**YouTube tutorial 43 – Deconstructors**

https://www.youtube.com/watch?v=4P4Im0vF\_mU&index=43&list=PLAE85DE8440AA6B83

**3rd class – Cat.cpp:**

#include "stdafx.h"

#include "Cat.h"

#include <iostream>

using namespace std;

Cat::Cat()

{

cout << "I'm the constructor" << endl;

}

Cat::~Cat() {

cout << "I'm the deconstructor" << endl;

}

**2nd class – Cat.h:**

class Cat

{

public:

Cat();

~Cat();

};

**1st class – CommonApplication2.cpp:**

#include "stdafx.h"

#include "Cat.h"

#include <iostream>

using namespace std;

int main()

{

Cat cat;

cout << "Wazzup niggas" << endl;

return 0;

}

**Important notes:**

* The deconstructor runs in the end.
* Keep in mind that deconstructors don't take any variables, can't be overloaded,

and don't have any return values or "void".

**YouTube tutorial 45 – Member Initializers**

https://www.youtube.com/watch?v=53VYYMy-LBo&index=45&list=PLAE85DE8440AA6B83

**3rd class - Cat.cpp:**

#include "stdafx.h"

#include "Cat.h"

#include <iostream>

using namespace std;

Cat::Cat(int a, int b)

: regVar(a),

constVar(b)

{

cout << "regVar is: " << regVar << endl;

cout << "constVar is: " << constVar << endl;

}

**2nd class – Cat.h:**

class Cat

{

public:

Cat(int a, int b);

void print();

private:

int regVar;

const int constVar;

};

**1st class – CommonApplication2.cpp:**

#include "stdafx.h"

#include "Cat.h"

#include <iostream>

using namespace std;

int main()

{

Cat cat(5,3);

return 0;

}

**Important note:**

* Keep in mind how the code in grey’s structured
* By writing a semi-colon, you’re telling the program that you’ll be using member any member initializer
* One of the values is const, so it can’t be changed anymore