

# For Friday

- Skim Savitch 4-6
- Turn in student information sheet
- Send an email to [mecaliff@ilstu.edu](mailto:mecaliff@ilstu.edu) with
  - Your name
  - ITK 279

# Hello World in C++

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

int main(int argc, char *argv[])
{
    cout << "Hello world!" << endl;
    return 0;
}
```

# Notes

- `#include` is used to access libraries and other code. It actually copies the text of the file into your file, so you must not include things more than once
- Namespaces are used to organize large projects. All identifiers from the various C++ libraries are in the `std` namespace.
- Functions in C++ do not have to be methods (members of a class). The main function is never a member of a class.

# Input/Output

- `cout` is the reference to standard output (typically the monitor)
- `cin` is the reference to standard input (typically the keyboard)
- The insertion operator, `<<`, lets you write to an output stream.
- The extraction operator, `>>`, lets you read from an input stream.

# Input Example

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

int main()
{
    int age;
    cout << "Please enter your age: ";
    cin >> age;
    cout << "You entered " << age << endl;
    return 0;
}
```

# Notes

- The arguments to main are optional.
- You can handle multiple inputs at once (just like outputs). Typically only useful with files.

# Primitive Data Types

- Basic integral types:
  - char
  - short
  - int
  - long
- Basic floating point types
  - float
  - double
- Boolean type
  - bool
  - Technically numeric – has values true and false but they are 1 and 0
  - In C++, 0 is false and everything else true

# Strings

- Traditional “C” strings
  - Arrays of chars ending in the null char (“\0”)
  - Required for things like file names
- String class
  - Somewhat similar to Java’s
  - Not immutable

# Uninitialized Variables

- Not an error
- Not initialized
- You just get whatever value happens to have been in those bits of memory . . .

# Output

- “\n” vs. endl
- Formatting
- cerr and cout

# Boolean Expressions

- Mostly the same
- Key issue is that they don't have to be type bool
- Creates some convenient expressions, but can lead to harder to spot errors.

# enum

- Enumerations are MUCH simpler to use than in Java
- Typical goal is to define a set (usually in sequence) of symbolic constants
- Often used for switch statements

# Access to Functions

- Must have a prototype or actual definition of the function in the file before the function call

# Random Numbers

- Not going to take time for them in class – but they will be needed for some programs
- Note that their use is explained starting on page 103

# Good Function Programming

- Put prototypes at the top of the file
- DO use the names as well as the types in the prototype
- Be sure to provide a comment explaining the use of the function with the prototype

# Scope

- File scope (global scope) is generally appropriate only for functions and constants
- Avoid the use of global variables
- Variables are local to the block in which they are declared
- Nested scopes may result in some variables being invisible