Pontificia Universidad Javeriana

Taller #1

Tomas Alejandro Silva Correal

Estructuras de datos

Jhon Jairo Corredor Franco

08 de Marzo de 2025

Índice:

Resumen

En el siguiente documento se trabajará el Taller 1 para la asignatura de estructuras de datos, basado en un plan de pruebas para una serie de programas con nombres: ***exercise1.cpp***, ***exercise2.cxx***, ***rectangle.cxx*** y ***rectangle.h***. Este documento tendrá muestras de la ejecución y compilación de los programas correspondientes según es explicado en el enunciado, así como la muestra y análisis de resultados de dichos programas.

Introducción

Los programas que se ejecutarán en este taller se basan en dos elementos, la creación de nodos para una lista, y un algoritmo para el análisis de rectángulos, incluyendo el área, el perímetro, y la distancia del mismo al punto de origen de las coordenadas dado algunos datos de entrada como el largo, el ancho, y las coordenadas X y Y de la figura. Algo que se puede evidenciar antes de probar los programas es que existe una clase modular, siendo esta ***rectangle.cxx***, esta cuenta con su programa, así como un ***rectangle.h*** que se utiliza para el llamado a funciones.

Ejercicio 1

A continuación, se mostrará la ejecución del programa ***exercise1.cpp***, junto a sus resultados y un análisis de los mismos.

Compilación y resultados:

```
~$ g++ -std=c++11 -o exercise1 exercise1.cpp
~$ ./exercise1
Creating Node, 1 are in existence right now
Creating Node, 2 are in existence right now
Creating Node, 3 are in existence right now
Creating Node, 4 are in existence right now
The fully created list is:
4
3
2
1

Now removing elements:
Creating Node, 5 are in existence right now
Destroying Node, 4 are in existence right now
4
3
2
1
```

Imagen 1, compilación y resultados del programa *exercise1.cpp*

Análisis de resultados:

De lo que es posible ver en el programa, se generan un total de cuatro nodos que se muestran, antes de crear un quinto nodo para eliminarlo de inmediato, este quinto nodo no debería existir, dado que es un error del programa, en el que crea un nodo si la cantidad de nodos no es igual a 0, lo cual no debería de ocurrir.

Ejercicio 2

A continuación, se mostrará la ejecución del programa ***exercise2.cpp*** y su módulo o librería (***rectangle.cxx*** y ***rectangle.h)***, junto a sus resultados, la utilización del plan de pruebas, y un análisis de los resultados.

Compilación:

A continuación, se mostrará una imagen con el proceso de compilación del programa del ejercicio:

```
~$ g++ -std=c++11 -g -o exercise2 exercise2.cxx
~$ gbd exercise2
bash: gbd: command not found
~$ gdb exercise2
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.
(gdb) run
Starting program: /home/user/exercise2
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Ingrese coordenada X de la posicion del rectangulo: 2
Ingrese coordenada Y de la posicion del rectangulo: 6
Ingrese ancho del rectangulo: 5
Ingrese alto del rectangulo: 8

Perimetro del rectangulo: 18
Area del rectangulo: 13
Distancia del rectangulo al origen de coordenadas: 6.32456
[Inferior 1 (process 906) exited normally]
(gdb) backtrace
No stack.
(gdb) Quit
Undefined command: "Quit".  Try "help".
(gdb) quit
~$
```

Imagen 2, compilación y prueba de *exercise2.cpp*

Plan de pruebas:

A continuación, se mostrará el plan de pruebas con los resultados obtenidos:

Plan de pruebas: función Perímetro del rectángulo			
Descripción de caso	Valores de entrada	Resultado esperado	Resultado obtenido
1: Alto como el doble de Ancho	Ancho = 2, Alto = 4	12	8
2: Alto igual a Ancho	Ancho = 3, Alto = 3	12	9
3: un numero en cero	Ancho = 5, Alto = 0	10 (error)	10

Tabla 1, Plan de pruebas para el perímetro de un rectángulo

Plan de pruebas: función Área del rectángulo			
Descripción de caso	Valores de entrada	Resultado esperado	Resultado obtenido
1: Alto como el doble de Ancho	Ancho = 2, Alto = 4	8	6
2: Alto igual a Ancho	Ancho = 3, Alto = 3	9	6
3: un numero en cero	Ancho = 5, Alto = 0	0	5

Tabla 2, Plan de pruebas para el área de un rectángulo

Plan de pruebas: función Distancia del rectángulo al origen			
Descripción de caso	Valores de entrada	Resultado esperado	Resultado obtenido
1: números positivos	x = 15, y = 32	35.34	35.34
2: un número 0	x = 0, y = 32	32	32
3: números iguales	x = 15, y = 15	21.21	21.2132

Tabla 13, Plan de pruebas para la distancia del rectángulo al origen

Análisis de resultados:

Como se puede evidenciar por medio del plan de pruebas, es posible notar que el programa es mayormente inconsistente o contiene fallas en los cálculos de área y perímetro, sin embargo, la distancia al origen funciona correctamente. Para el caso del perímetro, parece ser que el programa cuenta mal los lados, puesto que el programa cuenta dos veces el ancho más el alto en lugar de multiplicar la suma por dos (este es un problema del orden de operaciones, donde no multiplica la suma por 2, sino que en su lugar solo duplica el ancho). Mientras que para el área, parece sumar el ancho y alto en lugar de multiplicarlos. Como se explicó anteriormente, el algoritmo para la búsqueda del rectángulo al origen parece funcionar correctamente, siendo los mismos valores el esperado y obtenido.

El problema de los algoritmos recae en la librería ***rectangle.cxx***, la cual contiene la implementación de los algoritmos, donde usualmente los problemas son de orden de operaciones (más que nada en el caso del perímetro), o de lógica (como en el caso del área).

Directorio

Con tal de asegurar la existencia adecuada de los archivos, se presentará una imagen del directorio del taller:

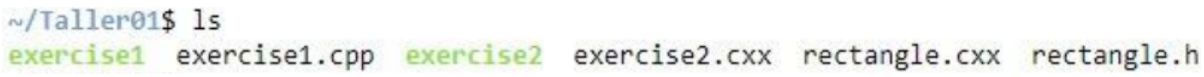

Imagen 3, Directorio de proyecto

Conclusiones

Para concluir, y como era el objetivo del taller, se comprendió y trabajó el concepto de un plan de pruebas como forma de confirmación de funciones para un programa, usando un programa que generaba resultados erróneos para algunas funciones, y correctas para otras. Este taller aporta a la comprensión de la importancia de un plan de pruebas, puesto que el mismo permite una comparación directa de los resultados que se esperan y los que resultan del proceso.

Fuera de lo ya concluido, también es necesario hacer la retroalimentación de los programas usados para el taller. Ninguno de los dos programas funcionaba correctamente, el primero dado a que generaba un nodo extra solo para eliminarlo de inmediato, lo cual puede ser un error de lógica a la hora de escribir el código, y el segundo al contener varios errores en la lógica del funcionamiento (Area y perimetro del rectangulo siendo los dos casos con fallas). Los errores en estos códigos no son complicados de corregir, pero pueden ser difíciles de evidenciar sin un plan de pruebas adecuado para su identificación y corrección.