CS 161C++
Separate Compilation and Header Files

The next step after breaking a program into multiple functions is to break it into separate files so that multiple people can work on it or you can use the functions in other projects.

The mechanism used for this in C++ is to create a header file with the declarations and any constants and a code file with the function definitions.

Header Files

Header files in C++ typically end in a .h or a .hpp. The use of .h dates back to the time of C programs which are denoted with a .c extension. Since C++ files are typically denoted with a .cpp file extension, it is common to use a .hpp for a C++ header. Sometimes you will still find people using .h files due to the historical connections between C and C++.

A header file normally has ***guards*** on it. This is the practice of including the following compiler directives at the start and end of the file.

```
#ifndef MYFUN_HEADER
#define MYFUN_HEADER

  // your code goes here

#endif //MYFUN_HEADER
```

The result of this is that the compiler, when reading this file for the first time to include it, will define the symbol "MYFUN_HEADER" and include the contents of this file in the program file being read. The next time the compiler tries to include this file, it will find that that symbol has already been defined and will skip down to the endif directive and not copy any of the contents.

Header files are used to declare functions, classes, and global constants. (We will talk about classes in CS 162).

Code Files

Since the function declaration is included in the header file, there needs to be a place for the function (or class) definitions. This is the code file. As with any C++ file, these files typically have a .cpp file extension.

Including header files

Both main and the code files should include the header file, so that they share the function declarations (to make sure that they are consistently defined) and to allow the use of any global constants.

This is done by the use of the include directive as follows:

```
#include "myfun.hpp"
```

Note that the function name is included in double quote marks. Standard library include files are denoted by angle brackets as in

```
#include <iostream>
```

This tells the compiler to find the appropriate file some default system location. The include directive using double quotes tells the compiler to find the file using the relative or absolute pathname in quotes.

Creating Header and Code Files

How you create a new header or code file depends on the IDE you are using. Go under the file tab and select new file for Xcode and new C++ file for Code::Blocks

Consider the following program:

```
#include <iostream>

using namespace std;

// function definition at top
int myfun(int num1, int num2)
{
   int result;
   result = num1 + num2;
   return result;
}

int main()
{
   int alpha = 6, beta = 4;

   cout << myfun(alpha,beta) << endl;

   return 0;
}
```

The first step is to separate the function into declaration and definition and move the definition to the bottom of the program file as shown below.

```
#include <iostream>

using namespace std;

// function definition at top
int myfun(int num1, int num2);


int main()
{
   int alpha = 6, beta = 4;

   cout << myfun(alpha,beta) << endl;

   return 0;
}

// function definition
int myfun(int num1, int num2)
{
   int result;
   result = num1 + num2;
   return result;
}
```

The next step is to move the function declaration into the header file and replace it with an include statement as shown below.

main.cpp

```
#include <iostream>

using namespace std;

// include header file
#include "myfun.hpp"

int main()
{
   int alpha = 6, beta = 4;

   cout << myfun(alpha,beta) << endl;

   return 0;
}

// function definition
int myfun(int num1, int num2)
{
   int result;
   result = num1 + num2;
   return result;
}
```

myfun.hpp

```
#ifndef myfun_hpp
#define myfun_hpp

// function definition in header
int myfun(int num1, int num2);

#endif /* myfun_hpp */
```

The final step is to move the function definition to the code file as shown below.

main.cpp

```
#include <iostream>

using namespace std;

// include header file
#include "myfun.hpp"

int main()
{
   int alpha = 6, beta = 4;

   cout << myfun(alpha,beta) << endl;

   return 0;
}
```

myfun.hpp

```
#ifndef myfun_hpp
#define myfun_hpp

// function definition in header
int myfun(int num1, int num2);

#endif /* myfun_hpp */
```

myfun.cpp

```
#include "myfun.hpp"

// function definition in code file
int myfun(int num1, int num2)
{
   int result;
   result = num1 + num2;
   return result;
}
```

Make sure that the code file has been included as part of your project.