Pontificia Universidad Católica del Perú

Estudios Generales Ciencias

1INF01 - Fundamentos de Programación

Guía de laboratorio #2

Elaboración de programas con estructuras algorítmicas selectivas

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

10 de septiembre de 2021

Índice general

Historial de revisiones 1

Siglas 2

1. Guía de Laboratorio #2 3

1.1. Introducción . . . 3
1.2. Materiales y métodos . . . 3
1.3. Estructura Algorítmica Selectiva . . . 3
1.3.1. Representación de la Estructura Algorítmica Selectiva Simple . . . 4
1.3.2. Representación de la Estructura Algorítmica Selectiva Doble . . . 5
1.4. Cálculo de las raíces de ecuaciones cuadráticas con una variable . . . 6
1.4.1. Verificación del discriminante . . . 6
1.4.2. Cálculo de las raíces complejas . . . 9
1.5. Verificación de datos de entrada para el problema de cambio de billetes . . . 11
1.6. Verificación de la base y operando en el cálculo de logaritmos . . . 12
1.7. ¿3 lados forman un triángulo? . . . 13
1.8. ¿Es el año bisiesto? . . . 14

2. Ejercicios propuestos 16

2.1. Nivel Básico . . . 16
2.1.1. Análisis de programas . . . 16
2.1.2. El valor absoluto . . . 18
2.1.3. La función techo . . . 18
2.1.4. La función piso . . . 19
2.1.5. ¿Cuántos días tiene determinado año? . . . 20
2.1.6. ¿Cuántos paquetes utilizar? . . . 20
2.1.7. ¿Cuánto cuesta la matrícula? . . . 21
2.1.8. Longitud de onda . . . 21
2.2. Nivel Intermedio . . . 22
2.2.1. Identidades trigonométricas . . . 22

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

2.2.2. La serie de Gregory . . . 23
2.2.3. Suma de los cuadrados de los números en un rango . . . 23
2.2.4. La ecuación de la recta . . . 24
2.2.5. La ecuación de la circunferencia . . . 24
2.2.6. Cálculo del Interés simple . . . 25
2.2.7. ¿Se mueve o no se mueve la caja? . . . 26
2.2.8. ¿Cómo puedo saber si mi peso es normal? . . . 27
2.3. Nivel Avanzado . . . 28
2.3.1. Distancia más cercana . . . 28
2.3.2. Clasificación de un triángulo según sus lados . . . 28
2.3.3. Área de un triángulo isósceles . . . 29
2.3.4. Operaciones con fracciones . . . 30
2.3.5. Los números Armstrong . . . 30
2.3.6. Números palíndromos . . . 31
2.3.7. Ecuación cúbica de una variable (adaptado del laboratorio 3 2020-1) . . . 32
2.3.8. Áreas de figuras planas (adaptado del laboratorio 3 2020-1) . . . 33
2.3.9. Ecuación de segundo grado (adaptado del laboratorio 3 2020-1) . . . 34
2.3.10. Ecuación de una paraboloide (adaptado del laboratorio 3 2020-1) . . . 36
2.3.11. Ecuación bicuadrada (adaptado del laboratorio 3 2020-1) . . . 37
2.3.12. Ángulo entre vectores (adaptado del laboratorio 2 2021-1) . . . 38
2.3.13. Propiedades de logaritmos (adaptado del laboratorio 2 2021-1) . . . 39
2.3.14. Trapezoide simétrico (adaptado del laboratorio 2 2021-1) . . . 40
2.3.15. Tipos de cuadriláteros (adaptado del laboratorio 2 2021-1) . . . 42
2.3.16. Área del Pentágono (adaptado del laboratorio 2 2021-1) . . . 43

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

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 lenguaje C y se completaron los casos de prueba de los problemas propuestos.
1.1	15.04.2019	L. Hirsh	Revisión de la versión 1.1.
2.0	06.07.2019	A. Melgar	Se cambio el contenido del documento para incluir tanto la estructura algorítmica selectiva simple como la estructura algorítmica selectiva doble.
2.0	21.08.2019	C. Aguilera	Revisión de la versión 2.0.
2.0	24.08.2019	A. Melgar	Corrección de la revisión de la versión 2.0 realizada por el profesor César Aguilera.
2.1	17.09.2020	S. Vargas	Clasificación de los ejercicios propuestos en nivel básico, intermedio y avanzado. Se agregaron ejercicios.
2.2	09.09.2021	D. Allasi	Se agregaron ejercicios.

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

Siglas

EEGGCC	Estudios Generales Ciencias
IDE	Entorno de Desarrollo Integrado
PUCP	Pontificia Universidad Católica del Perú

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

Capítulo 1

Guía de Laboratorio #2

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 algorítmicas selectivas".

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 algorítmicas selectivas 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 algorítmica selectiva.
- Diseñe algoritmos expresados en diagramas de flujo y pseudocódigos que controlen el flujo usando una estructura algorítmica selectiva.
- Implemente programas que utilicen la estructura algorítmica selectiva 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 C. Como Entorno de Desarrollo Integrado (IDE) para el lenguaje C se utilizará Dev C++[2]. No obstante, es posible utilizar otros IDEs como Netbeans y Eclipse.

1.3. Estructura Algorítmica Selectiva

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

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

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

Las estructuras selectivas, permiten que los programas ejecuten un conjunto de instrucciones si es que cumplen una 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 una determinada condición. Las estructuras algorítmicas selectivas por su lado se pueden clasificar en estructuras selectivas simples y estructuras selectivas dobles.

La estructura selectiva simple permite ejecutar un conjunto de instrucciones si y solo si se cumple una 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 dados sus 3 lados, antes se debe verificar si efectivamente dichos lados forman un triángulo.

La estructura selectiva doble complementa a la estructura selectiva simple. Al igual que la selectiva simple, la estructura selectiva doble permite ejecutar un conjunto de instrucciones si y solo si se cumple una determinada condición. Si la condición no se cumple, se ejecuta otro conjunto de instrucciones. Esto último es lo que diferencia a la selectiva doble de la selectiva simple. En la selectiva doble se cuentan con dos conjuntos de instrucciones que, dependiendo de la condición, se ejecuta uno o se ejecuta el otro. De ahí el nombre de selectiva doble.

La estructura selectiva doble es muy usada en la programación imperativa, gracias a ella se pueden tomar decisiones sobre cómo realizar determinado procesamiento. Muchas situaciones requieren que se realice el cálculo de diferente manera dependiendo de la condición. Por ejemplo, para calcular las raíces de una ecuación cuadrática de una variable, si el discriminante es mayor o igual a cero, se calcula de una forma pues la solución será real, pero si el discriminante es menor que cero, la solución involucrará un número complejo y el procesamiento, en este caso, es diferente. Otro uso muy común es la utilización de la selectiva doble en conjunción con la selectiva simple para ejecutar diferentes acciones en un menú de opciones; como es por ejemplo el caso de una calculadora, que dependiendo de la operación seleccionada (+,−,∗,/), ejecutará diferentes cálculos ($a+b$, $a-b$, $a*b$, a/b).

1.3.1. Representación de la Estructura Algorítmica Selectiva Simple

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

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 ($i \leftarrow 0$), cálculo de expresiones matemáticas ($suma \leftarrow suma + termino$) e inclusive otras estructuras selectivas (`Si i=0 Entonces`).

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

Figura 1.1: Pseudocódigo: Estructura selectiva simple

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

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`.

