Introduction to programming with C++

Learn
• Fundamental programming concepts
• Key techniques
• Basic C++ facilities
By the end of the course, you will be able to:
• Write small C++ programs
• Read much larger programs
• Learn the basics of many other languages
• Proceed to advanced C++ courses

Requirements
• Attendance and participation
• Lectures and lab sessions
• Labs assignments – roughly 6-8 across semester
• Quizzes – each 15 minutes, 5 across semester
• Final project
• Exams – 1 midterm, 1 final

• Academic integrity – may discuss assignments with your classmates, but you MUST write all your code and all your answers yourself

How to succeed in class
Ask questions
• In class
• In office hours JMH 328A, tutor room JMH 301
• Study together and discuss assignments with each other (without plagiarizing!)

Textbook
• Read and re-read the material
• Complete practice problems
Start coding and studying early

Course textbook
Problem Solving With C++
Ninth Edition
Walter Savitch

Course website
http://storm.cis.fordham.edu/leeds/cisc1600
Go online for
• Announcements
• Lecture slides
• Course materials/handouts
• Assignments
Instructor

Prof. Daniel Leeds
dleeds@fordham.edu
Office hours: Tues 11a-12p, Thu 12:30-1:30p
Office: JMH 328A

“Two courses in one”: CISC 1600 and CISC 1610
CISC 1600 and 1610 cover lecture and lab
• Lecture room: JMH 342
• Lab room: JMH 331
• We may work in lab during lecture time
• We may have lecture during lab time
Listen for announcements during class/online!

A program provides a computer with a set of simple instructions to achieve a goal

Programs are everywhere
On your computer:
• Web browser
  – Request and display information from distant sites
• Word processor
  – Record text, change appearance, save to disk

Programs are everywhere
In the dining hall:
• Cashier
  – Compute price of food purchase, charge payment to account, (if pay cash: compute change)
• HVAC
  – Monitor temperature, adjust A/C or heating

Programs are everywhere
In humans:
• Sports
  – When to run, where to run; when to pass, who to pass to; when to shoot
• The brain
  – Neurons working together to combine information about an image to recognize a dog or a car

Head detector
Tail detector
dog

Head detector
Tail detector
Computer system structure

Central processing unit (CPU) – performs all the instructions
Memory – stores data and instructions for CPU
Input – collects information from the world
Output – provides information to the world

C++ – high-level language

- High-level language
  - Uses words to describe instructions
  - More intuitive to people
  - Computer-independent
- Machine-language
  - Uses binary to describe instructions
  - Less intuitive to people
  - Computer-dependent

Why C++?

- Popular modern programming language
- In use since 1980's
- Similar structure to many/most other popular languages (Java, C#, Perl, Python)

Why C++?

Some programming history:
- C++ developed as improvement on C
- C developed as improvement on B
- B developed as improvement on ...
- BCPL – Basic Computer Programming Language
- Various languages before BCPL – ADA, COBOL, FORTRAN

Course outline

- Programming basics, input/output, arithmetic
- Conditional statements
- Loops
- Modularity – functions
- Complex data – arrays, strings, and classes

Throughout the semester:
- Proper programming style

Programming basics

- Program structure and components
- Output text
- Variables
- Input information
- Perform arithmetic
- Type safety
Our first program: “Hello world!”

```cpp
// include library of standard input and output commands
#include <iostream>
using namespace std;

int main()
{
    // Begin main function
    cout << "Hello world!\n";  // output "Hello world!"
    return 0;                 /* indicate successful
                              program completion */
}  // End main function
```

> ./myProgram
Hello world!
>

The components of “Hello world!”

- Comments  //, /* */
- main function
- Preprocessor directives #include

Using comments

```cpp
// include library of standard input and output commands
#include <iostream>
using namespace std;

int main()
{
    // Begin main function
    cout << "Hello world!\n";  // output "Hello world!"
    return 0;                 /* indicate successful
                              program completion */
}  // End main function
```

```
• Explain programs to other programmers
• Ignored by compiler
• Syntax:
  // single line comment
  /* multi-line
     comment */
```

Preprocessor directives

```cpp
// include library of standard input and output commands
#include <iostream>
using namespace std;

int main()
{
    // Begin main function
    cout << "Hello world!\n";  // output "Hello world!"
    return 0;                 /* indicate successful
                              program completion */
}  // End main function
```

```
• Lines beginning with #
• Executed before compiling the program
```

main function

```cpp
// include library of standard input and output commands
#include <iostream>
using namespace std;

int main()
{
    // Begin main function
    cout << "Hello world!\n";  // output "Hello world!"
    return 0;                 /* indicate successful
                              program completion */
}  // End main function
```

```
Every C++ program has the function int main()
• main contains the instructions to be executed by
  the program
• The instructions included in the “body” of main are
  placed between curly braces {
```

Statements

```cpp
// include library of standard input and output commands
#include <iostream>
using namespace std;

int main()
{
    // Begin main function
    cout << "Hello world!\n";  // output "Hello world!"
    return 0;                 /* indicate successful
                              program completion */
}  // End main function
```

```
• Instructions to be performed when the program is
  run
• Each statement is completed with a ;
```
Using “white spaces”

```cpp
// include library of standard input and output commands
#include <iostream>
using namespace std;

int main()
{   // Begin main function
    cout << "Hello world!\n";  // output "Hello world!"
    return 0;  /* indicate successful program completion */
} // End main function
```

- “White spaces” are blank lines, space characters, and tabs
- White spaces are ignored by the compiler
- Use indentation to group pieces of code together

Output command

```cpp
cout << "text"; // outputs the specified text to the screen
```

- `cout` is the output stream object
- The text is delimited by double-quotes " "
- Only use simple quotes (") not curly quotes (""")
- `<<` is the “stream insertion operator” directing the text into `cout`

Terminology:

A “character” is any single letter or symbol. E.g.: 'b', '?', '&'
A collection of characters is called a “string.” E.g.: "Hello world", "afe94n", "C++ is fun!"

Output command, part 2

```cpp
cout << "Hello world!\n";
```

- Escape character: backslash \  
- Escape sequence: backslash followed by another character  
  - New line: \n  - Tab: \t

Output command, part 3

```cpp
cout << "Hello world!\n";
```

- We can place multiple stream insertion operators in a sequence.

```cpp
cout << "Hello world" << "!!!";
cout << "How are \nyou today?";
```

```cpp
> ./myProgram
Hello world!
>
> ./myProgram
Hello world!
>