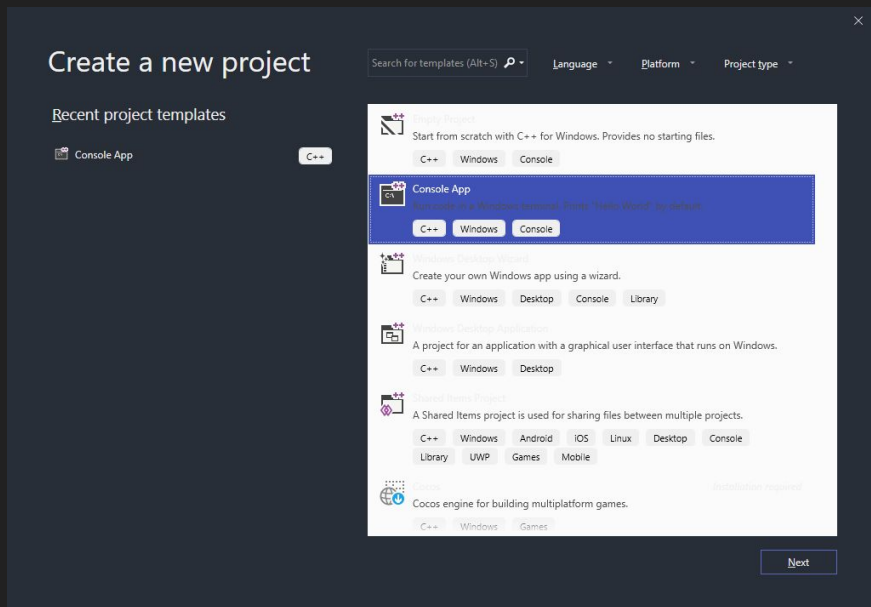


CPP Practice

420-J11-AS

New Project



```
#include <iostream>
```

```
int main()
{
    std::cout << "Hello World!\n";
}
```

```
// Hello World!
```

Free Function

```
#include <iostream>
```

```
int main()  
{  
    hello();  
}
```

```
void hello()  
{  
    std::cout << "Hello World!\n";  
}
```

```
// error C3861: 'hello': identifier not found
```

Forward Declaration

```
#include <iostream>
```

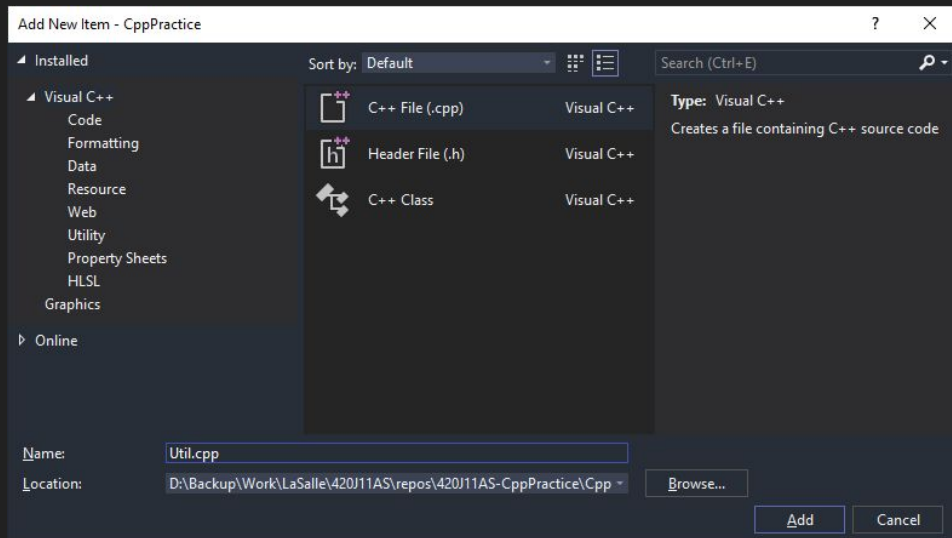
```
void hello();
```

```
int main()  
{  
    hello();  
}
```

```
void hello()  
{  
    std::cout << "Hello World!\n";  
}
```

```
// Hello World!
```

New Item - Util.cpp



```
// Util.cpp
```

```
#include <iostream>
```

```
void hello()
{
    std::cout << "Hello World!\n";
}
```

```
void bye()
{
    std::cout << "Bye World!\n";
}
```

Clean main

```
// CppPractice.cpp
```

```
void hello();  
void bye();
```

```
int main()  
{  
    hello();  
    bye();  
}
```

```
// Hello World!  
// Bye World!
```

