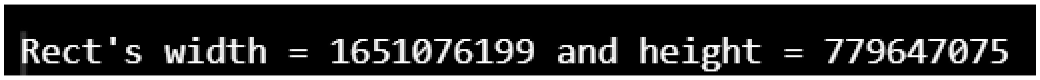
#1

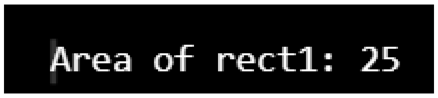
```
#include <iostream>
using namespace std;


class Rectangle {
 public:
 int width;
 int height;
};
int main() {

 Rectangle rect;
 cout << "Rect's width = " << rect.width << " and height = " << rect.height << endl;
 return 0;
}
```



```
Rect's width = 1651076199 and height = 779647075
```

#2

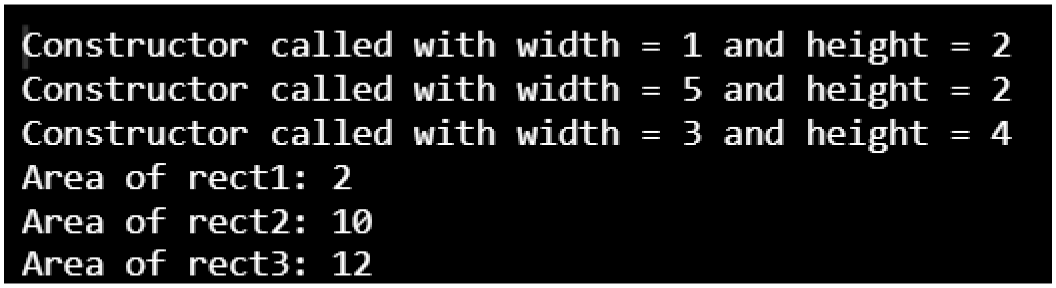
```
#include <iostream>
using namespace std;
class Rectangle {
 public:

Rectangle (int a, int b) : width(a), height(b) {}

 int area() {
 return width * height;
 }
private:
int width;
int height;
};
int main() {

 Rectangle rect1(5, 5);
 cout << "Area of rect1: " << rect1.area() << endl;
 return 0;
}
```



```
Area of rect1: 25
```

#3

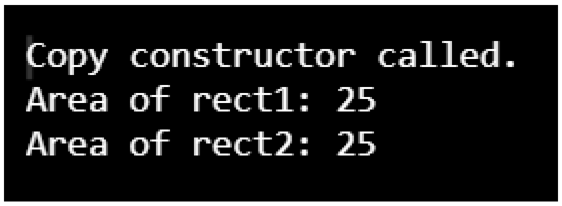
```
#include <iostream>
using namespace std;
class Rectangle {
 public:

Rectangle(int a = 1, int b = 2) : width(a), height(b) {
cout << "Constructor called with width = " << a << " and height = " << b << endl;
 }
 int area() {
 return width * height;
 }
 private:
 int width;
 int height;
};
int main() {

Rectangle rect1;

 Rectangle rect2(5);

 Rectangle rect3(3, 4);
 cout << "Area of rect1: " << rect1.area() << endl;
 cout << "Area of rect2: " << rect2.area() << endl;
 cout << "Area of rect3: " << rect3.area() << endl;
 return 0;
}
```



```
Constructor called with width = 1 and height = 2
Constructor called with width = 5 and height = 2
Constructor called with width = 3 and height = 4
Area of rect1: 2
Area of rect2: 10
Area of rect3: 12
```

#4

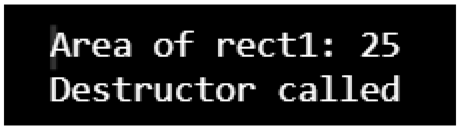
```
#include <iostream>
using namespace std;
class Rectangle {
 public:

Rectangle(int a, int b) : width(a), height(b) {
 }
Rectangle(const Rectangle &obj){
    width = obj.width;
    height = obj.height;
    cout << "Copy constructor called." << endl;
}
 int area() {
 return width * height;
 }
 private:
 int width;
 int height;
};
int main() {

Rectangle rect1(5,5);

Rectangle rect2 = rect1;

 cout << "Area of rect1: " << rect1.area() << endl;
 cout << "Area of rect2: " << rect2.area() << endl;

 return 0;
}
```



```
Copy constructor called.
Area of rect1: 25
Area of rect2: 25
```

