Chapter 2

Wednesday, August 30, 2017

Chapter 2: Introduction to C++

2.1 Parts of a C++ Program

```
// This is a sample C++ program.
include <iostream>
using namespace std;
   int main()
   {
      cout << "Hello there!" << endl;
      system("pause");
      return 0;
   }

```

- Every C++ program needs a main function.
- Braces must be balanced.
- Statements are terminated with semicolons.
- C++ is a case-sensitive language.
- C++ source code is stored in a file with a .cpp extension.
- Comments are ignored by the compiler.

2.2 The cout Object

- Streams output to standard output (i.e. console window)
- Need to use `#include <iostream>`

`<<`	Insertion Stream Operator

- Need to use #include <iostream>

<<	Insertion Stream Operator

```
cout << "Programming ";
cout << "is great fun!";
```

Output:

```
Programming is great fun!
```

endl

- starts a new line of output

```
cout << "Programming " << endl;
cout << "is great fun!";
```

Output:

```
Programming
is great fun!
```

C++ Escape Sequences

\n	Advances the cursor the next line
\t	Prints a tab
\\	Prints a \
\'	Prints a '
\"	Prints a "

```
cout << "These are our top sellers:\n";
cout << "\tComputer games\n\tCoffee\n";
cout << "\tAspirin";
```

Output:

```
These are our top sellers:
     Computer games
     Coffee
     Aspirin
```

```
char ch = '\n';
cout << "WELCOME" << ch << "ALL";
```

Output:

```
WELCOME
ALL
```

2.3 The #include directive

- Preprocessor directive
- Insert the contents of another file into the program
- No ; at the end of #include

2.4 Variables and Literals

Variable - storage location in memory (RAM)

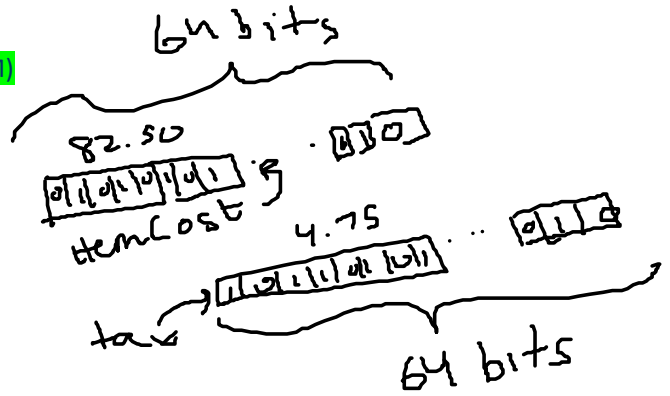
- Name
- Data type
- Value

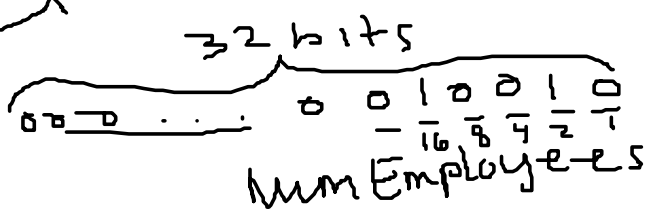
variables

```
double itemCost = 82.50;
double tax;
tax = 4.75;

int numEmployees = 18;
```

literals





Literals - value is written into the code of a program

2.5 Identifiers

Identifier - programmer-defined name for some part of a program

Variable names	`itemsOrdered`
Constants	`TAX_RATE`
Function names	`main`
Class names	`Car`

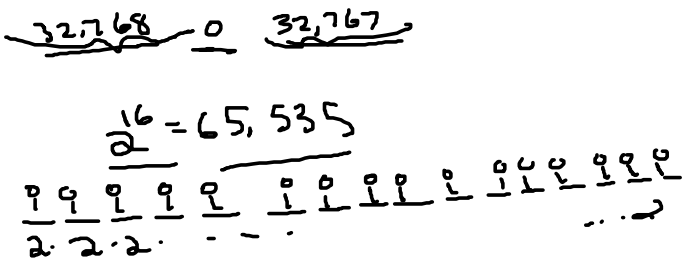
First: `A-Z a-z _`
After first character: `A-Z a-z _ 0-9`

Identifier	Valid?	Reason if Invalid
totalSales	Yes	
total_Sales	Yes	
total.Sales	No	Cannot contain .
4thQtrSales	No	Cannot begin with a digit
totalSale$	No	Cannot contain $

