



## Digital Design Verification

**Assignment # 03**

**Constructor in C++**

**Catching Bugs in C++**

**Submitted by:**

Name:	<b>Khalil Rehman</b>
Instructor:	<b>Hira Sohail</b>

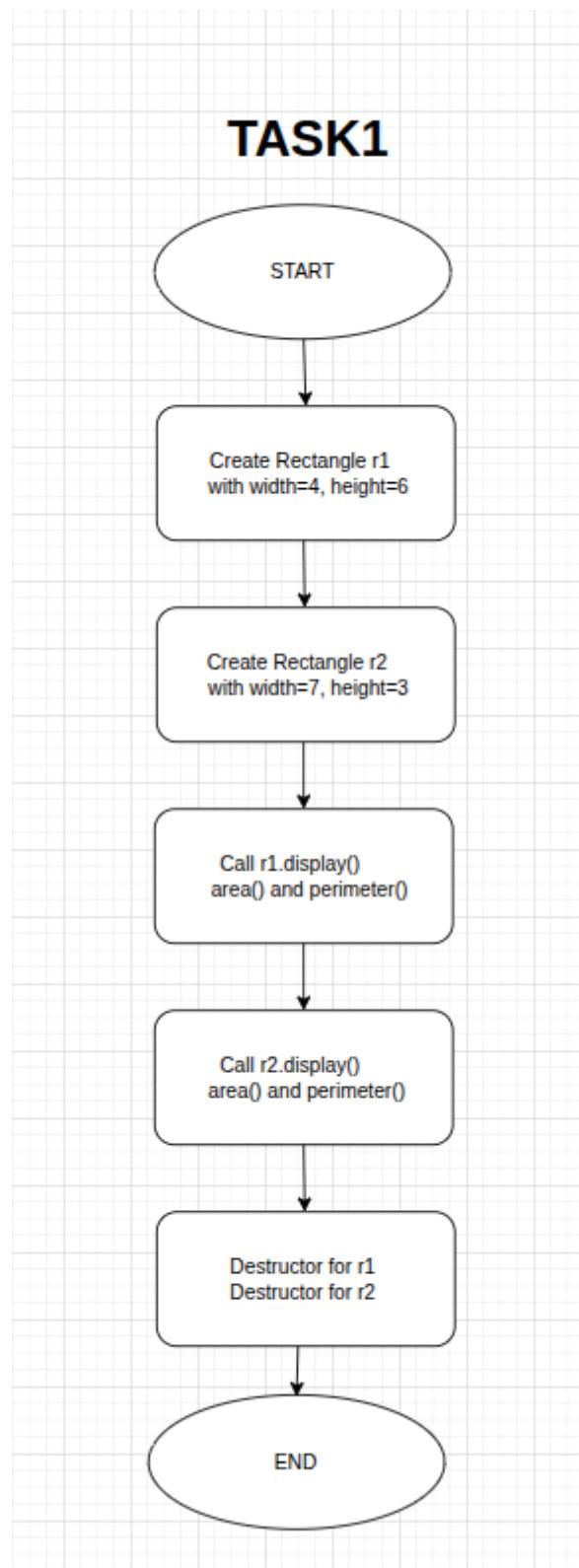
**Date:**  
July 26, 2025

**NUST Chip Design Centre (NCDC), Islamabad, Pakistan**



## TASK # 01:

Flow Chart:





## TERMINAL OUTPUT:

The terminal window shows the execution of a C++ program named task1. It includes the compilation command `g++ task1.cpp -o task1`, the execution command `./task1`, and the program's output. The output displays the creation of two rectangles (Width: 4, Height 6 and Width: 7, Height 3), their areas (Area 24 and Area 21), and their parameters (Parameter: 20). It also shows the destruction of these objects.

```
PROBLEMS OUTPUT TERMINAL PORTS ... [] X
▼ TERMINAL
● Khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/C++/Assignment#03$ g++ task1.cpp -o task1
● Khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/C++/Assignment#03$ ./task1
Constructor called. Rectangle created
Constructor called. Rectangle created
Width: 4, Height 6
Area 24, Parameter: 20
-----
Width: 7, Height 3
Area 21, Parameter: 20
-----
Destructor called. Deleting Rectangle
Destructor called. Deleting Rectangle
○ Khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/C++/Assignment#03$
```

## TASK#02

1.

