Laboratorio de Programación en C++

Guía de Ejercicios

26 de agosto de 2025

Instrucciones Generales

A continuación se presentan 4 ejercicios de programación en C++ diseñados para reforzar los conceptos fundamentales del lenguaje. Lea detenidamente cada enunciado y siga las buenas prácticas de programación para su resolución.

- **Claridad del Código:** Escriba código limpio, ordenado y fácil de leer. Utilice nombres de variables y funciones que sean descriptivos.
- **Comentarios:** Agregue comentarios en su código para explicar las partes más complejas o la lógica general de su solución.
- **Validación de Entradas: Es un requisito fundamental validar todos los datos ingresados por el usuario en cada ejercicio.** Si el programa espera un número, debe manejar el caso en que el usuario ingrese texto. Si se pide un número dentro de un rango (como una opción de menú o el número de una tarea), debe asegurarse de que la entrada esté dentro de ese rango válido. El programa no debe terminar de forma inesperada por una entrada incorrecta.
- **Compilación:** Asegúrese de que su código compile sin errores y se ejecute correctamente, produciendo las salidas esperadas para los casos de ejemplo.

Ejercicio 1: Solucionador de Ecuaciones Cuadráticas

Enunciado

Escriba un código que resuelva ecuaciones cuadráticas de la forma $ax^2 + bx + c = 0$. El programa debe solicitar al usuario los valores de los coeficientes **a**, **b** y **c**.

Realice la tarea de forma ordenada implementando la **fórmula cuadrática**. Preste especial atención al orden de las operaciones y al uso correcto de los paréntesis:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Su programa debe manejar los tres casos posibles basados en el valor del discriminante ($\Delta = b^2 - 4ac$):

1. **Si $\Delta > 0$:** La ecuación tiene dos raíces reales distintas. Calcule y muestre ambas.
2. **Si $\Delta = 0$:** La ecuación tiene una única raíz real. Calcule y muéstrela.
3. **Si $\Delta < 0$:** La ecuación tiene dos raíces complejas conjugadas. Muestre un mensaje indicándolo (no es necesario calcular las raíces complejas, solo informar que no hay solución en los números reales).

Cree una función que calcule las raíces. Esta función debe recibir los coeficientes y dos **punteros** a variables de tipo 'double' (para la raíz 1 y la raíz 2) donde almacenará los resultados. La función deberá devolver un valor entero que indique el número de raíces reales encontradas (2, 1, o 0).

Ejemplo de Entradas y Salidas

Entrada 1 (Dos raíces reales)

```
Ingrese el coeficiente a: 1
Ingrese el coeficiente b: -5
Ingrese el coeficiente c: 6
```

Listing 1: Consola

Salida Esperada 1:

```
La ecuacion tiene dos raices reales:
Raiz 1 = 3.00
Raiz 2 = 2.00
```

Entrada 2 (Una raíz real)

```
Ingrese el coeficiente a: 1
Ingrese el coeficiente b: -2
Ingrese el coeficiente c: 1
```

Listing 2: Consola

Salida Esperada 2:

```
La ecuacion tiene una unica raiz real:
Raiz = 1.00
```

Entrada 3 (Raíces complejas)

```
Ingrese el coeficiente a: 5
Ingrese el coeficiente b: 2
Ingrese el coeficiente c: 1
```

Listing 3: Consola

Salida Esperada 3:

```
El discriminante es negativo. No existen soluciones reales.
```

Ejercicio 2: Motor Básico de Aventura de Texto

Enunciado

Escriba un código que simule el motor de un juego de aventura de texto simple. El juego consiste en un mapa de 3x3 habitaciones. El jugador comienza en la posición (1, 1).

Pida al usuario que ingrese comandos para moverse por el mapa. Los comandos válidos son "norte", "sur", .^{es}te", .^{o}esteτ sus abreviaciones "n", "s", .^{e}, .^{o}". También debe aceptar un comando para "salir".

Realice la tarea de forma ordenada:

1. Cree un bucle **'while'** principal que mantenga el juego en ejecución.

2. Dentro del bucle, muestre la posición actual del jugador (ej. .^{Es}tás en la habitación [1,1]") y pida el siguiente comando.

3. Utilice una estructura **'switch'** para procesar el comando del usuario. Implemente la lógica de "fall-through"para que los comandos completos y sus abreviaciones (ej. "norteτ "n") ejecuten la misma acción.

4. Defina dos variables, 'posX' y 'posY', para la ubicación del jugador. Cree una función 'moverJugador' que reciba dos **punteros** a estas variables y un carácter que represente la dirección. La función modificará directamente las coordenadas del jugador.