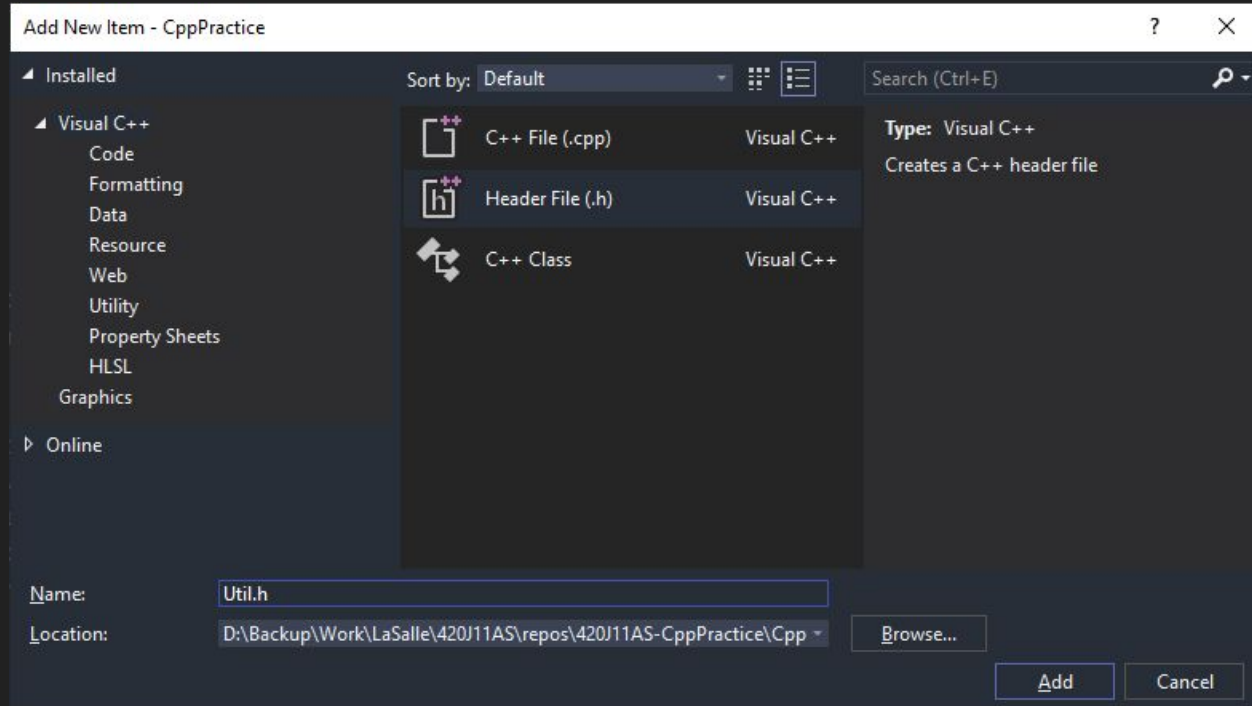
```
// CppPractice.cpp
```

```
#include "Util.cpp"
```

```
int main()  
{  
    hello();  
    bye();  
}
```

```
// fatal error LNK1169: one or more  
multiply defined symbols found
```