- `myPoint` declared as `const`, which means it cannot be modified. But `doubleVal()` modifies the member variables `x` and `y`, which violates the `const`.

### Code

The code editor shows a file named task2.cpp. It contains a class `Point` with private member variables `x` and `y`. The constructor takes two integers `u` and `v` and initializes `x` and `y` respectively. It has two `const` member functions `getX()` and `getY()` that return the current values of `x` and `y`. It also has a non-`const` member function `doubleVal()` that multiplies both `x` and `y` by 2. The `main` function creates a `const Point myPoint(5, 3);`, calls `myPoint.doubleVal();`, and then prints the values of `myPoint.getX()` and `myPoint.getY()`.

```
home > khalilrehman > Documents > C++ > Assignment#03 > task2.cpp > ...
1 #include <iostream>
2 using namespace std;
3
4
5 //#
6 class Point {
7 private:
8     int x, y;
9
10 public:
11     Point(int u, int v) : x(u), y(v) {}
12     int getX() const { return x; }
13     int getY() const { return y; }
14     void doubleVal() {
15         x *= 2;
16         y *= 2;
17     }
18 };
19
20 int main() {
21
22     //const Point myPoint(5, 3);
23     //myPoint.doubleVal();
24
25     Point myPoint(5, 3);
26     myPoint.doubleVal();
27     cout << myPoint.getX() << " " << myPoint.getY() << "\n";
28     return 0;
29 }
```



2.

- The method `setX()` is marked `const`, which means it cannot change any class member variables.

### Code

```
32
33 #include <iostream>
34 using namespace std;
35
36 class Point {
37 private:
38     int x, y;
39
40 public:
41     Point(int u, int v) : x(u), y(v) {}
42     int getX() const { return x; }
43     int getY() const { return y; }
44 //    void setX(int newX) const { x = newX; }
45     void setX(int newX) { x = newX; }
46 };
47
48 int main() {
49     Point p(5, 3);
50     p.setX(9001);
51     cout << p.getX() << ' ' << p.getY();
52     return 0;
53 }
```

3.

- `x` and `y` are declared `private`: in the `Point` class. Private members cannot be accessed directly outside the class to access them directly in `main()`, which causes a compilation error.



```
56 //cout << p.x << " " << p.y << "\n";
57
58
59
60 #include <iostream>
61 using namespace std;
62
63 class Point {
64 private:
65     int x, y;
66
67 public:
68     Point(int u, int v) : x(u), y(v) {}
69     int getX() { return x; }
70     int getY() { return y; }
71 };
72
73 int main() {
74     Point p(5, 3);
75     cout << p.getX() << " " << p.getY() << "\n"; //Corrected line
76     return 0;
77 }
78
```

#### 4

- Line 10: Redundant declaration of setX() with no body.
- Line 13: Valid definition of setX() is placed after the class is already closed.

```
79
80 //4
81 #include <iostream>
82 using namespace std;
83
84 class Point {
85 private:
86     int x, y;
87
88 public:
89     Point(int u, int v) : x(u), y(v) {}
90     int getX() { return x; }
91
92     void setX(int newX) { //Move inside class and define once
93         x = newX;
94     }
95 };
96
97 int main() {
98     Point p(5, 3);
99     p.setX(0); // Now allowed
100    cout << p.getX() << " " << "\n";
101    return 0;
102 }
```



## 5

- `new int[size]` dynamically allocates an array.
- must use `delete[]` to correctly free the memory which causes undefined behavior and a memory leak.

```
104 //5
105
106
107 int size;
108 cin >> size;
109
110 int* nums = new int[size];
111 for (int i = 0; i < size; ++i) {
112     cin >> nums[i];
113 }
114
115 // ... // Calculations with nums
116
117 delete[] nums; //Correctly deallocates dynamic array
118
```

## 6

- `new Point(...)` allocates memory on the heap. But `delete p;` is missing which causes a memory leak especially in larger programs.

```
119 //6
120
121 #include <iostream>
122 using namespace std;
123
124 class Point {
125 private:
126     int x, y;
127
128 public:
129     Point(int u, int v) : x(u), y(v) {}
130     int getX() { return x; }
131     int getY() { return y; }
132 };
133
134 int main() {
135     Point* p = new Point(5, 3);
136     cout << p->getX() << ' ' << p->getY() << endl;
137
138     delete p; //Prevent memory leak
139     return 0;
140 }
```