Lecture 2 8/21

Chapter 2: Intro to C++

2.1 The parts of a C++ Program

```
// sample C++ program          <-- comment
#include <iostream>            <-- preprocessor directive
using namespace std;           <-- which namespace to use
int main()                     <-- beginning of function named main
{                              <-- beginning of block for main
   cout << "Hello, there!";    <-- output statement
   return 0;                   <-- send 0 to operating system
}                              <-- end of block for main
```

- Special Characters

Character	Name	Meaning
//	Double Slash	Beginning of a comment
#	Pound sign	Beginning of preprocessor directive
<>	Open/Close brackets	Enclose filename in #include
()	Open/close parenthesis	Used when naming a function
{}	Open/close brace	Encloses a group of statements
" "	Open/close quotation marks	Encloses string of characters
;	Semicolon	End of a programming statement

2.2 The cout object

- Displays output on the screen
- Use stream insertion operator << to send output to cout

```
cout << "Hello!";
```

- Can be used to send more than one item
  - C++ prints these outputs on same line, use endl to get new line

```
cout << "Hello!" << endl << "How are you?";
```

↳ Hello!
How are you?

\n inside string will also create a new line
"Hello!\n";

2.3 The #include Directive

- Inserts contents of another file into the program
- This is a preprocessor directive, not part of C++ language
- # include lines not seen by compiler
- No semicolon at end of #include line

2.4 Variables & Literals

- Variable: A storage location in memory
  - Has a name and a type
  - Must be defined before use → int item;
- Literal: A value that is written into a program's code
  - "hello, there" (string literal)
  - 12 (integer literal)

2.5 Identifiers

- A programmer-defined name for some part of a program (variable, function, etc.)
- You cannot use any of the key words of your coding language as an identifier. These words have reserved meaning
  - There's a lot
- Variable names
  - Should represent the purpose of the variable
- Identifier Rules
  - First character must be an alphabetic character or underscore
  - After, can use alphabetics, numbers, or underscores
  - Upper- & lowercase characters are distinct (don't)
- "Camel-case"
  - Used commonly when naming variables
  - First word begins lowercase, following words start uppercase
    - Ex: firstName, daysWorked

2.6 Integer Data Types

Data Type	Typical Size
Short int	2 bytes
Unsigned short int	2 bytes
int	4 bytes
unsigned int	4 bytes
long int	4 bytes
unsigned long int	4 bytes
long long int	8 bytes
unsigned long long int	8 bytes

- Variables can be declared on seperate or the same line (if same type)

```
int length;
int width;
short int num;
```

or

```
int length, width;
short int num;
```

- Integer literals are stored in memory as ints by default
  - Can be stored in a long memory location with 'L' after int (123L)
  - "LL" for long long location (1234LL)
- Constants beginning w/ '0' are base 8: 075
- Beginning w/ '0x' are base 16: 0x75A

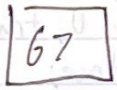
2.7 The char Data Type

- Holds character or very small integer value
- Usually 1 byte of memory
- Numeric value of character from character set stored in memory

CODE:

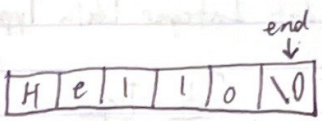
```
char letter;
letter = 'C';
```

MEMORY:



letter

- Character Strings
  - A series of characters
- Stored with null terminator, \0, at end
  - Tells memory where string ends



2.8 The C++ string Class

- Special data type supports working with strings
  - #include <string>
- Not a part of default C++ library

2.9 Floating-Point Data Types

- Float
- Double
- Long Double
- Can hold real numbers: 12.45, -3.8, etc.
- Stored in a form similar to scientific notation
- All floating-point numbers are signed

Type	Key Word	Size
Single Precision	float	4 bytes
Double Precision	double	8 bytes
Long double precision	long double	8 bytes

- Can be represented in decimal or E notation

2.10 The bool Data Type

- Represents True/False values
- Stored as small integers
- false is represented by 0, true by 1:

```
bool allDone = true;
bool finished = false;
```

2.11 Determining the Size of a Data Type

- The sizeof operator gives the size of any data type or variable

```
cout << sizeof(double);
```

2.12 Variable Assignments & Initialization

- = operator assigns value → item = 12;
- Recieving variable must be on left of operator
- Can initialize and assign on same line → int length = 12;
- Auto keyword
  - Auto can be used to let the compiler determine variable type

```
auto amount = 100; ← int
auto amount = 12.0; ← double
auto code = 'D' ← char
```

2.13 Scope

- The scope of a variable is the part of the program in which the variable can be accessed
- A variable cannot be used before it is defined

2.14 Arithmetic Operators

- Used for numeric calculations
- C++ has unary, binary, & ternary operators
  - Unary (1 operand) -5
  - Binary (2 operands) 13 - 7
  - Ternary (3 operands) exp1 ? exp2 : exp3

2.15 Comments

- Used to document parts of the program
- Ignored by compiler
- Used as discriptive tools
  - Great when working in teams

2.16 Named Constants

- Constant variables that cannot be changed during program execution
- Used for representing constant values with descriptive names:

```
const double TAX_RATE = 0.0675;
const int NUM_STATES = 50;
```

- Often named with uppercase letters

2.17 Programming Style

- The visual organization of the source code
- Includes the use of spaces, tabs, and blank lines
- Does not affect the syntax of the program
- Affects the readability of the source code
  - Have good programming style or you'll be a fool in the workplace.