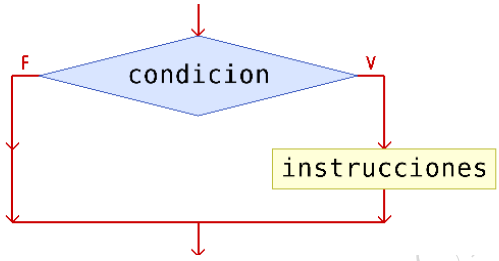

Figura 1.2: Diagrama de Flujo: Selectiva simple

Implementación en lenguaje C

En el lenguaje 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.

Como se sabe, el lenguaje C no implementa nativamente el tipo de dato lógico (`bool` o `boolean`). Entonces ¿cómo hace el lenguaje C para evaluar la condición de la instrucción `if`? el lenguaje 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, un valor diferente a 0 hará que se cumpla la condición. El comportamiento es muy 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*: $suma <= 100$, $i == 10$, $suma < i$ e i. Por otro lado, las siguientes expresiones serán consideradas *falsas*: $suma >= 100$, $i == 20$, $suma > i$ y $suma$.

Programa 1.1: Lenguaje C: Estructura selectiva simple

```
1    ...
2    if (condición){
3        conjunto de instrucciones;
4    }
5    ...
```

1.3.2. Representación de la Estructura Algorítmica Selectiva Doble

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

Representación en pseudocódigo

En la figura 1.3 se puede apreciar la representación de la estructura selectiva doble 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: $i < 100$, $nota <= 10$, $i <= max$ y $i <> 0$. Luego de la `condición` se coloca el identificador `Entonces`.

Como se mencionó anteriormente, a la estructura selectiva doble se le asocian dos conjuntos de instrucciones, un conjunto de instrucciones que se ejecuta si se cumple la condición (conjunto de instrucciones a en la figura

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

1.3) y otro conjunto de instrucciones sino se cumple la condición (conjunto de instrucciones *b* en la figura 1.3). Para separar ambos conjuntos de instrucciones se utiliza el identificador `SiNo`.

En ambos `conjuntos de instrucciones` se pueden colocar asignaciones ($i \leftarrow 0$), cálculo de expresiones matemáticas ($suma \leftarrow suma + termino$) e inclusive otras estructuras selectivas (`Si i=0 Entonces`).

Si *condición* **Entonces**
| conjunto de instrucciones a;
SiNo
| conjunto de instrucciones b;
Fin Si

Figura 1.3: Pseudocódigo: Estructura selectiva doble

Representación en diagrama de flujo

En la figura 1.4 se puede apreciar la representación de la estructura selectiva doble en diagrama de flujo. La estructura inicia con el bloque `condición` y si la condición es verdadera, se ejecuta el `conjunto de instrucciones a`. Si la condición es falsa, se ejecuta el `conjunto de instrucciones b`.

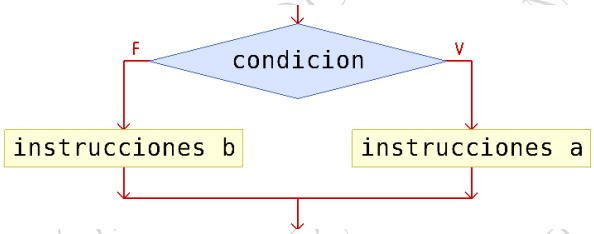

Figura 1.4: Diagrama de Flujo: Selectiva doble

Implementación en lenguaje C

En el lenguaje C la estructura selectiva doble se implementa a través de la instrucción `if`. La representación del `if` se puede apreciar en el programa 1.2. El segundo conjunto de instrucciones se coloca luego del identificador `else`. El conjunto de instrucciones en lenguaje C se delimita por los símbolos `{` y `}`. Esta delimitación es opcional cuando el conjunto de instrucciones está formada por una sola instrucción.

Programa 1.2: Lenguaje C: Estructura selectiva doble

```
1  ...
2  if (condición){
3          conjunto de instrucciones a;
4  }
5  else{
6          conjunto de instrucciones b;
7  }
8  ...
```

1.4. Cálculo de las raíces de ecuaciones cuadráticas con una variable

1.4.1. Verificación del discriminante

Una ecuación cuadrática con una variable 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 produce 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 lenguaje C que calcule las 2 raíces de la solución. En la guía #1 se resolvió este problema y se asumió que el discriminante siempre sería mayor o igual a 0 por lo que la solución daría siempre 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? Antes de procesar se debe verificar si los datos ingresados permiten dicho procesamiento. En este caso en particular, primero se debe hacer la lectura de los datos de la ecuación, luego calcular el discriminante y posteriormente verificar si este es mayor o igual a 0. Esta verificación se realiza con una estructura selectiva simple. Solo si se cumple la condición ($discriminante \geq 0$), se procede al cálculo de las raíces. Si no se cumple la condición, no se hace nada.

En la figura 1.5 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 $discriminante >= 0$).

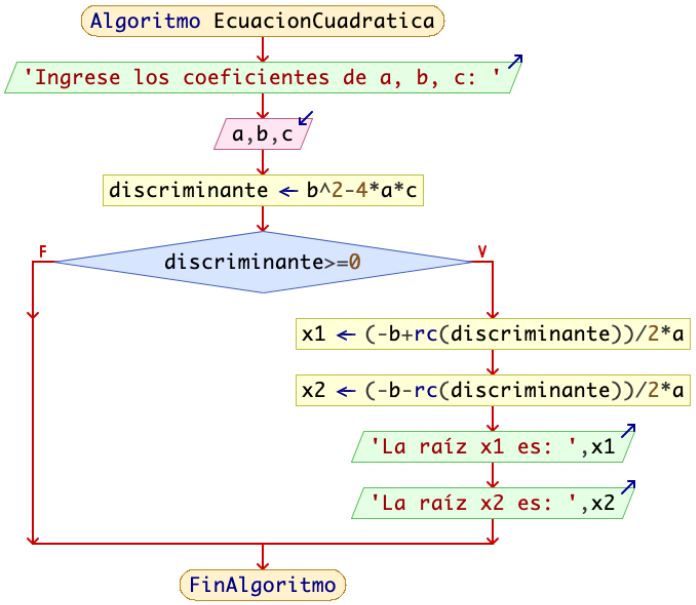

Figura 1.5: Diagrama de flujo: Ecuación cuadrática con estructura algorítmica selectiva simple

El problema con el diagrama de flujo que se visualiza en la figura 1.5 es que si no hay solución real ($discriminante < 0$), el algoritmo no imprime nada. Para solucionar este problema se puede utilizar una estructura algorítmica selectiva doble 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 doble se puede apreciar en la figura 1.6.

En la figura 1.7 se aprecia el pseudocódigo de la última alternativa de solución. La estructura selectiva doble inicia en la línea 5 y finaliza en la línea 12. Por razones estéticas, los conjuntos de instrucciones de la estructura selectiva doble 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 compilan el 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.

