 100 = 355$.

Recordar que:

La suma de los cuadrados de los n primeros números naturales se puede calcular mediante la siguiente serie notable:

$$\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6}$$

Sugerencia

- Utilice una selectiva simple para verificar que los valores de a y b cumplen la condición del problema.
- Utilice la serie notable para calcular la suma de los cuadrados de los n primeros números naturales tomando como n el valor de b.
- Utilice la serie notable para calcular la suma de los cuadrados de los n primeros números naturales tomando como n el valor de $a - 1$.
- Obtenga la diferencia entre ambas sumatorias calculadas.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si el rango fuese [3..8] se deberá imprimir 199.
- Si el rango fuese [4..4] se deberá imprimir 0.
- Si el rango fuese [7..11] se deberá imprimir 415.
- Si el rango fuese [8..3] se deberá imprimir 0.
- Si el rango fuese [14..19] se deberá imprimir 1651.

1.9.12. La ecuación de la recta

Una recta se puede representar mediante una ecuación de la siguiente forma $y = mx + b$, en donde el valor de m corresponde a la pendiente y el valor de b corresponde al punto de intercepción en la ordenada. Se desea determinar si dado determinado punto $P(x, y)$, este punto pertenece o no a una recta.

Sugerencia

- Lea en una variable el valor de la pendiente m y en otra variable el valor del punto de intercepción en la ordenada b.
- Lea en una variable el valor de la abscisa x y en otra variable la ordenada y.
- Utilizando la ecuación de la recta $mx + b$ calcule el valor que debería tener la ordenada para x. Almacene este valor en una variable denominada y_recta.
- Compare el valor y con el valor de y_recta, si son iguales imprima el texto `El punto forma parte de la recta`.
- Compare el valor y con el valor de y_recta, si no son iguales imprima el texto `El punto no forma parte de la recta`.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $m = 5$, $b = 3$, los siguientes son puntos de la recta $(1, 8)$, $(4, 23)$, $(6, 33)$
- Si $m = -3$, $b = 0$, los siguientes son puntos de la recta $(-3, 9)$, $(1, -3)$, $(10, -30)$
- Si $m = -1$, $b = 10$, los siguientes son puntos de la recta $(0, 10)$, $(5, 5)$, $(10, 0)$

1.9.13. La ecuación de la circunferencia

Una circunferencia se puede describir por la ecuación $(x-a)^2 + (x-b)^2 = r^2$, donde el punto (a, b) representa el centro de la circunferencia y r representa el radio de la circunsferencia. Se pide que dada una ecuación de una circunferencia y un punto (x, y) en un plano cartesiano, determine si el punto se encuentra dentro del círculo delimitado por la circunferencia descrita por la ecuación.

Sugerencia

- Lea el punto (a, b) que representa el centro.
- Lea el radio r.
- Lea el punto (x, y)
- Halle el valor de $(x-a)^2 + (x-b)^2$, si este valor es menor o igual que r^2 imprima el texto `El punto está dentro del círculo`.
- Halle el valor de $(x-a)^2 + (x-b)^2$, si este valor es mayor que r^2 imprima el texto `El punto no está dentro del círculo`.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $a = 2$, $b = 3$, $r = 5$, $x = 4$ e $y = 6$, se deberá imprimir `El punto está dentro del círculo.`
- Si $a = 2$, $b = 3$, $r = 5$, $x = 9$ e $y = 3$, se deberá imprimir `El punto no está dentro del círculo.`
- Si $a = 8$, $b = 9$, $r = 3$, $x = 9$ e $y = 10$, se deberá imprimir `El punto está dentro del círculo.`
- Si $a = 8$, $b = 9$, $r = 3$, $x = 14$ e $y = 10$, se deberá imprimir `El punto no está dentro del círculo.`

1.9.14. Cálculo del Interés simple

Un egresado de la Pontificia Universidad Católica del Perú posee un capital C de 50.000 soles y desea depositarlos en un banco a un plazo n de 3 años. La tasa de interés simple i que le ofrecen es de 30 % anual. Si el egresado en cuestión desea tener al final un saldo de por lo menos 90.000 soles, ¿Le conviene hacer el depósito en el banco?

Recordar que:

El saldo final S se calcula de la siguiente manera:

$$S = C(1 + n * i)$$

Donde:

- n está expresado en años.
- i es el interés anual, valor del porcentaje multiplicado por $0,01$.

Sugerencia

- Calcule el saldo final.
  - Si el saldo final es mayor que el monto que se desea obtener se concluye que la operación es conveniente.
  - Si el saldo final es menor o igual que el monto que se desea obtener se concluye que la operación no es conveniente.

