

- Exercise

In this exercise, you get to experiment with and explore the operators `new` and `delete` according to your needs.

WE'LL COVER THE FOLLOWING ^

- Try It Out!
 - Code 1
 - Code 2

Try It Out!

Adjust operator `new` and `delete` according to your needs in the codes below:

Code 1

main.cpp

myNew.hpp

myNew2.hpp

myNew3.hpp

```
// overloadOperatorNewAndDelete.cpp

// #include "myNew.hpp"
// #include "myNew2.hpp"
#include "myNew3.hpp"

#include <iostream>
#include <string>

class MyClass{
    float* p= new float[100];
};

class MyClass2{
    int five= 5;
    std::string s= "hello";
};

int main(){

    int* myInt= new int(1998);
    double* myDouble= new double(3.14);
    double* myDoubleArray= new double[2]{1.1,1.2};

    MyClass* myClass= new MyClass;
    MyClass2* myClass2= new MyClass2;
```

```
delete myDouble;
delete [] myDoubleArray;

delete myClass;
delete myClass2;

getInfo();

}
```



Code 2

main.cpp

myNew4.hpp

myNew5.hpp

```
// overloadOperatorNewAndDelete2.cpp

#include "myNew4.hpp"
#include "myNew5.hpp"

#define new new(__FILE__, __LINE__)

#include <iostream>
#include <new>
#include <string>

class MyClass{
    float* p= new float[100];
};

class MyClass2{
    int five= 5;
    std::string s= "hello";
};

int main(){

    int* myInt= new int(1998);
    double* myDouble= new double(3.14);
    double* myDoubleArray= new double[2]{1.1,1.2};

    MyClass* myClass= new MyClass;
    MyClass2* myClass2= new MyClass2;

    delete myDouble;
    delete [] myDoubleArray;
    delete myClass;
    delete myClass2;

    dummyFunction();

    getInfo();

}
```





See [here](#) for a detailed description.

For further information, read [operator new](#) and [operator delete](#) .

In the next lesson, we will study a few sequential containers that are essential to embedded programming.