#5

```
#include <iostream>
using namespace std;
class Rectangle {
 public:
 Rectangle(int a, int b) : width(a), height(b) {}

// Destructor
~Rectangle() {
cout << "Destructor called" << endl;
}
 int area() {
 return width * height;
 }
 private:
 int width;
 int height;
};
int main() {
 Rectangle rect1(5, 5);
 cout << "Area of rect1: " << rect1.area() << endl;
// Destructor is called automatically when main ends
 return 0;
}
```

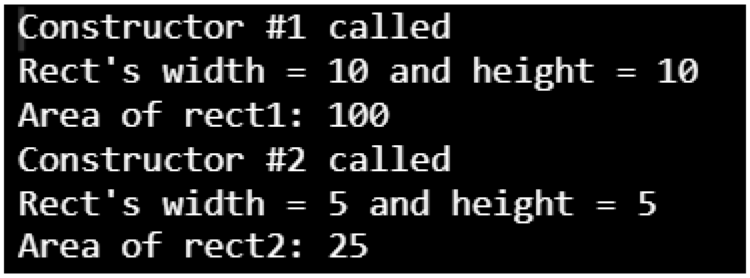


```
Area of rect1: 25
Destructor called
```

#6

```
#include <iostream>
using namespace std;
class Rectangle {
 public:


Rectangle () {
width = 10;
height = 10;
cout << "Constructor #1 called" << endl;
cout << "Rect's width = " << width << " and height = " << height << endl;
}


 Rectangle (int a, int b) : width(a), height(b) {
 cout << "Constructor #2 called" << endl;
 cout << "Rect's width = " << width << " and height = " << height << endl;
 }

 int area() {
 return width * height;
 }
 private:
 int width;
 int height;
};
int main() {
```

```
Rectangle rect1;
cout << "Area of rect1: " << rect1.area() << endl;
 Rectangle rect2(5, 5);
 cout << "Area of rect2: " << rect2.area() << endl;
 return 0;
}
```

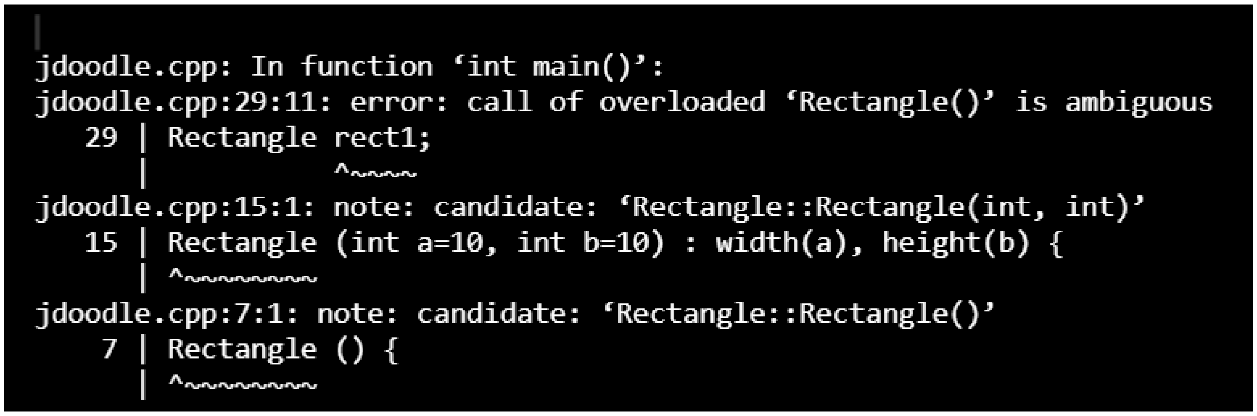


```
Constructor #1 called
Rect's width = 10 and height = 10
Area of rect1: 100
Constructor #2 called
Rect's width = 5 and height = 5
Area of rect2: 25
```