En el programa 1.3 se puede apreciar la alternativa de solución en lenguaje C. La estructura selectiva doble inicia en la línea 12 y finaliza en la línea 19. Al igual que en el pseudocódigo, el conjunto de instrucciones de la selectiva doble se encuentra indentado. Un aspecto importante de la sintaxis del lenguaje C es que cuando el conjunto de instrucciones está conformado por una sola instrucción, no es necesario delimitar el bloque con los símbolos `{` y `}`. Esto se puede apreciar en el bloque del `else`, entre las líneas 18 y 19. Otro detalle importante en lenguaje C es que la expresión que representa a la condición en la estructura selectiva doble siempre va

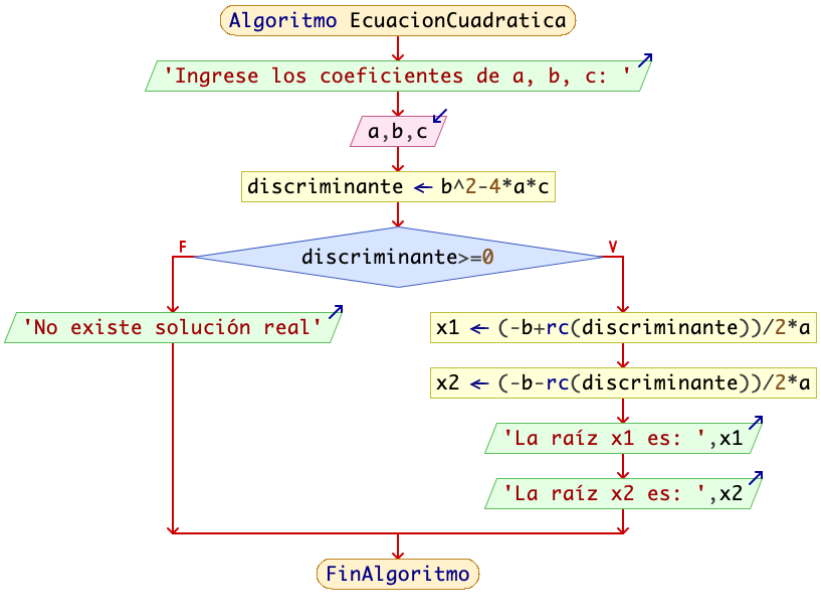

Figura 1.6: Diagrama de flujo: Ecuación cuadrática con estructura algorítmica selectiva doble

```
1  Algoritmo EcuacionCuadratica
2      Escribir 'Ingrese los coeficientes de a, b, c: '
3      Leer a,b,c
4      discriminante <- b^2-4*a*c
5      Si discriminante>=0 Entonces
6          x1 <- (-b+rc(discriminante))/2*a
7          x2 <- (-b-rc(discriminante))/2*a
8          Escribir 'La raíz x1 es: ',x1
9          Escribir 'La raíz x2 es: ',x2
10     SiNo
11         Escribir 'No existe solución real'
12     FinSi
13 FinAlgoritmo
```

Figura 1.7: Pseudocódigo: Ecuación cuadrática

entre paréntesis. Los paréntesis no son necesarios para el caso del diagrama de flujo ni para el pseudocódigo en PSeInt.

Programa 1.3: 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 ecuación: ");
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 raíz x1 es %lf\n", x1);
```

```
16          printf("La raíz x2 es  %lf\n", x2);
17      }
18      else
19          printf("No existe solución real");
20      return 0;
21  }
```

Para poner en práctica

¿Cuáles de las siguientes ecuaciones tiene solución real?

- $x^2 + x + 1 = 0$
- $x^2 + 3x + 6 = 0$
- $2x^2 + 3x + 4 = 0$
- $-2x^2 + 3x + 4 = 0$
- $-x^2 - 2x + 4 = 0$
- $x^2 - 4x - 7 = 0$

1.4.2. Cálculo de las raíces complejas

En la guía #1 se realizó el cálculo de las raíces de una ecuación cuadrática con una variable y se asumió que el discriminante siempre sería mayor o igual a 0 por lo que la solución daría siempre un número real. En la sección anterior se ha planteado una solución en donde se ha verificado que el discriminante sea siempre mayor o igual a 0 para evitar errores de procesamiento. Pero en el mundo de las matemáticas, si el discriminante es menor que 0, existe una solución compleja. ¿Cómo se puede adaptar el programa para que calcule una raíz real si es que el discriminante es mayor o igual a 0 y calcule una raíz compleja si el discriminantes es menor que 0? Este es una situación ideal para una estructura selectiva doble.

Se puede diseñar una estructura selectiva doble para que cuando el discriminantes sea mayor o igual a 0, siga la misma solución ya analizada en la sección anterior. Sino se cumple la condición aparece un conjunto de instrucciones diferente, uno que permita calcular las raíces como números complejos. Este nuevo conjunto de instrucciones es similar al anterior, solo que se debe obtener el valor absoluto del discriminante para poder calcular la raíz cuadrada. En PSeInt el valor absoluto se puede obtener mediante la función `abs`.

Recordar que:

En el contexto de los números complejos $i = \sqrt{-1}$

En la figura 1.8 se puede apreciar el diagrama de flujo que diseña la alternativa de solución propuesta. Note la inclusión de la estructura selectiva doble (bloque con la condición *discriminante* >= 0).

En la figura 1.9 se aprecia el pseudocódigo de la misma alternativa de solución. La estructura doble simple inicia en la línea 5 y finaliza en la línea 16. El bloque de instrucciones que se ejecuta cuando la condición se cumple se encuentra entre las líneas $6 - 9$. El bloque de instrucciones que se ejecuta cuando la condición no se cumple se encuentra entre las líneas $11 - 15$. Por razones estéticas, los conjuntos de instrucciones de la estructura selectiva doble se indenta.

En el programa 1.4 se puede apreciar la alternativa de solución en lenguaje C. La estructura selectiva doble inicia en la línea 12 y finaliza en la línea 26. El bloque de instrucciones que se ejecuta cuando la condición se cumple se encuentra entre las líneas $13 - 17$. El bloque de instrucciones que se ejecuta cuando la condición no se cumple se encuentra entre las líneas $20 - 25$. Al igual que en el pseudocódigo, los conjuntos de instrucciones de la selectiva doble se encuentran indentados.

Hay dos aspectos importantes a comentar en relación al programa 1.4. El primero viene en relación a la declaración de las variables `x1`, `x2`, `parteReal` y `parteImaginaria`. Como usted podrá notar en la línea 13 del programa 1.4 se declaran las variables `x1` y `x2` y en la línea 20 del mismo programa, se declaran las variables `parteReal` y `parteImaginaria`. El lenguaje C permite que las variables se puedan declarar dentro de cada bloque de instrucciones. Lo único que hay que tener en consideración es que dicha variable no se podrá usar

FUNDAMENTOS DE PROGRAMACIÓN
FACULTAD DE CIENCIAS
UNIVERSIDAD CATÓLICA DEL PERÚ

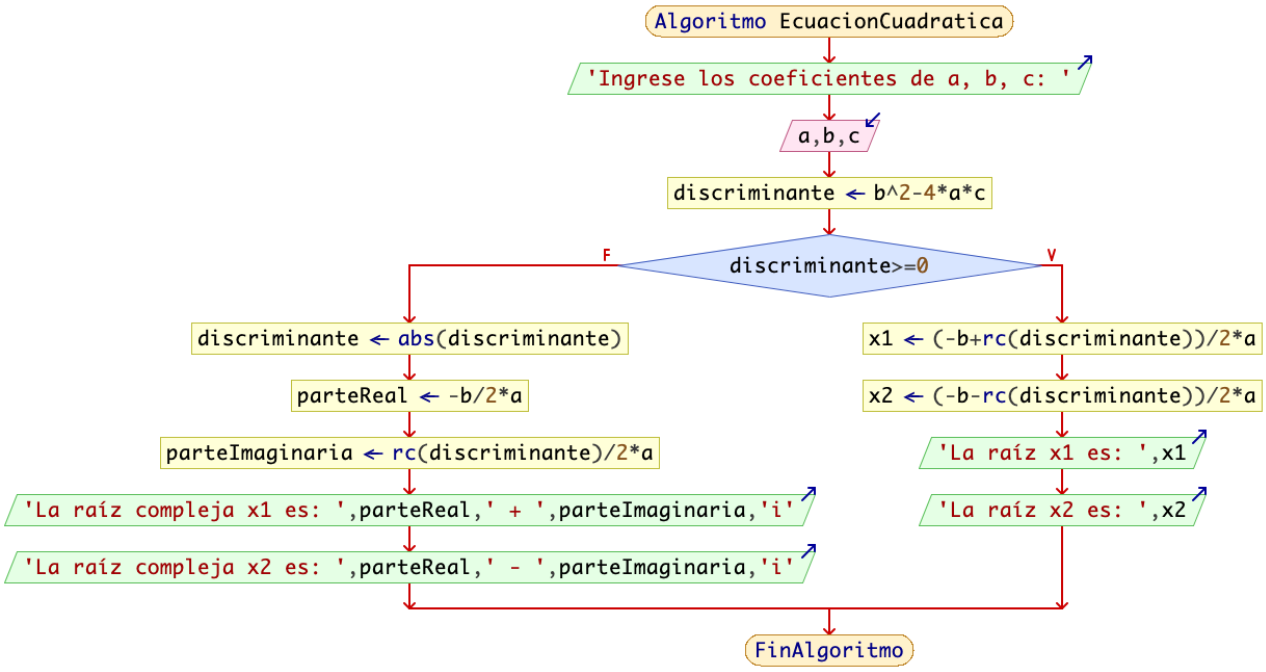

Figura 1.8: Diagrama de flujo: Ecuación cuadrática con raíces reales y complejas