2.6 Integer Data Types

```
int x = 12;
```

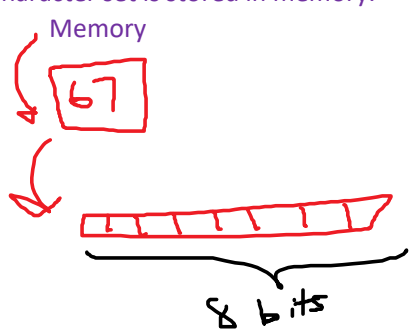
`short int`	2 bytes	-32,768 to 32,767
`unsigned short int`	2 bytes	0 to 65, 535
`int`	4 bytes	-2,147,483,648 to 2,147,483,647
`unsigned int`	4 bytes	0 to 4,294,967,295
`long int`	4 bytes	-2,147,483,648 to 2,147,483,647
`long long int`	8 bytes	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
`unsigned long long int`	8 bytes	0 to 18,446,774,073,709,551,615 1.8 x 10 ^ 19



2.7 The `char` Data Type

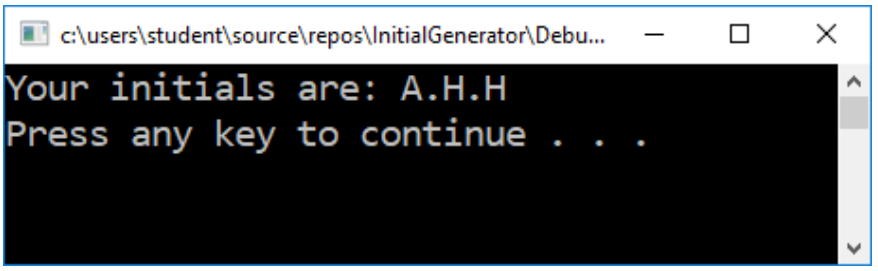
- Used to hold characters
- Usually takes up 1 byte of memory
- Numeric value of character from the character set is stored in memory:

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



```
1  #include <iostream>
2  using namespace std;
3
4  void main() {
5      char firstInitial = 'A';
6      char middleInitial = 'H';
7      char lastInitial = 'H';
8
9      cout << "Your initials are: " << firstInitial;
10     cout << '.' << middleInitial << '.' << lastInitial << endl;
11
12     system("pause");
13     //return 0;
14 }
15
```

Output:



2.8 The C++ String

string - series of characters stored in adjacent memory locations

Ex. "Hello"

H	**e**	**l**	**l**	**o**	**\0**

\0 - null terminator

C++ string class

- **Special data type that supports working with strings**
- **#include <string>**

```
#include <iostream>
#include <string> //Required for string class.
using namespace std;

int main(){
    string movieTitle; //declare a string object
    movieTitle = "The Notebook"; // assign a value to movieTitle
```

```
int main(){
     string movieTitle; //declare a string object
     movieTitle = "The Notebook"; // assign a value to movieTitle
     cout << "My favorite movie is "<< movieTitle << endl;
     system("pause");
     return 0;
}
```

Output:

```
My favorite movie is The Notebook
```

2.9 Floating Point Data Type

- **Used for holding real numbers (decimals)**
  - **12.45**
  - **3.8**
- **The floating-point data types are:**
  - **float**
  - **double**
    - **long double**

Float has 7 digits of accuracy
Double has 15 digits of accury

Data type	Memory Used
Float	4 bytes
Double	8 bytes

Floating-point literals are stored as doubles by default.
x = y + 3.5;
To store a floating-point literal as a float:
x = y + 3.5f;

2.10 The bool Data Type

- **true or false**
- **false is represented by 0, true by 1**

bool started = true;
bool finished = false;

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

int main(){

     bool value1 = true;
     bool value2 = false;
     cout << "value1 is " << value1 << endl;
     cout << "value2 is " << value2 << endl;
```

```
    bool value1 = true;
    bool value2 = false;
    cout << "value1 is " << value1 << endl;
    cout << "value2 is " << value2 << endl;
    value2 = true;
    cout << "value1 is " << value1 << endl;
    cout << "value2 is " << value2 << endl;
    system("pause");
    return 0;
}
```

Output

```
value1 is 1
value2 is 0
value1 is 1
value2 is 1
```

2.11 Determine the Size of a Data Type

