Digital Design Verification

Assignment # 03

Constructor in C++

Catching Bugs in C++

Release: 1.0

Date: 05-March-2025

NUST Chip Design Centre (NCDC), Islamabad, Pakistan

Revision History

Revision Number	Revision Date	Revision By	Nature of Revision	Approved By
1.0	23/07/2025	Hira Sohail	Manual	Sharjeel Khilji

Tools & Language

- GNU G++ compiler
- C++

Task # 01:

Design a C++ class named Rectangle to model the behavior of a rectangle.

The class should:

- Dynamically allocate memory for width and height using the *new* operator.
- Include a parameterized constructor to initialize these dimensions at the time of object creation.
- Include a destructor that deallocates the dynamically allocated memory and displays a message indicating object destruction.
- Implement a member function area() that returns the area of the rectangle.
- Implement a member function perimeter() that calculates and returns the perimeter of the rectangle.
- Add a method display() to show the rectangle's dimensions, area, and perimeter neatly.

In the main() function:

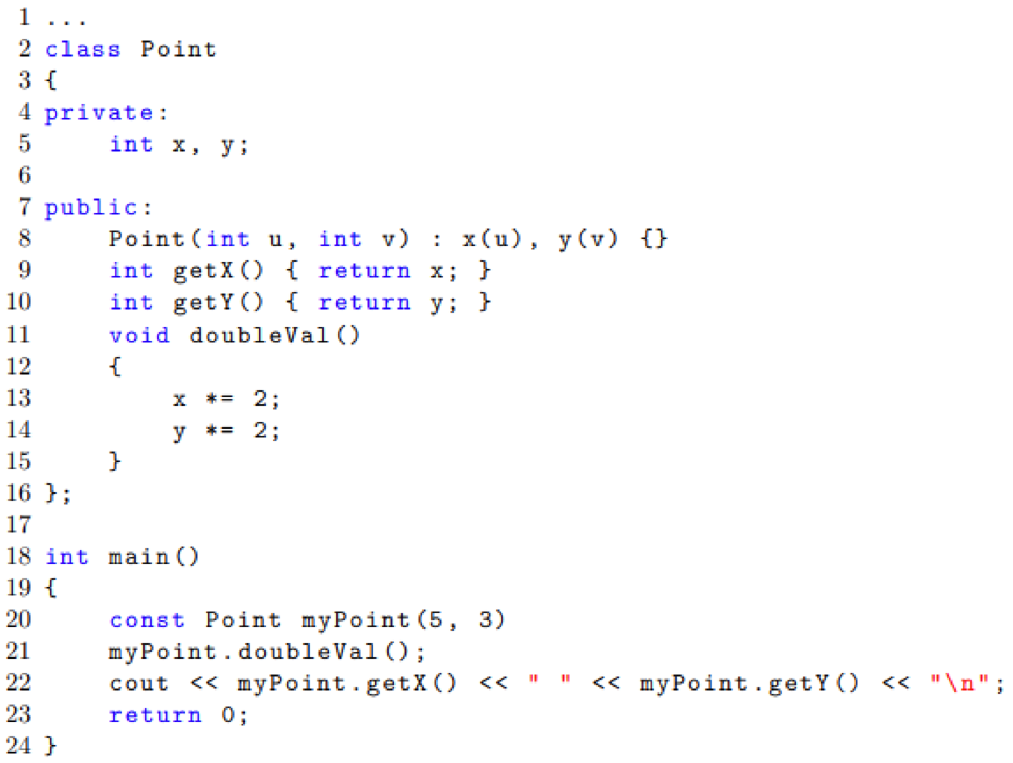
- Create two Rectangle objects with different dimensions.
- Call the display() method for each object.
- Observe the constructor and destructor messages to understand the object lifecycle and memory management.

Task #02:

Cath that Bug!

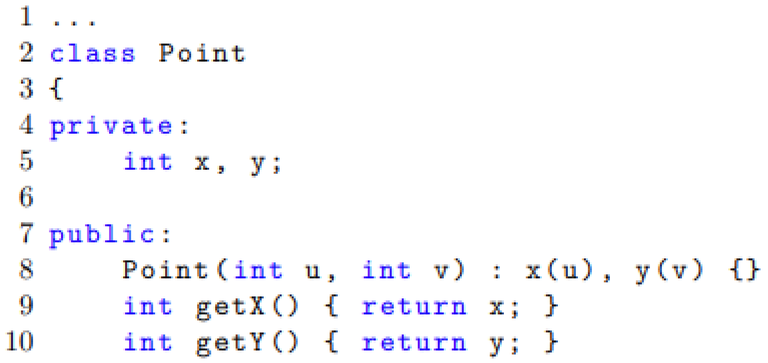
In this section, the following snippets will have bugs. Identify them and indicate how to correct them. Do these without the use of a computer!