```
1  Algoritmo EcuacionCuadratica
2      Escribir 'Ingrese los coeficientes de a, b, c: '
3      Leer a,b,c
4      discriminante <- b^2-4*a*c
5      Si discriminante>=0 Entonces
6          x1 <- (-b+rc(discriminante))/2*a
7          x2 <- (-b-rc(discriminante))/2*a
8          Escribir 'La raíz x1 es: ',x1
9          Escribir 'La raíz x2 es: ',x2
10     SiNo
11         discriminante <- abs(discriminante)
12         parteReal <- -b/2*a
13         parteImaginaria <- rc(discriminante)/2*a
14         Escribir 'La raíz compleja x1 es: ',parteReal,' + ',parteImaginaria,'i'
15         Escribir 'La raíz compleja x2 es: ',parteReal,' - ',parteImaginaria,'i'
16     FinSi
17 FinAlgoritmo
```

Figura 1.9: Pseudocódigo: Ecuación cuadrática con raíces reales y complejas

fuera del bloque en donde se declara. Esto es muy útil cuando se quiere encapsular el procesamiento a ciertos conjuntos de instrucciones.

El otro aspecto a considerar es el cálculo del valor absoluto en lenguaje C. Como puede notar, en la línea 21 del programa 1.3 se utiliza la función `fabs`. Esta función permite calcular el valor absoluto de un número real. Se encuentra declarada en el archivo de cabecera `math.h`. Para calcular el valor absoluto de un número entero se utiliza la función `abs` que se encuentra declarada en el archivo de cabecera `stdlib.h`.

Programa 1.4: Ecuación cuadrática con raíces reales y complejas

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

```
3
4  int main() {
5      int a, b, c;
6      double discriminante;
7
8      printf("Ingrese coeficientes a, b y c de la ecuación: ");
9      scanf("%d %d %d", &a, &b, &c);
10
11     discriminante = pow(b, 2) - 4 * a*c;
12     if (discriminante >= 0) {
13         double x1, x2;
14         x1 = (-b + sqrt(discriminante)) / 2 * a;
15         x2 = (-b - sqrt(discriminante)) / 2 * a;
16         printf("La raíz real x1 es %lf\n", x1);
17         printf("La raíz real x2 es %lf\n", x2);
18     }
19     else{
20         double parteReal, parteImaginaria;
21         discriminante = fabs(discriminante);
22         parteReal = -b/2*a;
23         parteImaginaria = sqrt(discriminante)/2*a;
24         printf("La raíz compleja x1 es %.2lf + %.2lfi\n", parteReal, parteImaginaria);
25         printf("La raíz compleja x2 es %.2lf - %.2lfi\n", parteReal, parteImaginaria);
26     }
27     return 0;
28 }
```

A continuación, sigue un ejemplo de ejecución de este programa:

```
Ingrese coeficientes a, b y c de la ecuación: 1 -4 13
La raiz compleja x1 es 2.00 + 3.00i
La raiz compleja x2 es 2.00 - 3.00i
```

Para poner en práctica

¿Cuáles son las raíces de las siguientes ecuaciones?

- $x^2 + x + 1 = 0$
- $x^2 + 3x + 6 = 0$
- $2x^2 + 3x + 4 = 0$
- $-2x^2 + 3x + 4 = 0$
- $-x^2 - 2x + 4 = 0$
- $x^2 - 4x - 7 = 0$

1.5. Verificación de datos de entrada para el problema de 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 #1, 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 necesaria la realización de una verificación antes del procesamiento de los datos.

En el programa 1.5 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, en este caso se presentará una mensaje informativo.

Programa 1.5: Verificación del monto de las monedas

```
1  #include <stdio.h>
2  #include <math.h>
3
```

```c
int main() {
    int x, cant50, cant20, cant10, resto;

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

    if (x > 0) {
        cant50 = x / 50;
        x %= 50;
        cant20 = x / 20;
        x %= 20;
        cant10 = x / 10;
        resto = x % 10;
        printf("Se requieren:\n");
        printf("\t%d billete de 50\n", cant50);
        printf("\t%d billete de 20\n", cant20);
        printf("\t%d billete de 10\n", cant10);
        printf("\t%d es el monto faltante\n", resto);
    }
    else
        printf("Ingrese un monto mayor que cero.");
    return 0;
}
```

Para poner en práctica

- Pruebe el programa introduciendo valores negativos o iguales a cero y vea qué sucede.
- 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.

1.6. 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 lenguaje C? En la guía #1 se estudiaron los operadores en lenguaje C, la condición anteriormente descrita se puede expresar como `b>0 && b!=1 && x>0`. En el programa 1.6 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 algorítmica selectiva (y en realidad en todas las estructuras de control de flujo) contenga expresiones complejas que combinen tanto operadores relacionales como operadores lógicos.

Programa 1.6: Verificación de la base y número de un logaritmo

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

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

```

```
7       printf("Ingrese el número y la base: ");
8       scanf("%lf %lf", &numero, &base);
9
10      if (base > 0 && base != 1 && numero > 0) {
11          logaritmo = log(numero) / log(base);
12          printf("El logaritmo es %lf\n", logaritmo);
13      }
14      else
15          printf("Los datos ingresados son inconsistentes\n");
16      return 0;
17  }
```

Para poner en práctica

- Diseñe el algoritmo expresado en diagrama de flujo que corresponde al programa 1.6.
- Diseñe el algoritmo expresado en pseudocódigo que corresponde al programa 1.6.

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"* (ver figura 1.10).

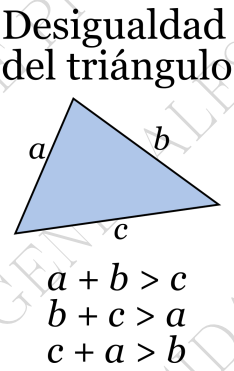

Figura 1.10: Teorema de la desigualdad. Imagen disponible en la URL https://commons.wikimedia.org/wiki/File:Desigualdad_del_tri%C3%A1ngulo.svg.

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 lenguaje C? Esta expresión se representará como una proposición lógica compuesta, usando la conjunción para unir las tres condiciones en la proposición. En lenguaje C el operador de conjunción se representa mediante el símbolo `&&`. Para una mejor lectura de la condición $x + y > z$ se agrupará la suma usando paréntesis de la siguiente manera: `(x+y)>z`. La solución propuesta para este problema se puede apreciar en el programa 1.7.

