Chapter 2
Introduction to C++

- Reading: Chapter 2 (2.4 to 2.10), Chapter 3 (3.1)
- Good Problems to Work: pp. 40 [2.5], pp.47 [2.7, 2.8], pp.53 [2.11, 2.12, 2.15]

Variables

- Named storage location for holding data
- Named piece of memory
- You need to determine what variables you need in your program
- What data do we need to handle?
Variable Definition

```
int number;
```
- Tells the compiler
  - the variable's type (int)
  - the variable's name (number)
- `int` is short for integer
- Variable definitions end with a semicolon
- Every variable must be defined

C++ Assignment Statement

```
number = 5;
```
- `=` is an operator that copies the value from the right into a variable on the left
- The item to the left of the `=` operator must be a variable
- You cannot write `5 = number;`

Variables in a program

```cpp
// This program has a variable called number
#include <iostream>
#include <string>
using namespace std;

int main() // what is the output of this program?
{
   int number;
   number = 5;
   cout << "Number is " << number << endl;
   number = 7;
   cout << "Now number is " << number << endl;
   return EXIT_SUCCESS;
}
```
cin object

- cin is an executable statement
- cin is the standard input object
- The keyboard is the standard input device
- cin is a stream object and works with streams of data
- The executable statement
  ```
  cin >> number;
  ```
  places the value a user types at the keyboard into the variable number?

Input operator (extraction operator): >>

- Standard input (from keyboard): cin
- Whatever the user types in is stored in the variable to the right of the operator (the right operand)
  - All variables must be previously declared
- When reading in the data typed by the user
  - Any spaces before the data item are skipped
  - Reading continues until the user hits return

What is the output?

- Consider the following program:
  ```
  int num1;
  int num2;
  cout << "Enter two numbers: ";
  cin >> num1 >> num2;
  cout << num1 << "  " << num2 << endl;
  ```
- What is output if the user enters: 10 12
  - What is the output if the user enters: 5 10 15
Variable Definition

We now know that:

```c
int number;
```

- **Data type**
- **Identifier**

What is an identifier?!

---

Identifiers (Variables)

- Programmer-defined names that represent some element of a program

- C++ limits on variable names:
  1. Identifiers must begin with a letter or an underscore
  2. Identifiers must consist of letters, numbers and underscore, nothing else
  3. Identifiers cannot be a `keyword`

---

Identifiers (Variables)

- Identifiers are case sensitive
  ```c
  int totalCost;
  int TotalCost;
  ```

- Use meaningful variable names
  ```c
  int width;
  int w;
  ```
Data types

- A data type defines:
  - how the computer interprets data in memory
- C++ has many data types including:
  - Numerical data: int, double, float
  - Textual data: string
  - Character data: char
  - Binary data: bool

Integer (int)

- The main integer data type is int
  - Others are short and long
  - ints are finite (why?)
- An int without a sign (+ or -) is assumed to be positive
- 2,353 is not an int while 2353 is an int
- Operations?

Character (char)

- The char data type is used to store a single character (a letter, a digit, or a special character)
  - ASCII is the internal representation for a char
- Character literals are enclosed in single quotes
- Examples of character literals are: ‘a’, ‘2’, ‘$’
Program

```cpp
#include <iostream>

using namespace std;

int main()
{
    char letter;
    letter = 'A';
    cout << letter << ' ';
    letter = 'B';
    cout << letter << endl;
    return EXIT_SUCCESS;
}
```

string Class

- string is used to store a list of characters
- Need to include the preprocessor directive

```cpp
#include <string>
```
- why?

string Questions

- How do we declare a variable of type string?
- How do we assign a value to the variable?
- How do we output a string literal and a string variable?
- What is the difference between 'A' and "A"?
Floating-Point (**double**)  

- **double, float, long double**  
  - positive and negative  
  - no unsigned float!  

**Scientific Notation**  

- **Examples:**  
  - 1.0, -2.3, -0.3, 12E5, -1E-2, 1.4e+8  
  - 2,353.99 is not a **double**  
  - 2353.99 is a **double**

---

**Examples**

- Remember, the format for declaring variables is:  
  - **data-type** identifier;  

- You can declare variables of the different data types as follows:
  - `int num1;`  
  - `double num2;`  
  - `char letter;`  
  - `string name;`

---

**Boolean (**bool**)**

- Variables of type **bool** can be either **true** or **false**  
  - They cannot be any other value  
  - For coding standards, we precede boolean variables with a `b`

- **Example**
  ```
  bool bValue;
  bValue = true;
  cout << bValue << endl;
  bValue = false;
  cout << bValue << endl;
  ```
Identifier Problem

- Which of the following declarations are invalid and why?
  1. `char Letter1;`
  2. `char lletter;`
  3. `double inches, kms;`
  4. `double inches*num;`
  5. `int joe's;`
  6. `Int cent_per_inch;`
  7. `double two-dimensional;`
  8. `char hello;`
  9. `int return;`
 10. `size int;`