1.

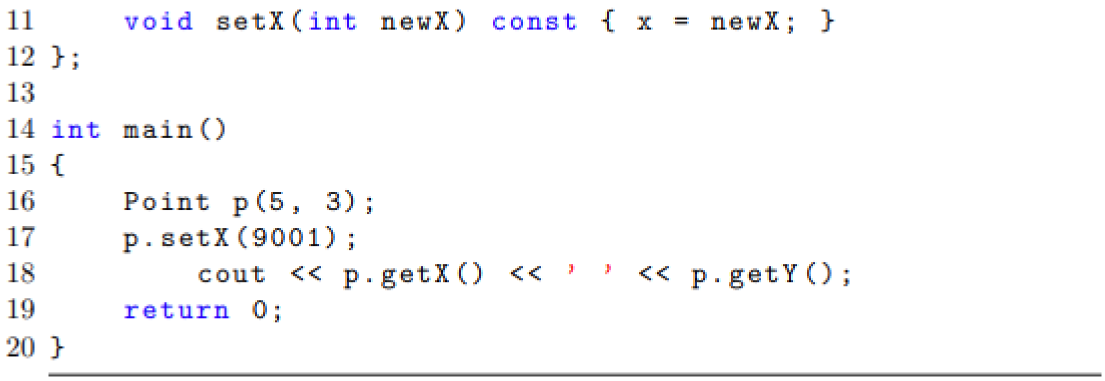


```
...
class Point
{
private:
    int x, y;

public:
    Point(int u, int v) : x(u), y(v) {}
    int getX() { return x; }
    int getY() { return y; }
    void doubleVal()
    {
        x *= 2;
        y *= 2;
    }
};

int main()
{
    const Point myPoint(5, 3)
    myPoint.doubleVal();
    cout << myPoint.getX() << " " << myPoint.getY() << "\n";
    return 0;
}
```

2.

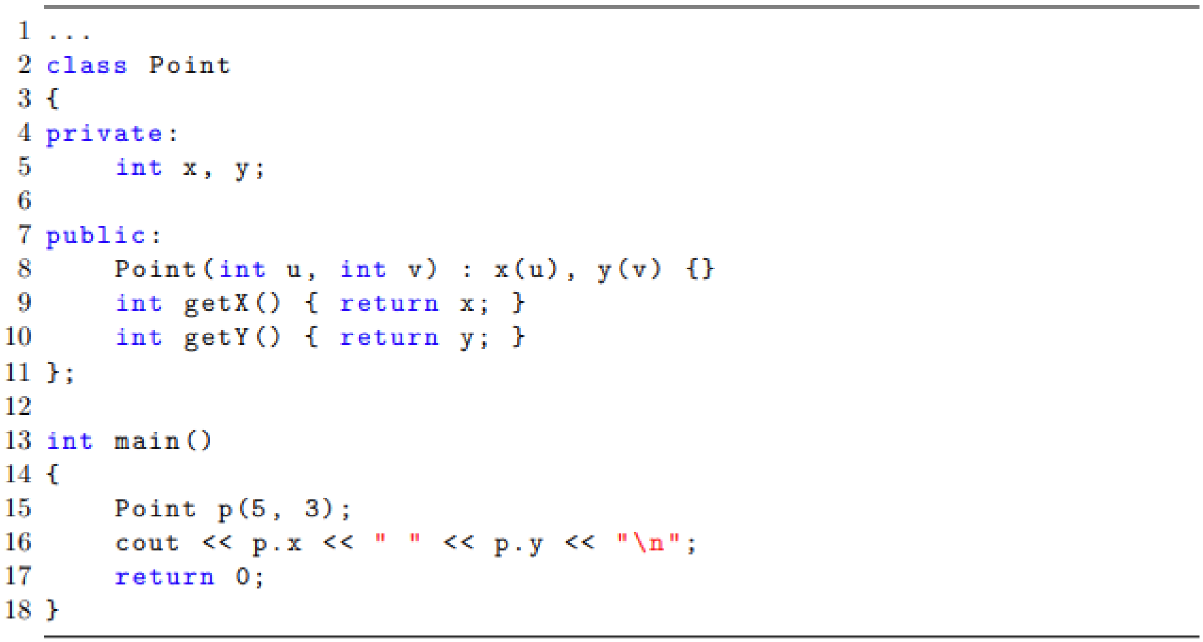


```
...
class Point
{
private:
    int x, y;

public:
    Point(int u, int v) : x(u), y(v) {}
    int getX() { return x; }
    int getY() { return y; }
```