Programa 1.7: Verificación de los lados de un triángulo

```
1   #include <stdio.h>
2
3   int main() {
4       int a, b, c, lados_forman_triangulo;
5
6       printf("Ingrese los lados de un triángulo: ");
7       scanf("%d %d %d", &a, &b, &c);
8
9       lados_forman_triangulo = (a+b)>c && (a+c)>b && (b+c)>a;
10      if (lados_forman_triangulo)
```

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

```
11          printf("Los lados forman un triángulo.\n");
12      else
13          printf("Los lados no forman un triángulo.\n");
14      return 0;
15  }
```

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 1.7.
- Diseñe el algoritmo expresado en pseudocódigo que corresponde al programa 1.7.

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 días exactos.

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[3].

¿Cómo se puede determinar si un año es bisiesto? 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.

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 1.8 se puede apreciar la implementación de esta expresión lógica en lenguaje C.

Programa 1.8: Verificación de un año bisiesto

```
1   #include <stdio.h>
2
3   int main() {
4       int anho, p, q, r, es_bisiesto;
5
6       printf("Ingrese el año: ");
7       scanf("%d", &anho);
8
9       p = (anho % 4) == 0;
10      q = (anho % 100) == 0;
11      r = (anho % 400) == 0;
12      es_bisiesto = p && (!q || r);
13
14      if (es_bisiesto)
```

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

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

```
15          printf("El año es bisiesto.\n");
16      else
17          printf("El año no es bisiesto.\n");
18      return 0;
19  }
```

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

Capítulo 2

Ejercicios propuestos

2.1. Nivel Básico

En la pregunta 2.1.1 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 C. Para cada uno de estos programas se le solicita que los analice y reflexione sobre el resultado que debería obtener. Luego ejecute el programa usando un Entorno de Desarrollo Integrado (IDE) de lenguaje 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 lenguaje C conforme a los temas revisados en las guías *preliminar* y #1 del curso Fundamentos de Programación.

2.1.1. Análisis de programas

Programa 2.1: 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     return 0;
8 }
```

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 2.2: Análisis de programa: Operador relacional de comparación

```
1 #include <stdio.h>
2
3 int main() {
4     int a=10, b=10, c=10;
5     if (a==b==c)
6         printf("Los 3 números son iguales\n");
7     return 0;
8 }
```

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

Para analizar

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

Programa 2.3: Análisis de programa: Operador relacional de comparación

```
1 #include <stdio.h>
2
3 int main() {
4     int a=1, b=1, c=1;
5     if (a==b==c)
6         printf("Los 3 números son iguales\n");
7     return 0;
8 }
```

Para analizar

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

Programa 2.4: Análisis de programa: Operador aritmético de suma

```
1 #include <stdio.h>
2
3 int main() {
4     int a=5, b=-5;
5     if (a+b)
6         printf("La suma es cero\n");
7     return 0;
8 }
```

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 2.5: Análisis de programa: Operador aritmético de suma

```
1 #include <stdio.h>
2
3 int main() {
4     int a=-3, b=-5;
5     if (a+b)
6         printf("La suma es diferente de cero\n");
7     return 0;
8 }
```

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?

2.1.2. 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.

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

Restricciones

- Para la implementación en lenguaje 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 $x = -3.9$, entonces se debe imprimir 3.9.
- Si $x = 2.5$, entonces se debe imprimir 2.5.
- Si $x = 6.3$, entonces se debe imprimir 6.3.

2.1.3. 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 lenguaje C la declaración de la función `trunc` se encuentra en el archivo de cabecera `math.h`.
- 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 lenguaje 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.

2.1.4. 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 lenguaje C la declaración de la función `trunc` se encuentra en el archivo de cabecera `math.h`.
- 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 de 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 lenguaje 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.

2.1.5. ¿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 $año = 2013$, deberá imprimir 365.
- Si $año = 2016$, deberá imprimir 366.
- Si $año = 2019$, deberá imprimir 365.
- Si $año = 2020$, deberá imprimir 366.
- Si $año = 2024$, deberá imprimir 366.
- Si $año = 2028$, deberá imprimir 366.

2.1.6. ¿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 lenguaje 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)`.

2.1.7. ¿Cuánto cuesta la matrícula?

En determinada universidad la matrícula del semestre tiene un costo de c soles. A los alumnos que se encuentran en facultad se les realiza un descuento de $d\,\%$. Se le solicita que dado el costo c de la matrícula, el descuento d y el número n del ciclo en que se encuentra matriculado 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`.

2.1.8. 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: $1\ Hz = \dfrac{1}{s}$

Casos de prueba

Utilice los siguientes datos para probar su solución.

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

2.2. Nivel Intermedio

2.2.1. 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 lenguaje C

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

Comparación de números reales

Muchas veces el resultado de la comparación de números reales a través de la igualdad no es el deseado. Esto sucede por la forma en que se representa internamente los reales, basta que exista una pequeña diferencia de precisión para que no se de la igualdad. En este caso es recomendable usar el valor absoluto de la diferencia de los números que se desean comparar. Si esta diferencia es cercana a cero, se puede asumir que son iguales.

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)
```

2.2.2. La serie de Gregory

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

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

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 lenguaje 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
- Si $n = 9$, se deberá retornar ≈ 0.058823529

2.2.3. 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 estructura algorítmica selectiva 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 es [5..10], deberá imprimir 355.
- Si el rango es [3..8], deberá imprimir 199.
- Si el rango es [4..4], deberá imprimir 0.
- Si el rango es [7..11], deberá imprimir 415.
- Si el rango es [8..3], deberá imprimir 0.
- Si el rango es [14..19], deberá imprimir 1651.

2.2.4. 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 intersección 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 intersección 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`. En caso contrario 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 $(0, 10)$, $(5, 5)$, $(10, 0)$

2.2.5. 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 circunferencia. 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`. En caso contrario 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`.

2.2.6. 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 S 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 anual 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 $S = 350000$ 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`.

2.2.7. ¿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`.

2.2.8. ¿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`. En caso contrario 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`.

2.3. Nivel Avanzado

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

Variación al problema

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

2.3.2. 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`.

2.3.3. 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 calcular el área del 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.

2.3.4. 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 $\frac{6}{35}$

2.3.5. 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.

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ FACULTAD DE CIENCIAS E INGENIERÍA

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` ($371 = 3^3 + 7^3 + 1^3$).

2.3.6. 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. Si no lo encuentra deberá imprimirse el mensaje `No se encontró el 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).

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
UNIVERSIDAD CATÓLICA DEL PERÚ

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`.

2.3.7. Ecuación cúbica de una variable (adaptado del laboratorio 3 2020-1)

La ecuación cúbica de una variable es una expresión de la siguiente forma:

$$ax^3 + bx^2 + cx + d = 0$$

Donde los coeficientes (a, b, c y d) son números reales y además se debe cumplir necesariamente que a es diferente de cero. Se sabe además que cuando el discriminante es mayor que cero, la ecuación tiene 3 raíces reales distintas. El discriminante de una ecuación cúbica es $18abcd - 4b^3d + b^2c^2 - 4ac^3 - 27a^2d^2$

El programa deberá realizar lo siguiente:

- Leer los coeficientes.
- Verificar si la ecuación es cúbica.
- Determinar si la ecuación posee 3 raíces reales distintas. Cuando la ecuación se verifique como cúbica y el discriminantes sea mayor que cero, debe emitirse el siguiente mensaje `La ecuación tiene 3 raíces reales distintas`. En caso contrario debe emitirse el siguiente mensaje `La ecuación no tiene 3 raíces reales distintas o no es cúbica`.