```
//sample program that determines the sizes of data types
#include <iostream>
#include <string>
using namespace std;

int main() {
    bool a;
    char b;
    short c;
    int d;
    long long e;
    float f;
    double g;
    string h;


    cout << "A bool is stored in " << sizeof(a) << " bytes.\n";
    cout << "A char is stored in " << sizeof(b) << " bytes.\n";
    cout << "A short is stored in " << sizeof(c) << " bytes.\n";
    cout << "An int is stored in " << sizeof(d) << " bytes.\n";
    cout << "A long long is stored in " << sizeof(e) << " bytes.\n";
    cout << "A float is stored in " << sizeof(f) << " bytes.\n";
    cout << "A double is stored in " << sizeof(g) << " bytes.\n";
    cout << "A string is stored in " << sizeof(h) << " bytes.\n";

    system("pause");
    return 0;

}
```

2.12 Variable Assignments and Initialization

Term	Definition
Declare	Reserve a space in memory
Assign	Change the contents of the variable
Initialize	Assigning a variable value for the first time

int a; // <=====declaration
a = 10; // <====assigned (**initialized**)
a = 20; // <====assigned (but not **initialized**)
int b = 10; <===declared and assignment (**initialized**)

auto keyword
auto interestRate = 12.0;
auto interestRate = 12.0f;
auto stockCode = 'D';
auto myVariable = true;
auto myVariable = 0;

2.13 Scope
Scope of a variable - the part of the program in which the variable can be accessed
Scope of a variable ranges from line in which it is declared to } that contains the declaration

```
1. //same program with a scope error
2. #include <iostream>
3. using namespace std;
4.  int main(){
5.    cout << "Value is " << value;
6.    int value = 99;
7.    system("pause");
8.    return 0;
9. }
```

value is in scope in lines 6-9

```
1. //same program with a scope error
2. #include <iostream>
3. using namespace std;
4.  int main(){
5.    int value = 99;
6.    cout << "Value is " << value;
7.    system("pause");
8.    return 0;
9. }
```

value is in scope in lines 5-9

```
8.   return 0;
9. }
```

2.14 Arithmetic Operators

+	Addition	total = price + tax;
-	Subtraction	total = price - discount;
*	Multiplication	tax = price * rate;
/	Division	average = sum / n;
%	Modulus	remainder = x % y;

```
int x = 27;
int y = 10;
int z = x / y;          //stores 2 to z.

double x = 27.0;
double y = 10.0;
double z = x / y;    //stores 2.7 to z.

int x = 27;
double y = 10.0;
double z = x / y;     //stores 2.7 to z.

int x = 27;
double y = 10.0
int z = x / y;          //stores 2 to z.
```

2.7

2

```
int x = 27;
int y = 10;
double z = x / y;
```

2

2.0

2 R 7

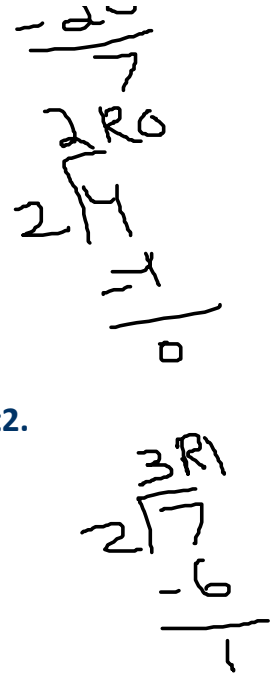
```
int x = 27;
int y = 10;
int z = x % y;        //stores 7 to z.
```

10) 27
-20
7

```
int z = x % y;        //stores 7 to z.

int result = 4 % 2;  //store 0 to result.

int result2 = 7 % 2; //store 1 to result2.
```




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

int main(){
        int a = 17;
        int b = 5;
        int result = a / b;
        int remainder = a % b;
        cout << result << endl;
        cout << remainder << endl;

        double x = 17.0;
        double y = 5.0;
        double result2 = x / y;
        cout << result2 << endl;

        system("pause");
        return 0;
}
```

A

3
2
3.4

Output

3
2
3.4

2.16 Named Constants

- Declared using the keyword `const`

- **Value that cannot be changed after first assigned.**

```
//Named constants
#include <iostream>
using namespace std;

int main(){
  const int NUM_STATES = 50;
  const double PI = 3.14159265;
  cout << "The value of pi is " << PI;
  cout << "There are " << NUM_STATES << " states" << endl;
  system("pause");
  return 0;
}
```