CS 101: Computer Programming and Utilization

Shivaram Kalyanakrishnan
(Abhiram Ranade's slides, borrowed and edited)
Lecture 22

Today's Lecture

- C++ preprocessor directives (#define, #include)
- Function declarations/prototypes
- Splitting code across files
- C++ without simplecpp

Function declaration

- A function declaration is essentially the definition without the body.
- Example: declaration of gcd function: `int gcd(int m, int n);`
- Also acceptable: `int gcd(int, int);`
- The declaration tells the compiler that if the name `gcd` appears later, it will be a function and take 2 arguments of the specified type.
    - Compiler can now check if the name is used correctly in the rest of the file.
- If a file contains a call to a function but does not contain its definition:
    - It can only be partially compiled into an object module.
    - To get an executable program, all the object modules containing all called functions must be linked together.
- Other names for “declaration” : Signature, Prototype
- Example next.

Example of code split over multiple files

- File gcd.cpp

```
int gcd(int m, int n)
{ … }
```

- File lcm.cpp

```
int gcd(int, int);
int lcm(int m, int n){
   return m*n/
gcd(m,n);}
```

- File main.cpp

```
int lcm(int, int);
int main(){
cout << lcm(36,24) <<
endl;
}
```

- Function definitions, function declarations
- Each file has declarations of called functions.
- To compile and link all files together:

```
s++ main.cpp lcm.cpp gcd.cpp
```

- To compile each file separately:

```
s++ -c lcm.cpp
```

- -c : produce lcm.o (object module).
- To get executable from Object modules:

```
s++ main.o lcm.o gcd.o
```

What you typically do: s++ pgm.cpp m1.o m2.o

- Compile your program pgm.cpp
- Link it to object modules developed by others

Your pgm.cpp must have the right declarations...

Header files

- Tedious to remember what declaration to include in each file.
- Instead, we can put all declarations into a header file, and “include” the header file into every file.
- Header file gcdlcm.h

```
int gcd(int, int);
int lcm(int,int);
```

- The directive “`#include filename`”
  - gets replaced by the content of the named file.
  - It is acceptable if we declare functions that do not get used.
  - It is acceptable if we have both a declaration and then the definition of a function in the same file.

- **File gcd.cpp**

```
#include “gcdlcm.h”
int gcd(int m, int n){ … }
```

- **File lcm.cpp**

```
#include “gcdlcm.h”
int lcm(int m, int n){ … }
```

- **File main.cpp**

```
#include <simplecpp>
#include “gcdlcm.h”
int main(){ ...}
```

More on header files

- Header files customarily have the suffix .h or .hpp., or no suffix.
- If header file is mentioned in “ “, it is picked up from the current directory.
- If it is mentioned in < >, it is picked up from some standard place, e.g. `simplecpp`

The main program is also a function

- In C++ the standard way to write the main program is to write it as a function.
  - The function must be named `main`.
  - Its return type must be `int`.
  - For now, it does not take any arguments.

  Instead of `main_program{ xxx },` you would write: `int main() { xxx }`
- Simplecpp translates `main_program` into the phrase "`int main()`"
- Simplecpp provides this feature so that you don't need to understand functions etc. on the first day.
- The function `main` is allowed not to have a return statement.
  - The value returned has little consequence anyway.
- From now on we will not use `main_program`, but use `main`.

Using C++ without simplecpp

- If you use C++ without `simplecpp`, you will not be able to do graphics.
- Also, when you write `#include <simplecpp>` it itself includes the following lines for you:

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

- These lines are useful:
  - The names `cin, cout, endl` are defined in the namespace `std`, in the standard header file `iostream`.
- If you do not include `simplecpp`, be sure to write these lines, that is enough!
- The header file `cmath` includes math functions such as `sqrt, sin, abs.`

Simple example: Using C++ without simplecpp

```
#include <iostream>

using namespace std;

int main(){

  int n;

   cin >> n;

   cout << n*n*n <<

              endl;

}
```

```
#include <iostream>


int main(){
   int n;
   std::cin >> n;
   std::cout <<
n*n*n <<

std::endl;
}
```