EECS402
Compiling and Running

Andrew M. Morgan

Compiler/Linker Used For This Course

- There are many compilers available
- For this course, we use a compiler named “g++”, available on the UNIX operating system
- Usage:
  - g++ -Wall filePrefix.cpp –o filePrefix
    - -Wall means to display all warnings detected (some are suppressed by default)
    - -o <fileName> specifies the name of the resulting output (executable) file
- Example:
  - g++ -Wall genResults.cpp –o genResults
  - Compiles and links the C++ source code in a file named genResults.cpp and creates an executable file named genResults
- Output executable is only created if no compile or link errors are found!

Example Program

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cour << "Min: " << x << " Max: " << y << endl
17
18   return (0);
19 }
20
21 void swap(int &val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val3 = temp;
27 }
```

Compile Command

My UNIX Prompt

```
[ 8 ] temp -> g++ -Wall myProgram.cpp -o myProgram
myProgram.cpp: In function `int main()':
myProgram.cpp:16: error: `cour' undeclared (first use this function)
myProgram.cpp:16: error: (Each undeclared identifier is reported only once for
  each function it appears in.)
myProgram.cpp:18: error: syntax error before `return'
myProgram.cpp: In function `void swap(int&, int&)':
myProgram.cpp:26: error: `val3' undeclared (first use this function)
[ 9 ] temp ->
```

Line Number

Error Description

Fixing Compile Errors

- *ALWAYS* start with the first error!!
  - Later errors are often a result of the compiler getting confused from earlier errors
- *Usually* try to fix the first error and then re-compile
  - Don't try to track down the first 10 errors without compiling in between
  - Since later errors often result from prior errors, you could be looking for an error that isn't really there
- The line number provided by the compiler is an *indication* of where the problem is
  - The error is actually often one or more lines prior to the line number reported

Example Program

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cour << "Min: " << x << " Max: " << y << endl
17
18   return (0);
19 }
20
21 void swap(int &val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val3 = temp;
27 }
```

```
[ 7 ] temp -: !g
g++ -Wall myProgram.cpp -o myProgram
myProgram.cpp: In function `int main()':
myProgram.cpp:16: error: `cour' undeclared (first use this function)
myProgram.cpp:16: error: (Each undeclared identifier is reported only once for
  each function it appears in.)
myProgram.cpp:18: error: syntax error before `return'
myProgram.cpp: In function `void swap(int&, int&)':
myProgram.cpp:26: error: `val3' undeclared (first use this function)
[ 8 ] temp -:
```

This error usually means you forgot to declare a variable OR mistyped the identifier of the variable you intended to reference

Example Program

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cour << "Min: " << x << " Max: " << y << endl
17
18   return (0);
19 }
20
21 void swap(int &val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val3 = temp;
27 }
```

```
[ 7 ] temp -: !g
g++ -Wall myProgram.cpp -o myProgram
myProgram.cpp: In function `int main()':
myProgram.cpp:16: error: `cour' undeclared (first use this function)
myProgram.cpp:16: error: (Each undeclared identifier is reported only once for
  each function it appears in.)
myProgram.cpp:18: error: syntax error before `return'
myProgram.cpp: In function `void swap(int&, int&)':
myProgram.cpp:26: error: `val3' undeclared (first use this function)
[ 8 ] temp -:
```

This error usually means you forgot to declare a variable OR mistyped the identifier of the variable you intended to reference

Example Program

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cout << "Min: " << x << " Max: " << y << endl
17
18   return (0);
19 }
20
21 void swap(int &val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val3 = temp;
27 }
```

```
[ 8 ] temp -: g++ -Wall myProgram.cpp -o myProgram
myProgram.cpp: In function `int main()':
myProgram.cpp:18: error: syntax error before `return'
myProgram.cpp: In function `void swap(int&, int&)':
myProgram.cpp:26: error: `val3' undeclared (first use this function)
myProgram.cpp:26: error: (Each undeclared identifier is reported only once for
  each function it appears in.)
[ 9 ] temp -:
```

Syntax errors occur when you didn't follow the rules of C++. Often due to things like missing parentheses around an expression in an if statement, or missing semi-colons, or mis-matched curly braces, etc.

Example Program

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cout << "Min: " << x << " Max: " << y << endl
17
18   return (0);
19 }
20
21 void swap(int &val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val3 = temp;
27 }
```

```
[ 8 ] temp -: g++ -Wall myProgram.cpp -o myProgram
myProgram.cpp: In function `int main()':
myProgram.cpp:18: error: syntax error before `return'
myProgram.cpp: In function `void swap(int&, int&)':
myProgram.cpp:26: error: `val3' undeclared (first use this function)
myProgram.cpp:26: error: (Each undeclared identifier is reported only once for
  each function it appears in.)
[ 9 ] temp -:
```

- Compiler realizes there is an error when it gets to line 18
- BUT line 18 is fine.
- Compiler got confused from an earlier line.

Syntax errors occur when you didn't follow the rules of C++. Often due to things like missing parentheses around an expression in an if statement, or missing semi-colons, or mis-matched curly braces, etc.

Example Program

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cout << "Min: " << x << " Max: " << y << endl;
17
18   return (0);
19 }
20
21 void swap(int &val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val3 = temp;
27 }
```

```
[ 4 ] temp -: g++ -Wall myProgram.cpp -o myProgram
myProgram.cpp: In function `void swap(int&, int&)':
myProgram.cpp:26: error: `val3' undeclared (first use this function)
myProgram.cpp:26: error: (Each undeclared identifier is reported only once for
  each function it appears in.)
[ 5 ] temp -:
```

Just keep fixing errors until there are no more..