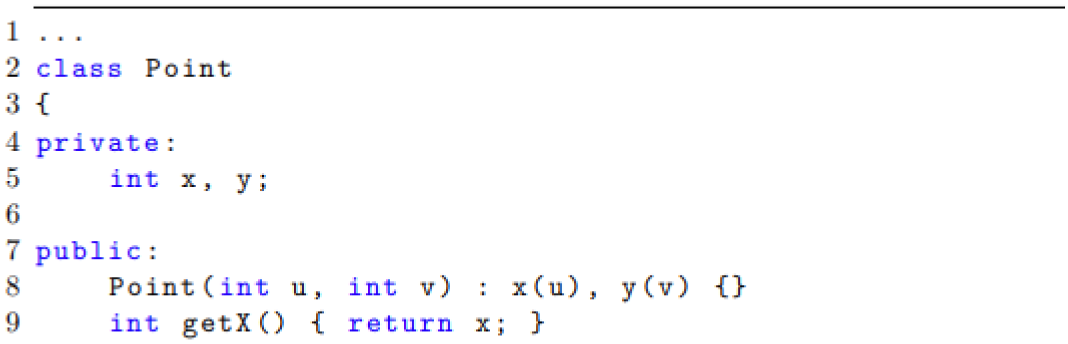
```cpp
    void setX(int newX) const { x = newX; }
};

int main()
{
    Point p(5, 3);
    p.setX(9001);
        cout << p.getX() << ' ' << p.getY();
    return 0;
}
```

3.

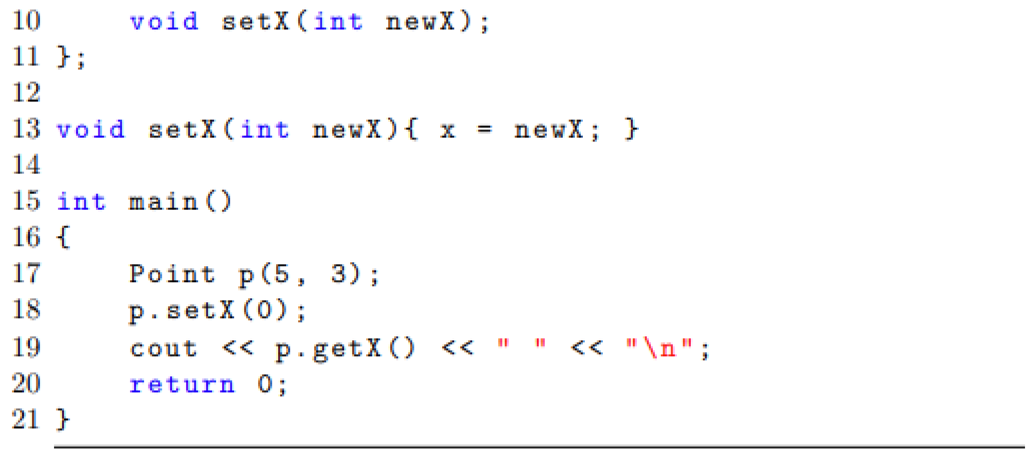


```cpp
...
class Point
{
private:
    int x, y;

public:
    Point(int u, int v) : x(u), y(v) {}
    int getX() { return x; }
    int getY() { return y; }
};

int main()
{
    Point p(5, 3);
    cout << p.x << " " << p.y << "\n";
    return 0;
}
```

4.

