Lecture 1

C++ Basics
Learning Objectives

• Introduction to C++
  – Origins, Object-Oriented Programming, Terms

• Variables, Expressions, and Assignment Statements

• Console Input/Output

• Program Style

• Libraries and Namespaces
Introduction to C++

• C++ Origins
  – Low-level languages
    • Machine, assembly
  – High-level languages
    • C, C++, ADA, COBOL, FORTRAN
  – Object-Oriented-Programming in C++

• C++ Terminology
  – Programs and functions
  – Basic Input/Output (I/O) with cin and cout
A Sample C++ Program

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

int main() {
    int numberOfLanguages;

    cout << "Hello reader.\n"
    << "Welcome to C++.\n";

    cout << "How many programming languages have you used? ";
    cin >> numberOfLanguages;

    if (numberOfLanguages < 1)
        cout << "Read the preface. You may prefer\n"
             << "a more elementary book by the same author.\n";
    else
        cout << "Enjoy the book.\n";

    return 0;
}
```
Display 1.1
A Sample C++ Program (2 of 2)

**Sample Dialogue 1**
Hello reader.
Welcome to C++.
How many programming languages have you used? 0
Read the preface. You may prefer
a more elementary book by the same author.

User types in 0 on the keyboard.

**Sample Dialogue 2**
Hello reader.
Welcome to C++.
How many programming languages have you used? 1
Enjoy the book

User types in 1 on the keyboard.
Constants

• Literal constants

```c++
int myAge=69;
```

• Named constants

**Typed constant**

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

const double PI = 3.14159;
const char NEWLINE = '\'n';

int main ()
{
    double circleRadius=5.0;
    double circle;

circle = 2*PI*circleRadius;
    cout<<circle;
    cout<<NEWLINE;
}
```

**Preprocessor definitions (#define)**

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

#define PI 3.14159
#define NEWLINE 'n'

int main ()
{
    double circleRadius(5.0);
    double circle;

circle = 2*PI*circleRadius;
    cout<<circle;
    cout<<NEWLINE;
}
```
Initializing Variables in Declarations

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

const double PI = 3.14159;
const char NEWLINE = '\n';

int main ()
{
    double circleRadius(5.0);
    double circle;

    circle = 2*PI*circleRadius;
    cout<<circle;
    cout<<NEWLINE;
}
```
Increment and Decrement Operators

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

int main ()
{
    int n = 8;
    int valueProduced = n--;
    cout << valueProduced << "\n";
    cout << n << "\n";
}
```

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

int main ()
{
    int n = 8;
    int valueProduced = --n;
    cout << valueProduced << "\n";
    cout << n << "\n";
}
```
Console Input/Output

- I/O objects `cin`, `cout`, `cerr`
- Defined in the C++ library called `<iostream>`
- Must have these lines (called pre-processor directives) near start of file:
  - `#include <iostream>
    using namespace std;`
  - Tells C++ to use appropriate library so we can use the I/O objects `cin`, `cout`, `cerr`
Console Output

• What can be outputted?
  – Any data can be outputted to display screen
    • Variables
    • Constants
    • Literals
    • Expressions (which can include all of above)
  – cout << numberOfGames << " games played.";
    2 values are outputted:
      "value" of variable numberOfGames,
      literal string " games played."

• Cascading: multiple values in one cout
Separating Lines of Output

• New lines in output
  – Recall: "\n" is escape sequence for the char "newline"

• A second method: object endl

• Examples:

```c++
int main ()
{
    cout << "Hello World!\n";
    cout << "I'm a C++ program";
}

int main ()
{
    cout << "Hello World!" << endl;
    cout << "I'm a C++ program";
}
```

Hello World!
I'm a C++ program
String type

• C++ has a data type of “string” to store sequences of characters

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

int main ()
{
    string fruitOne, fruitTwo, fruitThree;
    fruitOne = "durian";
    fruitTwo = "mango";
    cout<<fruitOne + fruitTwo <<endl;
    cin>>fruitThree;
    cout<<fruitThree;
}
```

– `cin >> str` where `str` is a string only reads up to the first whitespace character
Display 1.5  Using <cin> and <cout> with a string (part 1 of 2)

1 //Program to demonstrate <cin> and <cout> with strings
2 #include <iostream>
3 #include <string> // Needed to access the string class.
4 using namespace std;
5 int main()
6 {
7     string dogName;
8     int actualAge;
9     int humanAge;
10
11     cout << "How many years old is your dog?" << endl;
12     cin >> actualAge;
13     humanAge = actualAge * 7;
14
15     cout << "What is your dog's name?" << endl;
16     cin >> dogName;
17
18     cout << dogName << "'s age is approximately " <<
19         "equivalent to a " << humanAge << " year old human." << endl;
20
21     return 0;
22 }
Input/Output (2 of 2)

Display 1.5 Using `cin` and `cout` with a string (part 2 of 2)

Sample Dialogue 1

```
How many years old is your dog?
5
What is your dog's name?
Rex
Rex's age is approximately equivalent to a 35 year old human.
```

Sample Dialogue 2

```
How many years old is your dog?
10
What is your dog's name?
Mr. Bojangles
Mr.'s age is approximately equivalent to a 70 year old human.
```

"Bojangles" is not read into `dogName` because `cin` stops input at the space.
Formatting Output

• Formatting numeric values for output
  – Values may not display as you’d expect!
    
    cout << "The price is $" << price << endl;

  • If price (declared double) has value 78.5, you might get:
    – The price is $78.500000  or:
    – The price is $78.5

• We must explicitly tell C++ how to output numbers in our programs!
Formatting Numbers

• "Magic Formula" to force decimal sizes:
  
cout.setf(ios::fixed);
cout.setf(ios::showpoint);
cout.precision(2);

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

int main ()
{
    double price=78.5;
    cout.setf(ios::fixed);
    cout.setf(ios::showpoint);
    cout.precision(2);
    cout<<"The price is $"<< price<<endl;
}
```