Example Program

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cout << "Min: " << x << " Max: " << y << endl;
17
18   return (0);
19 }
20
21 void swap(int &val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val3 = temp;
27 }
```

```
[ 4 ] temp -: g++ -Wall myProgram.cpp -o myProgram
myProgram.cpp: In function `void swap(int&, int&)':
myProgram.cpp:26: error: `val3' undeclared (first use this function)
myProgram.cpp:26: error: (Each undeclared identifier is reported only once for
  each function it appears in.)
[ 5 ] temp -:
```

Just keep fixing errors until there are no more..

Example Program

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cout << "Min: " << x << " Max: " << y << endl;
17
18   return (0);
19 }
20
21 void swap(int &val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val2 = temp;
27 }
```

```
[ 3 ] temp -: g++ -Wall myProgram.cpp -o myProgram
/tmp/ccCqT5PG.o(.text+0x31): In function `main':
: undefined reference to `swap(int, int&)'
collect2: ld returned 1 exit status
[ 4 ] temp -:
```

As the compiler gets farther along, newly discovered errors may turn up.

This error is a *linker* error, so it looks different than a compiler error. No line numbers, etc

Linker errors usually involve missing function bodies, inconsistent function signatures, etc..

Example Program

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cout << "Min: " << x << " Max: " << y << endl;
17
18   return (0);
19 }
20
21 void swap(int &val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val2 = temp;
27 }
```

```
[ 3 ] temp -: g++ -Wall myProgram.cpp -o myProgram
/tmp/ccCqT5PG.o(.text+0x31): In function `main':
: undefined reference to `swap(int, int&)'
collect2: ld returned 1 exit status
[ 4 ] temp -:
```

As the compiler gets farther along, newly discovered errors may turn up.

This error is a *linker* error, so it looks different than a compiler error. No line numbers, etc

Linker errors usually involve missing function bodies, inconsistent function signatures, etc..

Example Program

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cout << "Min: " << x << " Max: " << y << endl;
17
18   return (0);
19 }
20
21 void swap(int val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val2 = temp;
27 }
```

```
[ 5 ] temp -: g++ -Wall myProgram.cpp -o myProgram
[ 6 ] temp -:
```

No more compile or link errors!!!

You can now run the executable file created to check its results!

Executing Your Program

- To run your program:
  - In the directory containing the executable, type a dot, a slash, and the name of the executable
  - The "./" tells the computer to run the executable with the name provided in the current directory
  - Without the "./" a different executable with the same name may be executed and the results can be confusing
- Examples:
  - ./myProgram
  - ./runTheProgram

Back to the Example

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cout << "Min: " << x << " Max: " << y << endl;
17
18   return (0);
19 }
20
21 void swap(int val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val2 = temp;
27 }
```

```
[ 10 ] temp -: g++ -Wall myProgram.cpp -o myProgram
[ 11 ] temp -: ./myProgram
Min: 45 Max: 45
[ 12 ] temp -:
```

NOTE: Results are not what we expected!

No compile/link errors does NOT imply the program is correct!

This type of error is called a logic error.

Back to the Example

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cout << "Min: " << x << " Max: " << y << endl;
17
18   return (0);
19 }
20
21 void swap(int val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val2 = temp;
27 }
```

```
[ 10 ] temp -: g++ -Wall myProgram.cpp -o myProgram
[ 11 ] temp -: ./myProgram
Min: 45 Max: 45
[ 12 ] temp -:
```

NOTE: Results are not what we expected!

No compile/link errors does NOT imply the program is correct!

This type of error is called a logic error.

Back to the Example

```
1  #include <iostream>
2  using namespace std;
3
4  void swap(int &val1, int &val2);
5
6  int main()
7  {
8    int x = 45;
9    int y = 30;
10
11   if (y < x)
12   {
13     swap(x, y);
14   }
15
16   cout << "Min: " << x << " Max: " << y << endl;
17
18   return (0);
19 }
20
21 void swap(int &val1, int &val2)
22 {
23   int temp;
24   temp = val1;
25   val1 = val2;
26   val2 = temp;
27 }
```

```
[ 10 ] temp -: g++ -Wall myProgram.cpp -o myProgram
[ 11 ] temp -: ./myProgram
Min: 30 Max: 45
[ 12 ] temp -:
```

NOTE: Results are not what we expected!

No compile/link errors does NOT imply the program is correct!

This type of error is called a logic error.