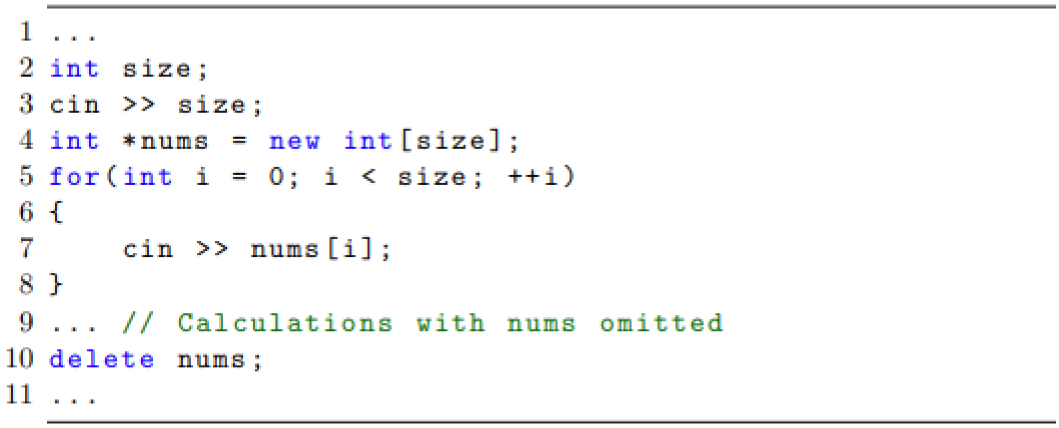


```cpp
...
class Point
{
private:
    int x, y;

public:
    Point(int u, int v) : x(u), y(v) {}
    int getX() { return x; }
```



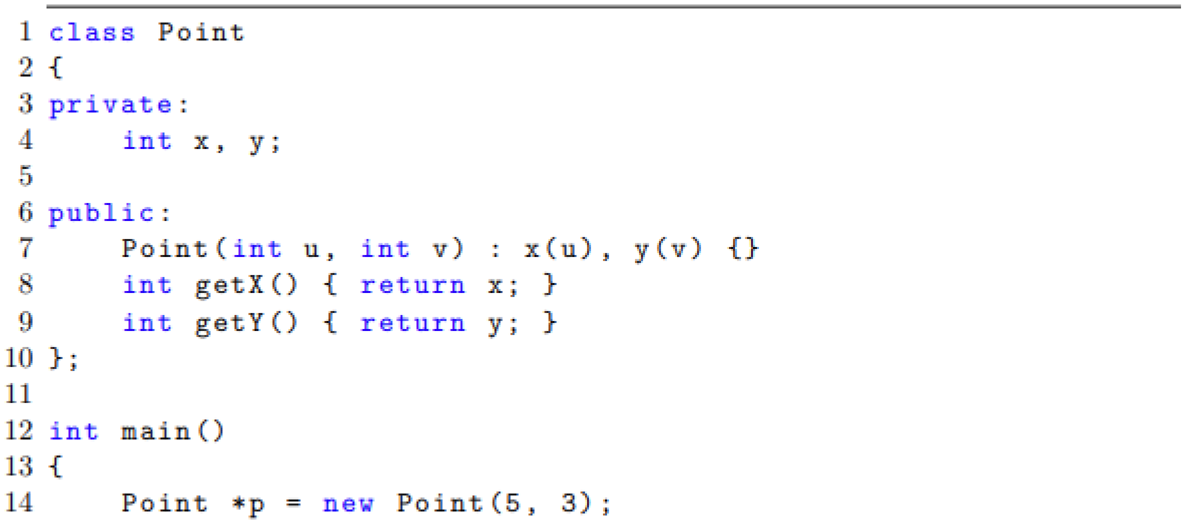
```cpp
    void setX(int newX);
};

void setX(int newX){ x = newX; }

int main()
{
    Point p(5, 3);
    p.setX(0);
    cout << p.getX() << " " << "\n";
    return 0;
}
```

5.

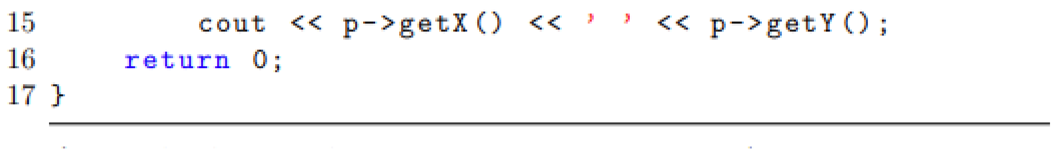


```cpp
...
int size;
cin >> size;
int *nums = new int[size];
for(int i = 0; i < size; ++i)
{
    cin >> nums[i];
}
... // Calculations with nums omitted
delete nums;
...
```

6.



```cpp
class Point
{
private:
    int x, y;

public:
    Point(int u, int v) : x(u), y(v) {}
    int getX() { return x; }
    int getY() { return y; }
};

int main()
{
    Point *p = new Point(5, 3);
```



```
        cout << p->getX() << ' ' << p->getY();
    return 0;
}
```

Submission:

Upload .cpp file of task 1 along with report containing flow chart of task 1 and answers to task 2.