#7

```
#include <iostream>
using namespace std;
class Rectangle {
 public:


Rectangle () {
width = 10;
height = 10;
 cout << "Constructor #1 called" << endl;
 cout << "Rect's width = " << width << " and height = " << height << endl;
 }


Rectangle (int a=10, int b=10) : width(a), height(b) {
 cout << "Constructor #2 called" << endl;
 cout << "Rect's width = " << width << " and height = " << height << endl;
 }

 int area() {
 return width * height;
 }
 private:
 int width;
 int height;
};
int main() {
```

Rectangle rect1;
cout << "Area of rect1: " << rect1.area() << endl;
 Rectangle rect2(5, 5);
 cout << "Area of rect2: " << rect2.area() << endl;
 return 0;
}



```
jdoodle.cpp: In function ‘int main()’:
jdoodle.cpp:29:11: error: call of overloaded ‘Rectangle()’ is ambiguous
   29 | Rectangle rect1;
      |           ^~~~~
jdoodle.cpp:15:1: note: candidate: ‘Rectangle::Rectangle(int, int)’
   15 | Rectangle (int a=10, int b=10) : width(a), height(b) {
      | ^~~~~~~~~
jdoodle.cpp:7:1: note: candidate: ‘Rectangle::Rectangle()’
    7 | Rectangle () {
      | ^~~~~~~~~
```

#8
#include <iostream>
using namespace std;
class Rectangle {
 public:
 Rectangle(int a, int b) : width(a), height(b) {}

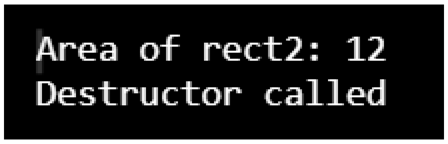
 ~Rectangle() {
 cout << "Destructor called" << endl;
 }
 int area() {
 return width * height;
 }
 private:
 int width;
 int height;
};
int main() {

Rectangle* rect2 = new Rectangle(3, 4);

 cout << "Area of rect2: " << rect2->area() << endl;

delete rect2;

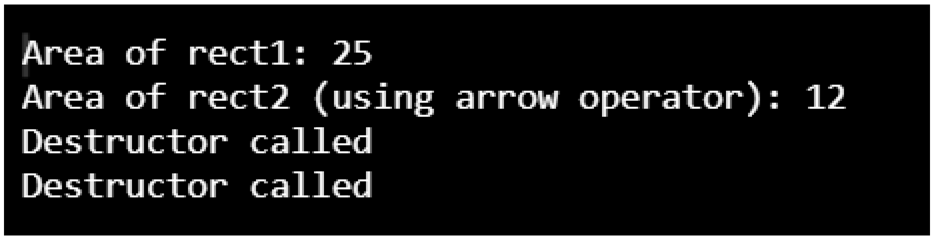
```
 return 0;
}
```



```
Area of rect2: 12
Destructor called
```

#9

```
#include <iostream>
using namespace std;
class Rectangle {
 public:
 Rectangle(int a, int b) : width(a), height(b) {}

 ~Rectangle() {
 cout << "Destructor called" << endl;
 }
 int area() {
 return width * height;
 }
 private:
 int width;
 int height;
};
int main() {

 Rectangle rect1(5, 5);
 cout << "Area of rect1: " << rect1.area() << endl;

 Rectangle* rect2 = new Rectangle(3, 4);
 cout << "Area of rect2 (using arrow operator): " << rect2->area()
<< endl;

 delete rect2;
 return 0;
}
```



```
Area of rect1: 25
Area of rect2 (using arrow operator): 12
Destructor called
Destructor called
```