The price is $78.50
Input Using cin

- cin for input, cout for output

- Differences:
  - ">>>" (extraction operator) points opposite
    - Think of it as "pointing toward where the data goes"
  - Object name "cin" used instead of "cout"
  - No literals allowed for cin
    - Must input "to a variable"

- cin >> num;
  - Waits on-screen for keyboard entry
  - Value entered at keyboard is "assigned" to num
Prompting for Input: cin and cout

• Always "prompt" user for input
  cout << "Enter number of dragons: ";
  cin >> numOfDragons;
  – Note no "\n" in cout. Prompt "waits" on same line for keyboard input as follows:

    Enter number of dragons: _____

• Underscore above denotes where keyboard entry is made

• Every cin should have cout prompt
  – Maximizes user-friendly input/output
Getline

```c++
int main ()
{
    string Name;

    cout<<"Please enter your name"<<endl;
    getline (cin,Name);
    cout<<"Hello, "<<Name<<endl;
}
```

```c++
int main ()
{
    char Name[256];

    cout<<"Please enter your name"<<endl;
    cin.getline (Name,256);
    cout<<"Hello, "<<Name<<endl;
}
```
Error Output

• Output with cerr
  – cerr works same as cout
  – Provides mechanism for distinguishing between regular output and error output

```cpp
int main()
{
    string firstName, lastName;
    int score, age;
    fstream inputStream;

    inputStream.open("player1.txt");
    if (inputStream.fail())
    {
        cerr<<"Error!"<<endl;
    }
}
```

```
int main ()
{
    string firstName, lastName;
    int score, age;
    fstream inputStream;  

    inputStream.open("player1.txt");
    if (inputStream.fail())
    {
        cout<<"Error!"<<endl;
    }
}
```

Error!
Name:
Score: 4202192
Age: 1985305540

Error!
Name:
Score: 1985305540
Age: 1984958818
Program Style

• Bottom-line: Make programs easy to read and modify

• Comments, two methods:
  – // Two slashes indicate entire line is to be ignored
  – /* Delimiters indicates everything between is ignored */
  – Both methods commonly used

• Identifier naming
  – ALL_CAPS for constants
  – lowerToUpper for variables
  – Most important: MEANINGFUL NAMES!
Libraries

• C++ Standard Libraries
• `#include <Library_Name>`
  – Directive to "add" contents of library file to your program
  – Called "preprocessor directive"
    • Executes before compiler, and simply "copies" library file into your program file

• C++ has many libraries
  – Input/output, math, strings, etc.
Namespaces

• Namespaces defined:
  – Collection of name definitions

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

int main ()
{
    cout << "Hello World! ";
    cout << "I'm a C++ program";
}
```

```cpp
#include <iostream>

int main ()
{
    std::cout << "Hello World! ";
    std::cout << "I'm a C++ program";
}
```

```cpp
/* my second program in C++
   with more comments */
#include <iostream>
using std::cout;

int main ()
{
    cout << "Hello World! ";    // prints Hello World!
    cout << "I'm a C++ program";    // prints I'm a C++ program
}
```
Introduction to File Input

• We can use `cin` to read from a file in a manner very similar to reading from the keyboard

• Only an introduction is given here, more details are in chapter 12
  – Just enough so you can read from text files and process larger amounts of data that would be too much work to type in
Opening a Text File

• Add at the top
  ```cpp
  #include <fstream>
  using namespace std;
  ```

• You can then declare an input stream just as you would declare any other variable.
  ```cpp
  fstream inputStream;
  ```

• Next you must connect the inputStream variable to a text file on the disk.
  ```cpp
  inputStream.open("filename.txt");
  ```

• The “filename.txt” is the pathname to a text file or a file in the current directory
Reading from a Text File

• Use

```cpp
inputStream >> var;
```

• The result is the same as using `cin >> var` except the input is coming from the text file and not the keyboard

• When done with the file close it with

```cpp
inputStream.close();
```
File Input Example (1 of 2)

- Consider a text file named player.txt with the following text

```
100510
Gordon Freeman
```

Display 2.10  Sample Text File, `player.txt`, to Store a Player’s High Score and Name
File Input Example (2 of 2)

Display 2.11  Program to Read the Text File in Display 2.10

```cpp
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

int main()
{
    string firstName, lastName;
    int score;
    fstream inputStream;

    inputStream.open("player.txt");

    inputStream >> score;
    inputStream >> firstName >> lastName;

    cout << "Name: " << firstName << " "
        << lastName << endl;
    cout << "Score: " << score << endl;
    inputStream.close();

    return 0;
}
```

Sample Dialogue

Name: Gordon Freeman
Score: 100510
Summary 1

• C++ is case-sensitive
• Use meaningful names
  – For variables and constants
• Variables must be declared before use
  – Should also be initialized
• Use care in numeric manipulation
  – Precision, parentheses, order of operations
• #include C++ libraries as needed
Summary 2

- **Object cout**
  - Used for console output
- **Object cin**
  - Used for console input
- **Object cerr**
  - Used for error messages
- **Use comments to aid understanding of your program**
  - Do not overcomment