Casos de prueba para verificar la solución

Use los siguientes casos para verificar si la solución está correcta.

a	b	c	d	Impresión
0	5	3	3	La ecuación no tiene 3 raíces reales distintas o no es cúbica.
5	1	2	16	La ecuación no tiene 3 raíces reales distintas o no es cúbica.
2	-5	-9	18	La ecuación tiene 3 raíces reales distintas.
2	-7	7	-2	La ecuación tiene 3 raíces reales distintas.

2.3.8. Áreas de figuras planas (adaptado del laboratorio 3 2020-1)

Dado los 3 lados de un triángulo escaleno (a, b y c) y el lado de un héxagono regular (l). Elabore un programa en lenguaje C que permite determinar cuál de las dos figuras geométricas posee el área mayor. Asuma que los lados ingresados forman un triángulo.

El programa deberá realizar lo siguiente:

- Leer los lados del triángulo.
- Leer el lado del hexágono.
- Calcular el área del triángulo.
- Calcular el área del hexágono.
- Determinar la mayor área. Deberá imprimir que figura posee la mayor área.

Casos de prueba para verificar la solución

Use los siguientes casos para verificar si la solución está correcta.

a	b	c	l	Impresión
3	4	5	1.5	El triángulo posee mayor área.
2.24	2.83	1	3	El hexágono posee mayor área.
11	11	7.4	3.2	El triángulo posee mayor área.
4	5	9	7.8	El hexágono posee mayor área.

Recordar que:

Según la fórmula de Herón el área del triángulo se calcula de la siguiente manera:

$$area = \sqrt{s(s-a)(s-b)(s-c)}$$

donde:

- $s = \dfrac{a+b+c}{2}$.
- a, b y c son los lados del triángulo.

Recordar que:

El área de un hexágono regular se calcula de la siguiente manera:

$$area = \frac{3l^2}{2tan(30°)}$$

Recordar que:

$360° = 2\pi$ radianes

Sugerencia para lenguaje C

La función `tan` retorna la tangente del ángulo pasado como parámetro. El ángulo debe estar expresado en radianes. Esta función se encuentra definida en el archivo de cabecera `math.h`

Variación al problema

Desarrolle el programa para un **triángulo equilátero y un decágono regular**.
El área de un triángulo equilátero se puede calcular usando la siguiente fórmula:

$$area = \frac{lado^2\sqrt{3}}{4}$$

No debe usar la fórmula de Herón para calcular el área del triángulo equilátero.
El área de un decágono regular se calcula de la siguiente manera:

$$area = \frac{10l^2}{4tan(18°)}$$

Use los siguientes casos de prueba para verificar si la solución está correcta.

a	l	Impresión
8	1.5	El triángulo posee mayor área.
1	3	El decágono posee mayor área.
7.4	1.2	El triángulo posee mayor área.
12	7.8	El decágono posee mayor área.

Variación al problema

Desarrolle el programa para un **triángulo isósceles y un heptágono regular**.
El área de un triángulo isósceles 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.

No debe usar la fórmula de Herón para calcular el área del triángulo isósceles.
El área de un heptágono regular se calcula de la siguiente manera:

$$area = \frac{7l^2}{4tan(25.71°)}$$

Use los siguientes casos de prueba para verificar si la solución está correcta.

a	b	l	Impresión
3	4	0.5	El triángulo posee mayor área.
2.24	2.83	3	El heptágono posee mayor área.
11	7.4	2.6	El triángulo posee mayor área.
4	5	7.8	El heptágono posee mayor área.

2.3.9. Ecuación de segundo grado (adaptado del laboratorio 3 2020-1)

La ecuación general de segundo grado es una expresión con dos variables x e y de la siguiente forma:

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$$

Donde los coeficientes (A, B, C, D, E y F) son números reales y además se debe cumplir necesariamente que al menos uno de los valores de los coeficientes A, B o C es diferente de cero. Se sabe además que cuando $AC - \frac{B^2}{4} < 0$ se dice que la ecuación es de tipo hiperbólica.

El programa deberá realizar lo siguiente::

- Leer los coeficientes.
- Verificar si la ecuación es general de segundo grado.
- Determinar si la ecuación es hiperbólica. Cuando la ecuación sea general de segundo grado y sea del tipo hiperbólica, debe emitirse el siguiente mensaje `La ecuación es de tipo hiperbólica`. En caso contrario debe emitirse el siguiente mensaje `La ecuación no es de segundo grado o no es de tipo hiperbólica`.

Casos de prueba para verificar la solución

Use los siguientes casos para verificar si la solución está correcta.

A	B	C	D	E	F	Impresión
0	0	0	3.3	1.2	4.5	La ecuación no es de segundo grado o no es de tipo hiperbólica.
1	0	1	3.6	2.7	5.7	La ecuación no es de segundo grado o no es de tipo hiperbólica.
1.3	4.5	2.5	3.5	1.6	4.5	La ecuación es de tipo hiperbólica.
-2.7	-5.4	-2.4	2.6	5.6	1.2	La ecuación es de tipo hiperbólica.

Variación al problema

Desarrolle el programa para verificar si la ecuación es de tipo **parabólica**, cuando $AC - \frac{B^2}{4} = 0$.
Use los siguientes casos de prueba para verificar si la solución está correcta. Solo debe leer los coeficientes A, B y C.

A	B	C	Impresión
0	0	0	La ecuación no es de segundo grado o no es de tipo parabólica.
1	0	1	La ecuación no es de segundo grado o no es de tipo parabólica.
1	2.5	1.5625	La ecuación es de tipo parabólica.
4	8	4	La ecuación es de tipo parabólica.

Variación al problema

Desarrolle el programa para verificar si la ecuación **no es de tipo elíptica**. Una ecuación de segundo grado es de tipo elíptica cuando $AC - \frac{B^2}{4} > 0$.
Use los siguientes casos de prueba para verificar si la solución está correcta. Solo debe leer los coeficientes A, B y C.

A	B	C	Impresión
0	0	0	La ecuación no es de segundo grado o es de tipo elíptica.
1.5	2.5	4.5	La ecuación no es de segundo grado o es de tipo elíptica.
2.3	5.4	1.6	La ecuación no es de tipo elíptica.
5.1	7.4	0.4	La ecuación no es de tipo elíptica

Variación al problema

Desarrolle el programa para verificar si la ecuación **no es de tipo circunferencia**, cuando $A = C$. Use los siguientes casos de prueba para verificar si la solución está correcta.

A	B	C	D	E	F	Impresión
0	0	0	3.3	1.2	4.5	La ecuación no es de segundo grado o es una circunferencia
1	0	1	3.6	2.7	5.7	La ecuación no es de segundo grado o es una circunferencia
1.3	4.5	2.5	3.5	1.6	4.5	La ecuación no es una circunferencia.
-2.7	-5.4	-2.4	2.6	5.6	1.2	La ecuación no es una circunferencia.

2.3.10. Ecuación de una paraboloide (adaptado del laboratorio 3 2020-1)

Un paraboloide se puede expresar a través de una ecuación cuadrática definida en un espacio tridimensional, por lo que los puntos se definen en base a tres coordenadas x, y y z. Una de las formas canónicas del paraboloide es:

$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 - z = 0$$

En donde los coeficientes a y b son números reales diferentes de cero.

El programa deberá realizar lo siguiente:

- Leer los coeficientes de la ecuación (a y b).
- Leer un punto en el espacio ($P = (x, y, z)$)
- Verificar si el punto ($P = (x, y, z)$) pertenece al paraboloide e informar al usuario el resultado de la verificación. Si el punto pertenece al paraboloide el programa debe emitir el mensaje `El punto pertenece al paraboloide`, en caso contrario debe emitir el mensaje `El punto no pertenece al paraboloide`.