Sugerencia

A pesar que este problema puede ser resuelto de forma particular, generalice su solución para:

- Leer el capital C.
- Leer el plazo n.
- Leer la tasa de interés simple i.
- Leer el saldo final deseado S.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $c = 50000$ soles, $n = 3$ años, $i = 30\,\%$ y el saldo final deseado $S = 90000$ soles, se debe imprimir `Conviene depositar en el banco`.
- Si $c = 100000$ soles, $n = 5$ años, $i = 32\,\%$ y el saldo final deseado $S = 250000$ soles, se debe imprimir `No conviene depositar en el banco`.
- Si $c = 70000$ soles, $n = 4$ años, $i = 25\,\%$ y el saldo final deseado $S = 135000$ soles, se debe imprimir `Conviene depositar en el banco`.

1.9.15. ¿Se mueve o no se mueve la caja?

Sobre un plano horizontal se tiene una caja que pesa 35 N. Se sabe además que se tiene una coeficiente de rozamiento estático de $\mu_s = 0.5$. Si se le aplica una fuerza de 10 N, ¿la caja se mueve?

Recordar que:

La fuerza de rozamiento estático máximo se calcula de la siguiente manera:

$$fs\_max = \mu_s \times N$$

Donde:

- μ_s es el coeficiente de rozamiento estático.
- N es la fuerza normal.

Sugerencia

- Calcule la fuerza de rozamiento estático máximo. Recuerde que la fuerza de rozamiento estático máximo equivale a la fuerza mínima para iniciar un movimiento.
- Si la fuerza aplicada es mayor que la fuerza de rozamiento estático máximo, la caja se moverá.
- Si la fuerza aplicada es menor o igual que la fuerza de rozamiento estático máximo, la caja no se moverá.

Sugerencia

A pesar que este problema puede ser resuelto de forma particular, generalice su solución para:

- Leer el peso de una caja en N.
- Leer el coeficiente de rozamiento estático μ_s.
- Leer la fuerza que se aplicará en N.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $peso = 35N$, $\mu_s = 0.5$ y la fuerza aplicada es $10N$, se debe imprimir `No se mueve la caja`.
- Si $peso = 35N$, $\mu_s = 0.5$ y la fuerza aplicada es $18N$, se debe imprimir `Sí se mueve la caja`.
- Si $peso = 50N$, $\mu_s = 0.3$ y la fuerza aplicada es $16N$, se debe imprimir `Sí se mueve la caja`.
- Si $peso = 25N$, $\mu_s = 0.8$ y la fuerza aplicada es $19N$, se debe imprimir `No se mueve la caja`.

1.9.16. ¿Cómo puedo saber si mi peso es normal?

El Índice de Masa Corporal (IMC) es una razón matemática que asocia la masa y la talla de un individuo. Se calcula de la siguiente manera:

$$IMC = \frac{masa}{estatura^2}$$

Donde la *masa* se expresa en kg y la *estatura* en m^2. De acuerdo a la Organización Mundial de la Salud, si el IMC se encuentra entre los valores de 18.50 y 24.90, se dice que el peso es normal. Dado el valor de la *masa* en kg y la *estatura* en m^2 determine si el peso es normal.

Sugerencia

- Lea en una variable el valor de la *masa*.
- Lea en una variable el valor de la *estatura*.
- Con los valores leídos calcule el IMC.
- Si el valor calculado se encuentra en el rango [18.50..24.90] imprima el texto `El peso es normal`.
- Si el valor calculado no se encuentra en el rango [18.50..24.90] imprima el texto `El peso no es normal`.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $masa = 86\ kg$ y $estatura = 1.70\ m$, se deberá imprimir `El peso no es normal`.
- Si $masa = 71\ kg$ y $estatura = 1.70\ m$, se deberá imprimir `El peso es normal`.
- Si $masa = 75\ kg$ y $estatura = 1.65\ m$, se deberá imprimir `El peso no es normal`.
- Si $masa = 79\ kg$ y $estatura = 1.80\ m$, se deberá imprimir `El peso es normal`.

1.9.17. Distancia más cercana

Dados 3 puntos A, B, C en el plano cartesiano ($P_A(x_A, y_A)$, $P_B(x_B, y_B)$, $P_C(x_C, y_C)$) y un punto X ($P_X(x_X, y_X)$), se le solicita que imprima el punto que se encuentra más cercano a el punto X. Asuma que las distancias del punto X a los demás puntos serán siempre diferentes.

