Informe del Taller 00: Compilación, Pruebas y Depuración

Integrantes: Carlos Antonio D'Silvestri Ruiz, Giseth Villalobos Rocha, Diego Alejandro Albarracín.

5.1 Compilación

1. ¿Qué contiene el archivo?

- **Excercise1.cxx:** es la interfaz para poder digitar la información pertinente con respecto al rectángulo, es decir, la coordenada en x, la coordenada en y, ancho y alto del rectángulo.
    - Después de digitar todos los datos se invocan las funciones de areaRect(rect1), perimeterRect(rect1) y distOriginRect(rect1).
- **Rectangle.cxx:** esta clase contiene todo lo necesario para poder realizar el cálculo del área, perímetro y el cálculo de la distancia del rectángulo al origen de coordenadas 0,0
- **Rectangle.h:** este código es una librería que se caracteriza por tener la definición de la estructura y los métodos que son necesarios para representar un rectángulo.

2. ¿Cuál es el comando completo a escribir en la consola que permite compilar y encadenar estos archivos sin errores?

3. ¿Cuál es el comando que permite ejecutar en la consola el programa anteriormente compilado?

5.2 Plan de Pruebas

- **Perímetro del rectángulo**

Se evalúan los valores ingresados que en el ancho y alto del rectángulo. Se asume que no puede existir un ancho o un alto que sea negativo, por ende, el resultado esperado de la operación para calcular perímetro cuando se ingresa un valor negativo debe ser error.

Descripción del caso	Valores entrada	Valor esperado	Resultado obtenido
1. Valores positivos	Rect.width = 12 Rect.height = 10	Perim = 44	Perímetro: 34.
2. Un valor = 0	Rect.width = 0 Rect.height = 10	Perim = 0	Perímetro: 10.
3. Dos valores iguales	Rect.width = 12 Rect.height = 12	Perim = 48	Perímetro: 36.
4. Un valor negativo	Rect.width = 12 Rect.height = -2	Perim = Error	Perírmetro: 22.

Los resultados son erróneos dado que en el archivo rectangle.cxx, la línea 7 se está multiplicando la variable rect.width con 2.0 y este resultado se está sumando después con el valor de rect.height.

La modificación requerida para que funcione de manera correcta es reemplazar la línea 7 por la siguiente línea: perim = (rect.width + rect.height) * 2.0;

- **Área del rectángulo**

Se evalúan los valores ingresados en el ancho y alto del rectángulo. Se asume que no puede existir un ancho o un alto que sea negativo, por ende, el resultado esperado de la operación para calcular el área del rectángulo cuando se ingresa un valor negativo debe ser error.

Descripción del caso	Valores entrada	Valor esperado	Resultado obtenido
1. Valores positivos	Rect.width = 12 Rect.height = 10	Area = 120	Area: 22.
2. Un valor = 0	Rect.width = 0 Rect.height = 10	Area = 0	Area: 10.
3. Dos valores iguales	Rect.width = 12 Rect.height = 12	Area = 144	Area: 24.
4. Un valor negativo	Rect.width = 12 Rect.height = -2	Area = Error	Area: 10.

Los resultados son erróneos dado que, en el archivo rectangle.cxx, la línea 14 está realizando una suma entre la variable rect.width y la variable rect.height.

La modificación requerida para que funcione correctamente es reemplazar la línea 14 por la siguiente linea: area = rect.width * rect.height;

- **Distancia del rectángulo al punto origen**

Se evalúan los valores ingresados en la posición en X y la posición en Y del rectángulo ingresadas. Para este caso sí hay posibilidad de que un valor sea negativo, sin embargo, el resultado de cada intento siempre debe ser positivo, ya que la distancia entre el punto origen y la ubicación del rectángulo es un valor positivo.

Descripción del caso	Valores entrada	Valor esperado	Resultado obtenido
1. Valores positivos	Rect.posX = 3 Rect.posY = 4	Dist = 5	5
2. Un valor = 0	Rect.posX = 4 Rect.posY = 0	Dist = 4	4
3. Dos valores iguales	Rect.posX = 6 Rect.posY = 6	Dist = 8.48528	8.48528
4. Un valor negativo	Rect.posX = -4 Rect.posY = 8	Dist = 8.94427	8.94427
5. Dos valores negativos	Rect.posX = -2 Rect.PosY = -6	Dist = 6.32456	6.32456

Los resultados obtenidos al ejecutar el programa son los esperados en las pruebas definidas. El código está correctamente definido.

5.3 Depuración

1. ¿Qué contiene el archivo?

a. Excercise2.cxx: este código lo que contiene es la manera de crear y manipular una lista enlazada, es decir, crea la lista y tiene todos los métodos para poder eliminar e imprime el contenido de dicha lista, entonces, en el main está metiendo los números 1,2,3 y 4 en la lista, los imprime para ver la información, remueve el 4, 1, 2 y 3 en el orden mencionado y elimina la lista.

2. ¿Cuál es el comando completo a escribir en la consola que permite compilar y encadenar el archivo sin errores?
    a. g++ -std=c++11 -c *.cpp (comando para compilar y ejecutar excercise2.cpp)
3. ¿Cuál es el comando que permite ejecutar en la consola el programa anteriormente compilado?
    a. g++ -std=c++11 -o ejecutable *.o
4. ¿El programa corre sin errores? Si es así, hasta aquí llega el ejercicio. En caso contrario, es necesario depurar el código en busca de la fuente del error. Para eso, responda en su reporte las siguientes preguntas:
    a. El programa no posee errores de ejecución, el problema si posee errores en el tema de la lógica. El error radica en que a la hora de realizar remove en la lista se realiza un bucle infinito, por lo que no eliminaba el dato a suprimir, la corrección es la eliminación de marker = 0 que lo ocacionaba.

    Código sin corrección:

```
int remove(const T &item_to_remove) {

  Node<T> *marker = head_;

  Node<T> *temp = 0; // temp points to one behind as we iterate


  while (marker != 0) {

   if (marker->value() == item_to_remove) {

    if (temp == 0) { // marker is the first element in the list

     if (marker->next() == 0) {

      head_ = 0;

      delete marker; // marker is the only element in the list

      marker = 0;

     } else {

      head_ = new Node<T>(marker->value(), marker->next());

      delete marker;

     }

     return 0;

    } else {

     temp->next(marker->next());

     delete temp;
```

```
        temp = 0;
        return 0;
      }
    }
    marker = 0; // reset the marker
    temp = marker;
    marker = marker->next();
  }

  return -1; // failure
```

consola:

```
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
```

Segmentation fault (core dumped)

Corrección:

```
int remove(const T &item_to_remove) {
   Node<T> *marker = head_;
   Node<T> *temp = 0;

   while (marker != 0) {
     if (marker->value() == item_to_remove) {
       if (temp == 0) {
         head_ = marker->next();
         delete marker;
         return 0;
       } else {
         temp->next(marker->next());
         delete marker;
         return 0;
       }
     }
     temp = marker;
     marker = marker->next();
   }

   return -1;
 }
```

Commando:

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:

Destroying Node, 3 are in existence right now

3

2

1

Destroying Node, 2 are in existence right now

3

2

Destroying Node, 1 are in existence right now

3

Destroying Node, 0 are in existence right now

5. ¿Cuál es el comando completo a escribir en la consola que permite compilar y encadenar el archivo para su uso dentro del depurador?

Gdb nombre del ejecutable