New Item - Util.h



Fixed main

```
// Util.h
```

```
void hello();  
void bye();
```

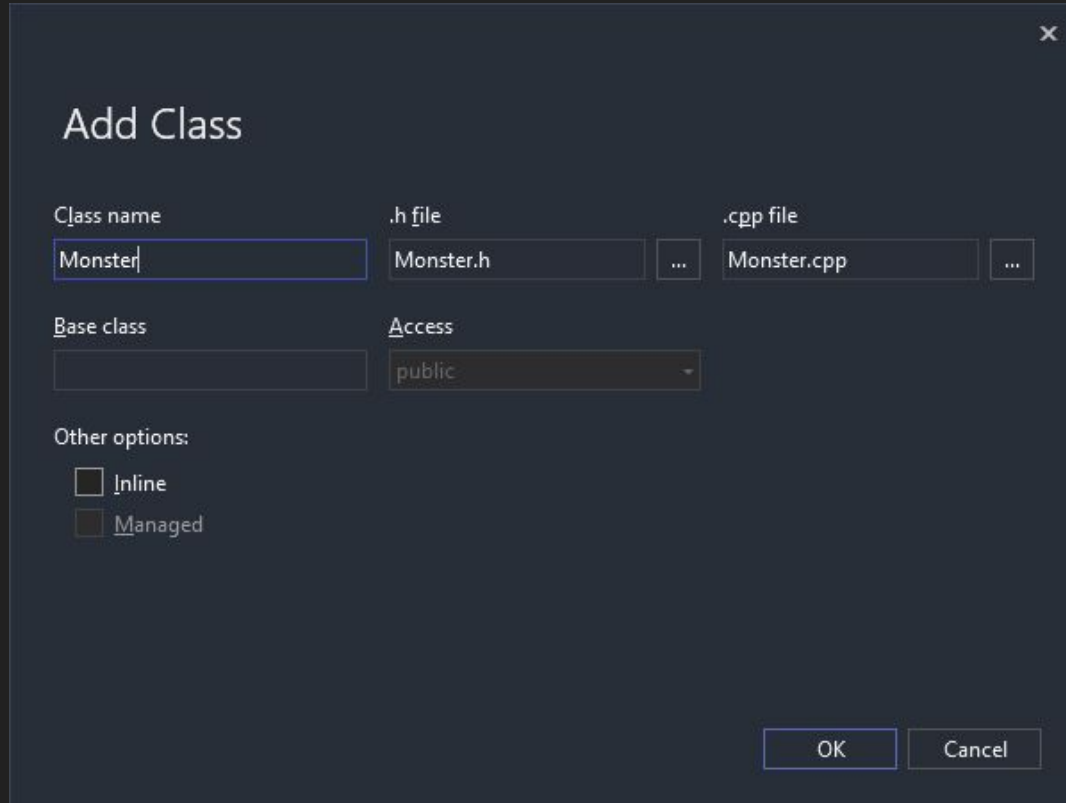
```
// CppPractice.cpp
```

```
#include "Util.h"
```

```
int main()  
{  
    hello();  
    bye();  
}
```

```
// Hello World!  
// Bye World!
```


New Class - Monster



A dark-themed dialog box titled "Add Class" with a close button (X) in the top right corner. The dialog contains several input fields and options for creating a new class.

Add Class

Class name
Monster

.h file
Monster.h ...

.cpp file
Monster.cpp ...

Base class
[Empty text box]

Access
public

Other options:

- ☐ Inline
- ☐ Managed

OK **Cancel**

Monster Header and Implementation

```
// Monster.h
```

```
#include <string>
```

```
class Monster
```

```
{
```

```
public:
```

```
    Monster(std::string);
```

```
    std::string name;
```

```
};
```

```
// Monster.cpp
```

```
#include "Monster.h"
```

```
#include <iostream>
```

```
Monster::Monster(std::string givenName)
```

```
{
```

```
    name = givenName;
```

```
    std::cout << "My name is " << name << ".\n";
```

```
}
```

Hero Class - Header and Implementation

// Hero.h

```
#include "Monster.h"
```

```
class Hero
```

```
{
```

```
public:
```

```
    void attack(Monster);
```

```
};
```

// Hero.cpp

```
#include "Hero.h"
```

```
#include <iostream>
```

```
using namespace std;
```

```
void Hero::attack(Monster monster)
```

```
{
```

```
    cout << "Hero attacks " << monster.name << "\n";
```

```
}
```

Update main

```
// CppPractice.cpp
#include "Util.h"
#include "Monster.h"
#include "Hero.h"
```

```
Monster monster("Le Goblin");
Hero hero;
```

```
int main()
{
    hello();

    hero.attack(monster);

    bye();
}
```

```
// error C2011: 'Monster': 'class'
type redefinition
```

Header Guard

```
// Monster.h  
#pragma once
```

```
#ifndef MONSTER_H  
#define MONSTER_H
```

```
#include <string>
```

```
class Monster  
{  
public:  
    Monster(std::string);  
  
    std::string name;  
};  
#endif
```

```
// My name is Le Goblin.  
// Hello World!  
// Hero attacks Le Goblin.  
// Bye World!
```

Pointers - Indirection / Dereference

```
// CppPractice.cpp
#include "Util.h"
#include "Monster.h"
#include "Hero.h"
```

```
Monster* monster;
```

```
Hero hero;
```

```
int main()
```

```
{
```

```
    hello();
```

```
    monster = new Monster("Le Goblin");
```

```
    hero.attack(*monster);
```

```
    bye();
```

```
}
```

```
// Hello World!
// My name is Le Goblin.
// Hero attacks Le Goblin.
// Bye World!
```