Casos de prueba para verificar la solución

Use los siguiente casos para verificar si la solución está correcta.

a	b	x	y	z	Impresión
1	1	3	4	3	El punto no pertenece al paraboloide.
1	1	2	1	4	El punto no pertenece al paraboloide.
1	1	1	1	2	El punto pertenece al paraboloide.
2	3	16	9	73	El punto pertenece al paraboloide.

FUNDAMENTOS DEL PROGRAMACIÓN
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

Variación al problema

Desarrolle el programa para un **hiperboloide**. Una de las formas canónicas del hiperboloide es:

$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 - \left(\frac{z}{c}\right)^2 = 1$$

Use los siguientes casos de prueba para verificar si la solución está correcta. Debe leer los coeficientes a, b y c; números reales diferentes de cero.

a	b	c	x	y	z	Impresión
6	5	3	3	4	3	El punto no pertenece al hiperboloide.
5	1	2	19	9	4	El punto no pertenece al hiperboloide.
2	2	4	2	2	4	El punto pertenece al hiperboloide.
2	3	1	0	3	0	El punto pertenece al hiperboloide.

Variación al problema

Desarrolle el programa para un **elipsoide**. Una de las formas canónicas del elipsoide es:

$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 + \left(\frac{z}{c}\right)^2 = 1$$

Use los siguientes casos de prueba para verificar si la solución está correcta. Debe leer los coeficientes a, b y c; números reales diferentes de cero.

a	b	c	x	y	z	Impresión
6	5	3	3	4	3	El punto no pertenece al elipsoide.
5	1	2	19	9	4	El punto no pertenece al elipsoide.
1	1	1	1	0	0	El punto pertenece al elipsoide.
0.5	0.5	0.5	0	0.5	0	El punto pertenece al elipsoide.

2.3.11. Ecuación bicuadrada (adaptado del laboratorio 3 2020-1)

La ecuación bicuadrada es una expresión de la siguiente forma:

$$ax^4 + bx^2 + c = 0$$

Donde los coeficientes (a, b y c) son números reales y además se debe cumplir necesariamente que a es diferente de cero. La ecuación bicuadrada posee 4 soluciones reales dependiendo del discriminante, pero para este problema asuma que siempre existirán las 4 soluciones reales. Use los casos de prueba de esta hoja para verifica su solución.

Las raíces serán:

$$x_1 = +\sqrt{\frac{-b+\sqrt{b^2-4ac}}{2a}} \quad x_2 = +\sqrt{\frac{-b-\sqrt{b^2-4ac}}{2a}}$$
$$x_3 = -\sqrt{\frac{-b+\sqrt{b^2-4ac}}{2a}} \quad x_4 = -\sqrt{\frac{-b-\sqrt{b^2-4ac}}{2a}}$$

El programa deberá realizar lo siguiente:

- Leer los coeficientes.
- Verificar si la ecuación es bicuadrada.

- En caso que la ecuación no sea bicuadrada debe emitir el siguiente mensaje `La ecuación no es bicuadrada`. En caso contrario debe calcular e imprimir cada una de las 4 raíces.
- *No debe repetir cálculos, debe usar variables intermedias siempre que lo necesite.*

.

Casos de prueba para verificar la solución

Use los siguiente casos para verificar si la solución está correcta.

a	**b**	**c**	x_1	x_2	x_3	x_4
1	-13	36	3	2	-3	-2
1	-10	9	3	1	-3	-1
1	-61	900	6	5	-6	-5

2.3.12. Ángulo entre vectores (adaptado del laboratorio 2 2021-1)

Dados dos vectores que parten del origen en un plano cartesiano, se pide que elabore un algoritmo expresado en pseudocódigo que determine el ángulo que se forma entre estos dos vectores.

El algoritmo deberá realizar lo siguiente:

- Leer un punto $P_1(x_1, y_1)$ que representará al vector 1.
- Leer un punto $P_2(x_2, y_2)$ que representará al vector 2.
- Calcular el producto punto entre los dos vectores.
- Calcular el módulo de cada vector.
- Usando el producto punto y el módulo de cada vector, deberá calcular el coseno del ángulo formado.
- Calcular el ángulo en grados sexagesimales y mostrarlo en pantalla. Recuerde que la función `acos` en PSeInt retornar el ángulo en radianes.
- Determinar e imprimir si el ángulo es agudo. Si el ángulo es agudo se deberá imprimir el siguiente mensaje `El ángulo es agudo`. En caso contrario se mostrará el mensaje `El ángulo es recto u obtuso`.

.

Recordar que:

Para transformar angulos de radianes a sexagesimales puede usar la siguiente relación:

$$360° = 2\pi \text{ radianes}$$

El producto punto se calcula de la siguiente manera:

$$\vec{u}(x_1, y_1) \cdot \vec{v}(x_2, y_2) = x_1 \times x_2 + y_1 \times y_2$$

El módulo de un vector se calcula de la siguiente manera:

$$|\vec{u}(x, y)| = \sqrt{x^2 + y^2}$$

El coseno entre los vectores $\vec{u}$ y $\vec{v}$ se calcula de la siguiente manera:

$$\cos(\theta) = \frac{\vec{u} \cdot \vec{v}}{|\vec{u}| \times |\vec{v}|}$$

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
UNIVERSIDAD CATÓLICA DEL PERÚ

A continuación siguen algunos ejemplos de ejecución del algoritmo:

```
*** Ejecución Iniciada. ***
Ingrese primer vector:
> 4
> -1
Ingrese segundo vector vector:
> 2
> 5
En ángulo entre los vectores es: 82.2348339816
El ángulo es agudo
*** Ejecución Finalizada. ***
```

```
*** Ejecución Iniciada. ***
Ingrese primer vector:
> 5
> 3
Ingrese segundo vector vector:
> 1
> 2
En ángulo entre los vectores es: 32.4711922908
El ángulo es agudo
*** Ejecución Finalizada. ***
```

```
*** Ejecución Iniciada. ***
Ingrese primer vector:
> -2
> -7
Ingrese segundo vector vector:
> -1
> 5
En ángulo entre los vectores es: 152.7446716251
El ángulo es recto u obtuso
*** Ejecución Finalizada. ***
```

2.3.13. Propiedades de logaritmos (adaptado del laboratorio 2 2021-1)

Los logaritmos poseen ciertas propiedades, entre ellas se encuentran las siguientes:

- $\log_{base}(a \times b) = \log_{base}(a) + \log_{base}(b)$
- $\log_{base}(\frac{a}{b}) = \log_{base}(a) - \log_{base}(b)$

Se pide que elabore un algoritmo expresado en pseudocódigo que permita verificar si ambas propiedades se cumplen.

El algoritmo deberá realizar lo siguiente:

- Leer la base.
- Leer los números a y b
- Calcular e imprimir en pantalla los valores de $\log_{base}(a \times b)$ y $\log_{base}(\frac{a}{b})$.
- Calcular por separado los valores de $\log_{base}(a)$ y $\log_{base}(b)$.

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

- Usando los valores calculados diseñe las expresiones lógicas que permitan determinar si se cumple cada una de las propiedades especificadas en este enunciado.
- Determinar, usando las expresiones lógicas calculadas, si se cumplen las propiedades. Si las propiedades se pueden verificar mediante el algoritmo, se debe imprimir el siguiente mensaje `Se cumplen las propiedades`. En caso contrario se mostrará el mensaje `No se cumplen las propiedades`.

.