Recuerdar que:

Recuerde que la distancia euclidiana d se calcula de la siguiente manera:

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

Casos de prueba

- Si $P_A = (3, 5)$, $P_B = (4, 2)$, $P_C = (7, 7)$ y $P_X = (4, 4)$, debe retornar `El punto más cercano es A.`
- Si $P_A = (3, 5)$, $P_B = (4, 2)$, $P_C = (7, 7)$ y $P_X = (2, 1)$, debe retornar `El punto más cercano es B.`
- Si $P_A = (3, 5)$, $P_B = (4, 2)$, $P_C = (7, 7)$ y $P_X = (8, 7)$, debe retornar `El punto más cercano es C.`

Sugerencia

Realice los cambios necesarios en el programa para que ahora imprima el punto más lejano al punto X. Una vez hecho los cambios, utilice los siguientes datos para probar su solución.

- Si $P_A = (3, 5)$, $P_B = (4, 2)$, $P_C = (7, 7)$ y $P_X = (11, 3)$, debe retornar `El punto más lejano es A.`
- Si $P_A = (3, 5)$, $P_B = (4, 2)$, $P_C = (7, 7)$ y $P_X = (2, 1)$, debe retornar `El punto más lejano es C.`
- Si $P_A = (3, 5)$, $P_B = (4, 2)$, $P_C = (7, 7)$ y $P_X = (8, 7)$, debe retornar `El punto más lejano es B.`

1.9.18. Clasificación de un triángulo según sus lados

Dados 3 números reales que representan los lados de un triángulo, se le solicita que luego de verificar si los 3 lados forman un triángulo, imprima el tipo de triángulo según sus lados.

Recordar que:

La clasificación de triángulos según sus lados es como sigue:

Equilátero si sus 3 lados son iguales.

Isósceles si 2 de sus lados, y solo 2, son iguales.

Escaleno si los 3 lados son diferentes.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $l1 = 3$, $l2 = 4$ y $l3 = 5$, deberá imprimirse `Triángulo Escaleno`.
- Si $l1 = 3$, $l2 = 3$ y $l3 = 2$, deberá imprimirse `Triángulo Isósceles`.
- Si $l1 = 3$, $l2 = 2$ y $l3 = 3$, deberá imprimirse `Triángulo Isósceles`.
- Si $l1 = 2$, $l2 = 3$ y $l3 = 3$, deberá imprimirse `Triángulo Isósceles`.
- Si $l1 = 10.5$, $l2 = 10.5$ y $l3 = 10.5$, deberá imprimirse `Triángulo Equilátero`.
- Si $l1 = 1$, $l2 = 10$ y $l3 = 1$, deberá imprimirse `Los lados ingresados no forman un triángulo`.

1.9.19. Área de un triángulo isósceles

Dados 3 lados que forman un triángulo isósceles, se le solicita que determine el área de dicho triángulo. Asuma que los 3 lados forman un triángulo. Asuma que siempre tendrá 2 lados iguales y uno diferente. Además el usuario podrá ingresar los lados en cualquier orden.

Recordar que:

El área de un triángulo isóceles se puede calcular usando la siguiente fórmula:

$$area = \frac{b}{4} \times \sqrt{4a^2 - b^2}$$

Donde:

- a es uno de los lados iguales.
- b es el lado diferente.

Restricciones

- No podrá aplicar la fórmula de Herón.

Sugerencia

- Halle primero el lado que se repite y asígnelo a la variable a.
- Halle luego el lado que no se repite y asígnelo a la variable b.
- Aplique la fórmula para el triángulo isósceles.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $l1 = 3$, $l2 = 3$, $l3 = 2$, entonces el área debe ser ≈ 2.828427125.
- Si $l1 = 3$, $l2 = 2$, $l3 = 3$, entonces el área debe ser ≈ 2.828427125.
- Si $l1 = 2$, $l2 = 3$, $l3 = 3$, entonces el área debe ser ≈ 2.828427125.
- Si $l1 = 5$, $l2 = 4$, $l3 = 5$, entonces el área debe ser ≈ 9.16515139.

1.9.20. Operaciones con fracciones

Una fracción se puede representar usando 2 variables, una para el numerador y otra para el denominador. Se solicita que lea dos fracciones y una operación (`+`, `-`, `*`, `/`) y retorne el resultado de aplicar la operación a las dos fracciones.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si la fracción 1 es $\frac{1}{2}$, la fracción 2 es $\frac{1}{3}$ y la operación es $+$, se deberá retornar $\frac{5}{6}$
- Si la fracción 1 es $\frac{1}{2}$, la fracción 2 es $\frac{1}{3}$ y la operación es $-$, se deberá retornar $\frac{1}{6}$
- Si la fracción 1 es $\frac{2}{7}$, la fracción 2 es $\frac{5}{3}$ y la operación es $*$, se deberá retornar $\frac{10}{21}$
- Si la fracción 1 es $\frac{2}{7}$, la fracción 2 es $\frac{5}{3}$ y la operación es $/$, se deberá retornar $\frac{6}{35}$

1.9.21. Los números Armstrong

Un número Armstrong, también llamado número narcisista, es todo aquel número que es igual a la suma de cada uno de sus dígitos elevado al número total de dígitos.