5. Valide los movimientos. Si el jugador intenta moverse fuera de los límites del mapa (0 a 2 para cada coordenada), muestre un mensaje como "No puedes ir en esa dirección, hay una pared.τ no actualice su posición.

6. El juego termina cuando el jugador introduce el comando "salir".

Ejemplo de Entradas y Salidas

```
Est s en la habitacion [1,1].
> norte
Te mueves al norte.
Est s en la habitacion [0,1].
> n
Te mueves al norte.
No puedes ir en esa direccion, hay una pared.
Est s en la habitacion [0,1].
> este
Te mueves al este.
Est s en la habitacion [0,2].
> salir
Gracias por jugar.
```

Listing 4: Interacción en Consola

Ejercicio 3: Analizador de Datos de Sensores

Enunciado

Escriba un código que analice un conjunto de datos predefinido de un sensor de temperatura. La lista de lecturas de temperatura (en grados Celsius) debe estar declarada directamente en el código dentro de un arreglo.

```
double temperaturas[] = {22.5, 24.1, 21.9, 25.3, 26.0, 19.8, 23.7,
    28.1, 18.5, 22.5};
```

Realice la tarea de forma ordenada: cree un menú dentro de un bucle **'while'** que le permita al usuario realizar consultas sobre estos datos hasta que decida salir. Use una estructura **'switch'**. Las funciones que realicen los cálculos deben recibir el arreglo usando **punteros**.

El menú debe ofrecer las siguientes opciones:

1. **Calcular Promedio:** Recorra la lista y calcule la temperatura promedio en Celsius.
2. **Encontrar Máximo y Mínimo:** Encuentre los valores más altos y más bajos en Celsius.
3. **Contar Temperaturas Superiores a un Umbral:** Pida al usuario un valor de umbral (en Celsius) y cuente cuántas lecturas en la lista superan ese valor.
4. **Mostrar datos en Celsius:** Imprima todos los valores de la lista en su escala original.
5. **Mostrar datos en Fahrenheit:** Recorra la lista, convierta cada valor a Fahrenheit usando la fórmula $F = C \times \frac{9}{5} + 32$ y muéstrelos.

Ejemplo de Entradas y Salidas

```
--- Analizador de Temperaturas ---
1. Calcular Promedio
2. Encontrar Maximo y Minimo
3. Contar Superiores a Umbral
4. Mostrar datos en Celsius
5. Mostrar datos en Fahrenheit
6. Salir
Seleccione una opcion: 1
La temperatura promedio es: 23.24 C

Seleccione una opcion: 5
--- Temperaturas en Fahrenheit ---
72.50 F, 75.38 F, 71.42 F, 77.54 F, 78.80 F, 67.64 F, 74.66 F, 82.58 F,
    65.30 F, 72.50 F
---------------------------------

Seleccione una opcion: 6
Saliendo del programa.
```

Listing 5: Interacción en Consola

Ejercicio 4: Gestor de Lista de Tareas

Enunciado

Escriba un código que funcione como una simple aplicación de lista de tareas (To-Do List). El programa debe usar un **vector** de tipo 'string' para almacenar las tareas.

Realice la tarea de forma ordenada: el programa debe mostrar un menú de opciones dentro de un bucle **'while'**, permitiendo al usuario interactuar con su lista de tareas hasta que seleccione la opción de salir.

Utilice una estructura **'switch'** para manejar las opciones del menú:

1. **Agregar Tarea:** Pida al usuario que ingrese una nueva tarea y añádala al final del vector.
2. **Ver Tareas:** Muestre todas las tareas en la lista, numeradas. Si la lista está vacía, debe mostrar un mensaje indicándolo.
3. **Marcar Tarea como Completada:** Muestre la lista numerada y pida al usuario el número de la tarea que desea eliminar. Borre esa tarea del vector. Valide que el número ingresado sea correcto.
4. **Salir:** Termina el programa.

Ejemplo de Entradas y Salidas

```
--- Mi Lista de Tareas ---
1. Agregar Tarea
2. Ver Tareas
3. Marcar Tarea como Completada
4. Salir
Opcion: 1
Ingrese la nueva tarea: Preparar el laboratorio de C++

Opcion: 1
Ingrese la nueva tarea: Estudiar para el examen

Opcion: 2
--- TAREAS PENDIENTES ---
1. Preparar el laboratorio de C++
2. Estudiar para el examen
--------------------------

Opcion: 3
Numero de la tarea a completar: 1
Tarea "Preparar el laboratorio de C++" completada.

Opcion: 2
--- TAREAS PENDIENTES ---
1. Estudiar para el examen
--------------------------

Opcion: 4
```

```
Hasta luego!
```

Listing 6: Interacción en Consola