Recordar que:

PSeInt solamente incluye la función `ln` que implementa el logaritmo natural. Si necesita obtener el logaritmo en otra base, debe usar el teorema de cambio de base. En particula puede usar la siguiente relación:

$$\log_{base}(x) = \frac{ln(x)}{ln(base)}$$

A continuación siguen algunos ejemplos de ejecución del algoritmo:

```
*** Ejecución Iniciada. ***
Ingrese la base:
> 2
Ingrese dos números:
> 8
> 4
logaritmo del producto = 5
logaritmo del cociente = 1
Se cumplen las propiedades
*** Ejecución Finalizada. ***

*** Ejecución Iniciada. ***
Ingrese la base:
> 4
Ingrese dos números:
> 6
> 7
logaritmo del producto = 2.6961587114
logaritmo del cociente = -0.1111962107
Se cumplen las propiedades
*** Ejecución Finalizada. ***

*** Ejecución Iniciada. ***
Ingrese la base:
> 3
Ingrese dos números:
> 27
> 81
logaritmo del producto = 7
logaritmo del cociente = -1
Se cumplen las propiedades
*** Ejecución Finalizada. ***
```

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

2.3.14. Trapezoide simétrico (adaptado del laboratorio 2 2021-1)

Dados 4 puntos en el plano cartesiano que representan los vértices de un trapezoide, se pide que elabore un algoritmo expresado en pseudocódigo que determine si el trapezoide es simétrico o no. Un trapezoide es

simétrico si tiene los dos pares de lados consecutivos iguales.

El algoritmo deberá realizar lo siguiente:

- Leer los 4 puntos, $P_1(x_1, y_1)$, $P_2(x_2, y_2)$, $P_3(x_3, y_3)$ y $P_4(x_4, y_4)$, que representan a los vértices consecutivos de un trapezoide.
- Calcular los 4 lados del trapezoide considerando que se han ingresado los vértices de forma consecutiva.
- Determinar e imprimir si el trapezoide es simétrico o no. Si el trapezoide es simétrico se deberá imprimir el siguiente mensaje `Es un trapezoide simétrico`. En caso contrario se mostrará el mensaje `No es un trapezoide simétrico`. Para fines de esta pregunta, asuma que los lados consecutivos son i) el formado por los vértices 1, 2 y 3 y ii) el formado por los vértices 1, 4 y 3.

.

Recuerdar que:

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

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

A continuación siguen algunos ejemplos de ejecución del algoritmo:

```
*** Ejecución Iniciada. ***
Ingrese punto 1:
> -2
> 0
Ingrese punto 2:
> 0
> 3
Ingrese punto 3:
> 2
> 0
Ingrese punto 4:
> 0
> -4
Es un trapezoide simétrico
*** Ejecución Finalizada. ***

*** Ejecución Iniciada. ***
Ingrese punto 1:
> -2
> -1
Ingrese punto 2:
> 2
> 3
Ingrese punto 3:
> 2
> 6
Ingrese punto 4:
> -1
> -7
No es un trapezoide simétrico
*** Ejecución Finalizada. ***
```

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

```
*** Ejecución Iniciada. ***
Ingrese punto 1:
> -2
> -3
Ingrese punto 2:
> 4
> 6
Ingrese punto 3:
> 1
> 5
Ingrese punto 4:
> -4
> -5
No es un trapezoide simétrico
*** Ejecución Finalizada. ***
```

2.3.15. Tipos de cuadriláteros (adaptado del laboratorio 2 2021-1)

Dados 4 puntos en el plano cartesiano que representan los vértices de un cuadrilátero, se pide que elabore un algoritmo expresado en pseudocódigo que determine si el cuadrilátero es un rombo. Un cuadrilátero es clasificado como rombo si tiene los 4 lados iguales pero dos de sus ángulos son menores que los otros dos.

El algoritmo deberá realizar lo siguiente:

- Leer los 4 puntos, $P_1(x_1, y_1)$, $P_2(x_2, y_2)$, $P_3(x_3, y_3)$ y $P_4(x_4, y_4)$, que representan a los vértices consecutivos de un cuadrilátero.
- Calcular los 4 lados del cuadrilátero considerando que se han ingresado los vértices de forma consecutiva.
- Determinar e imprimir si el cuadrilátero es un rombo o no. Si el cuadrilátero se puede clasificar como rombo, deberá imprimir el siguiente mensaje `Es un rombo`. En caso contrario se mostrará el mensaje `No es un rombo`. Para fines de esta pregunta, asuma que los lados consecutivos son i) el formado por los vértices 1, 2 y 3 y ii) el formado por los vértices 1, 4 y 3. Asuma además que el usuario nunca ingresará un cuadrado.

.

Recuerdar que:

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

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

A continuación siguen algunos ejemplos de ejecución del algoritmo:

```
*** Ejecución Iniciada. ***
Ingrese punto 1:
> -3
> 2
Ingrese punto 2:
> 2
> 2
Ingrese punto 3:
> 5
> -2
```

FUNDAMENTOS DE PROGRAMACIÓN
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
ESTUDIOS GENERALES CIENCIAS

```
Ingrese punto 4:
> 0
> -2
Es un rombo
*** Ejecución Finalizada. ***

*** Ejecución Iniciada. ***
Ingrese punto 1:
> 1
> 4
Ingrese punto 2:
> 3
> 7
Ingrese punto 3:
> 5
> 4
Ingrese punto 4:
> 3
> 1
Es un rombo
*** Ejecución Finalizada. ***

*** Ejecución Iniciada. ***
Ingrese punto 1:
> -4
> 2
Ingrese punto 2:
> 1
> 4
Ingrese punto 3:
> -1
> -1
Ingrese punto 4:
> -5
> 1
No es un rombo
*** Ejecución Finalizada. ***
```

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

2.3.16. Área del Pentágono (adaptado del laboratorio 2 2021-1)

Se desea comparar el área de dos figuras geométricas, un pentágono regular y un círculo. El objetivo es deteminar cual de ellas es mayor. Se pide que elabore un algoritmo expresado en pseudocódigo que dado dos vértices consecutivos de un pentágono regular y el radio de un círculo, determine la figura que posee mayor área

El algoritmo deberá realizar lo siguiente:

- Leer los 2 vértices, $P_1(x_1, y_1)$ y $P_2(x_2, y_2)$ que representan a dos vértices consecutivos de un pentágono regular.
- Leer el radio del círculo.
- Calcular el lado del pentágono y calcular su área. El área de un pentágono regular es aproximadamente $1.72 \times lado^2$.
- Calcular el área del círculo.

- Determinar e imprimir que figura posee mayor área. Si es el pentágono regular se deberá imprimir el siguiente mensaje `El área del pentágono es mayor`. En caso contrario se mostrará el mensaje `El área del círculo es mayor o igual`. En ambos casos deberá imprimirse el área calculada.

.

A continuación siguen algunos ejemplos de ejecución del algoritmo:

```
*** Ejecución Iniciada. ***
Ingrese vértice 1:
> 1
> 2
Ingrese vértice 2:
> 4
> 5
Ingrese radio:
> 6
El área del círculo es mayor o igual 113.0973355292
*** Ejecución Finalizada. ***

*** Ejecución Iniciada. ***
Ingrese vértice 1:
> 1
> 2
Ingrese vértice 2:
> 4
> 5
Ingrese radio:
> 3
El área del pentágono es mayor 30.96
*** Ejecución Finalizada. ***

*** Ejecución Iniciada. ***
Ingrese vértice 1:
> 1
> 2
Ingrese vértice 2:
> 3
> 4
Ingrese radio:
> 5
El área del círculo es mayor o igual 78.5398163397
*** Ejecución Finalizada. ***
```

FUNDAMENTOS DE PROGRAMACIÓN
ESTUDIOS GENERALES CIENCIAS
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