A continuación siguen algunos ejemplos de números Armstrong:

- $371 = 3^3 + 7^3 + 1^3$. Total de dígitos 3.
- $8208 = 8^4 + 2^4 + 0^4 + 8^4$. Total de dígitos 4.
- $4210818 = 4^7 + 2^7 + 1^7 + 0^7 + 8^7 + 1^7 + 8^7$. Total de dígitos 7.

Dado un número n de exactamente 3 cifras, determine si dicho número es Armstrong. Si el número no tiene exactamente 3 cifras, no realice ningún procesamiento.

Sugerencia

- Para determinar si un número tiene exactamente 3 cifras, simplemente verifique que sea mayor o igual que 100 y menor o igual que 999.
- Para obtener las cifras de un número n de 3 cifras siga el siguiente algoritmo:
  - Obtenga el dígito de la unidades utilizando la operación de módulo del número n entre 10 (si $n = 153$, `153%10=3`).
  - Actualice el número n con el valor que resulta de dividir n entre 10 (si $n = 153$, `n=153/10=15`).
  - Obtenga el dígito de la decenas utilizando la operación de módulo del nuevo número n entre 10 (si $n = 15$, `15%10=5`).
  - Actualice el número n con el valor que resulta de dividir n entre 10 (si $n = 15$, `n=15/10=1`).
  - El dígito de las centenas estará en la variable n.

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $n = 153$ se deberá imprimir `El número es Armstrong` $(153 = 1^3 + 5^3 + 3^3)$.
- Si $n = 271$ se deberá imprimir `El número no es Armstrong` $(352 = 2^3 + 7^3 + 1^3)$.
- Si $n = 370$ se deberá imprimir `El número es Armstrong` $(370 = 3^3 + 7^3 + 1^0)$.
- Si $n = 435$ se deberá imprimir `El número no es Armstrong` $(216 = 4^3 + 3^3 + 5^3)$.
- Si $n = 371$ se deberá imprimir `El número es Armstrong` $(371 = 3^3 + 7^3 + 1^3)$.

1.9.22. Números palíndromos

Un número natural es un palíndromo si se lee igual de izquierda a derecha y de derecha a izquierda. Son ejemplos de números palíndromos 1, 33, 141, 45654, 13531. Una forma para calcular un número palíndromo, pero que no siempre funciona, es:

- Seleccionar un número n.
- Invertir el número n
- Sumar al número invertido el valor de n
- El resultado de la suma es un candidato para ser palíndromo.

Este algoritmo algunas veces obtiene un número palíndromo. Por ejemplo:

```
n=13
inverso de n=31
suma = 13 + 31 = 44

n=65
inverso de n=56
suma 65 + 56 = 121
```

Dado un número n de exactamente 2 cifras sin incluir el 0, determine si con el algoritmo antes presentado se puede encontrar un número palíndromo. En caso de encontrarlo deberá imprimir dicho número palíndromo.

Sugerencia

Para invertir un número n de 2 cifras siga el siguiente algoritmo:

- Obtenga el dígito de la unidades utilizando la operación de módulo del número n entre 10 (si $n = 13$, `13%10=3`).
- Obtenga el dígito de la decenas utilizando la operación de división entera del número n entre 10 (si $n = 13$, `13/10=1`).
- Multiplique el dígito de las unidades por 10 y súmele el dígito de las decenas (si $n = 13$, `3*10+1=31`)
- El resultado de la suma será el número invertido (si $n = 13$, el invertido es 31).

Sugerencia

Para invertir un número n de 3 cifras siga el siguiente algoritmo:

- Obtenga el dígito de la unidades utilizando la operación de módulo del número n entre 100 (si $n = 156$, `156 %100=6`).
- Obtenga un nuevo número m que resulta de dividir n entre 10 (si $n = 156$, `m=156/10=15`).
- Invierta el número m que ahora tiene dos cifras (si $m = 15$ el invertido será 51). Como el número tiene 2 cifras puede utilizar el algoritmo mencionado previamente.
- Multiplique el dígito de la unidades por 100 y súmele el número m (`6*100+51=651`)
- El resultado de la suma será el número invertido (si $n = 156$, el invertido es 651).

Casos de prueba

Utilice los siguientes datos para probar su solución.

- Si $n = 15$ se deberá imprimir `Se encontró el palíndromo 66`.
- Si $n = 64$ se deberá imprimir `No se encontró el palíndromo`.
- Si $n = 25$ se deberá imprimir `Se encontró el palíndromo 77`.
- Si $n = 78$ se deberá imprimir `No se encontró el palíndromo`.