

#1

```
#include <iostream>
```

```
using namespace std;
```

```
int main() {
```

```
    int x = 10;
```

```
    int *ptr_to_int = new int(20);
```

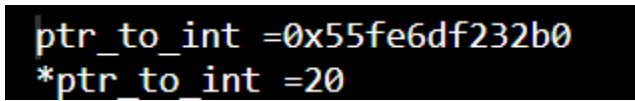
```
    /*ptr_to_int = 20;
```

```
    cout << "ptr_to_int =" << ptr_to_int << endl;
```

```
    cout << "*ptr_to_int =" << *ptr_to_int << endl;
```

```
    return 0;
```

```
}
```



```
ptr_to_int =0x55fe6df232b0  
*ptr_to_int =20
```

#2

```
#include <iostream>
```

```
using namespace std;
```

```
int main() {
```

```
    double* array = new double[4];
```

```
    array[0] = 1;
```

```
    array[1] = 2;
```

```
    array[2] = 3;
```

```
    array[3] = 4;
```

```
    for (int i=0; i <4;i++){
```

```
        cout << "array[" << i << "]=" << array[i] << endl;
```

```
    }
```

```
    return 0;
}
```

```
array[0]=1
array[1]=2
array[2]=3
array[3]=4
```

#3

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

```
int main() {
```

```
    string* strPtr = new string("Hello world");
```

```
    cout << "Length of the string: " << strPtr->length() << endl;
```

```
    delete strPtr;
```

```
    return 0;
}
```

```
Length of the string: 11
```

#4

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

```
int main() {
```

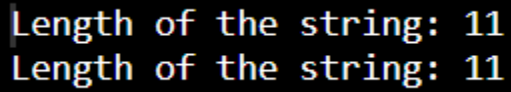
```
    string* strPtr = new string("Hello world");
```

```
    cout << "Length of the string: " << strPtr->length() << endl;
```

```
    cout << "Length of the string: " << (*strPtr).length() << endl;
```

```
delete strPtr;
```

```
    return 0;  
}
```



```
Length of the string: 11  
Length of the string: 11
```

#5

```
#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, height;  
};
```

```
int main() {
```

```
    Rectangle rect1(4, 9);
```

```
    Rectangle* rect2 = new Rectangle(6, 2);
```

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

```
    cout << "Area of rect2 (using dereference): " <<  
    (*rect2).area() << endl;
```

```
cout << "Area of rect2 (using arrow operator): " <<  
rect2->area() << endl;
```

```
delete rect2;
```

```
return 0;
```

```
}
```

```
Area of rect1: 36
```

```
Area of rect2 (using dereference): 12
```

```
Area of rect2 (using arrow operator): 12
```

```
Destructor called
```

```
Destructor called
```