Pointers - Member Selection

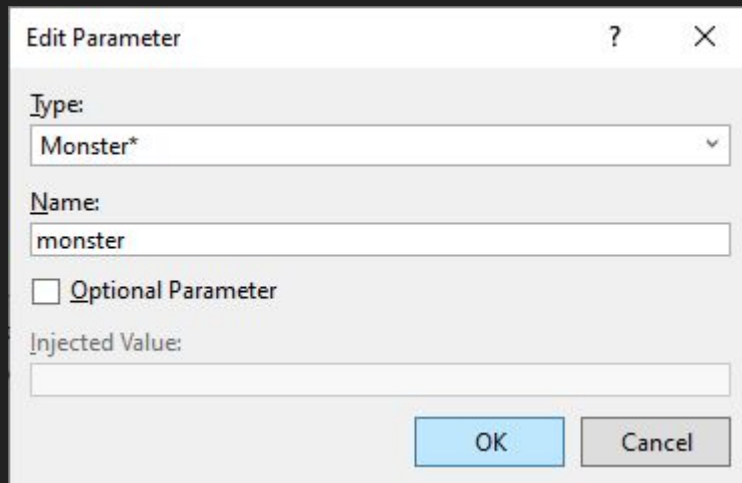
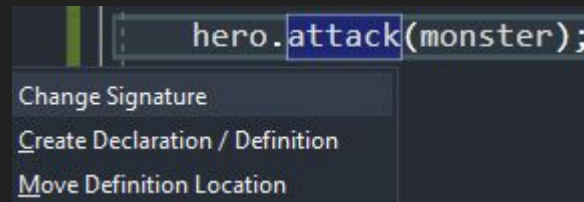
```
// CppPractice.cpp
#include "Util.h"
#include "Monster.h"
#include "Hero.h"
```

```
Monster* monster;
Hero hero;
```

```
int main()
{
    hello();

    monster = new Monster("Le Goblin");
    hero.attack(monster);

    bye();
}
```



Templates - Vector

```
// Monster.h
#include "Monster.h"
#include "Hero.h"
#include <vector>
using namespace std;

vector<Monster*> monsters;
Hero hero;

int main()
{
    monsters.push_back(new Monster("Alice"));
    monsters.push_back(new Monster("Bob"));
    monsters.push_back(new Monster("Charlie"));

    hero.attack(monsters[1]);
}
```

```
// My name is Alice.
// My name is Bob.
// My name is Charlie.
// Hero attacks Bob.
```


Range-Based For

```
// Monster.h
#include "Monster.h"
#include "Hero.h"
#include <vector>
using namespace std;

vector<Monster*> monsters;
Hero hero;

int main()
{
    monsters.push_back(new Monster("Alice"));
    monsters.push_back(new Monster("Bob"));
    monsters.push_back(new Monster("Charlie"));

    for (auto monster : monsters)
        hero.attack(monster);
}
```

```
// My name is Alice.
// My name is Bob.
// My name is Charlie.
// Hero attacks Alice.
// Hero attacks Bob.
// Hero attacks Charlie.
```

Const Correctness

```
// Hero.h
#include "Monster.h"

class Hero
{
public:
    void attack(const Monster * monster) const;

private:
    int level;
};
```

Const Parameter

```
// Hero.h
#include "Hero.h"
#include <iostream>
using namespace std;

void Hero::attack(const Monster * monster) const
{
    monster->name = "oof";
    cout << "Hero attacks " << monster->name << ".\n";
}
```

```
// error C2678: binary '=': no operator found which takes a left-hand operand of
type 'const std::string' (or there is no acceptable conversion)
```

Const Member Function

```
// Hero.h
#include "Hero.h"
#include <iostream>
using namespace std;

void Hero::attack(const Monster * monster) const
{
    level++;
    cout << "Hero attacks " << monster->name << ".\n";
}
```

```
// error C3490: 'level' cannot be modified because it is being accessed through a
const object
```

References

- <https://www.learncpp.com/cpp-tutorial/forward-declarations/>
- <https://docs.microsoft.com/en-us/cpp/cpp/header-files-cpp?view=vs-2019>
- <https://www.learncpp.com/cpp-tutorial/header-guards/>
- <http://www.cplusplus.com/doc/tutorial/pointers/>
- <https://docs.microsoft.com/en-us/cpp/standard-library/vector-class?view=vs-2019>
- <https://docs.microsoft.com/en-us/cpp/cpp/range-based-for-statement-cpp?view=vs-2019>
- <https://isocpp.org/wiki/faq/const-correctness>