

Declaration Statements

September 3, 2010

Today

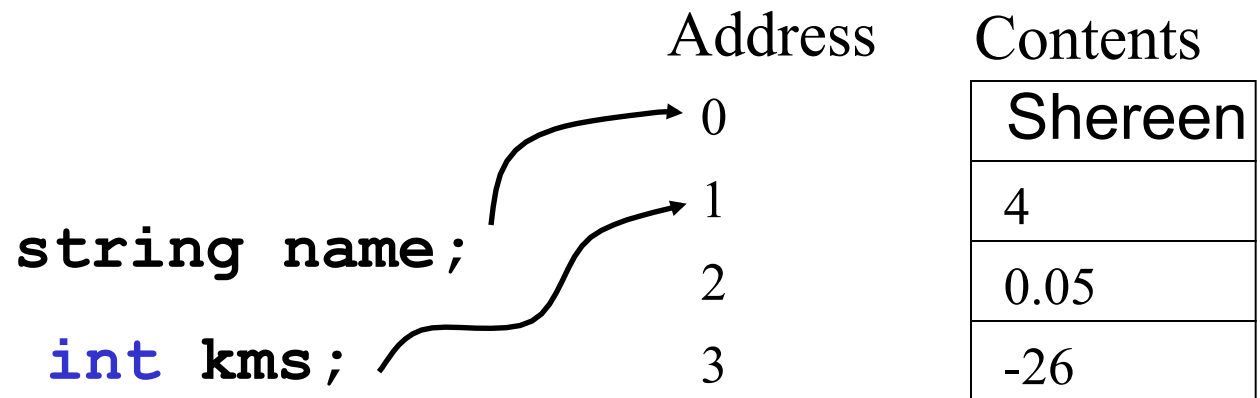
- Last time, we covered the basic components of a C++ program and the cout Object
- What are the main components of every C++ program?
- Today we will
 - learn about variables and data types

Declaration Statements

Variables

Variables


- Named storage location for holding data
 - named piece of memory
- You need to determine what variables you need
 - 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

Assignment

- `number = 5;`
- = is an operator that copies the value on its right into the variable on its left
- The item to the left of the = operator must be a variable
- Let's look at program 2-7 on p. 38, also on the next slide with some modifications

Variables

```
1 // This program has a variable
2 #include <iostream>
3 #include <string>
4 using namespace std;
5
6 int main() // what is the output of this program?
7 {
8     int number;
9
10    number = 5;
11    cout << "The value of number is " << number << endl;
12
13    number = 7;
14    cout << "Now the value of number is " << number << endl;
15
16    return 0;
17 }
```

Input

- 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)
 - That variable must have already been declared
- When reading in the data typed by the user
 - Any spaces before the data item are skipped
 - Continues to read until the user hits return

What is the Output?

- Examples:

```
int num1;
```

```
int num2;
```

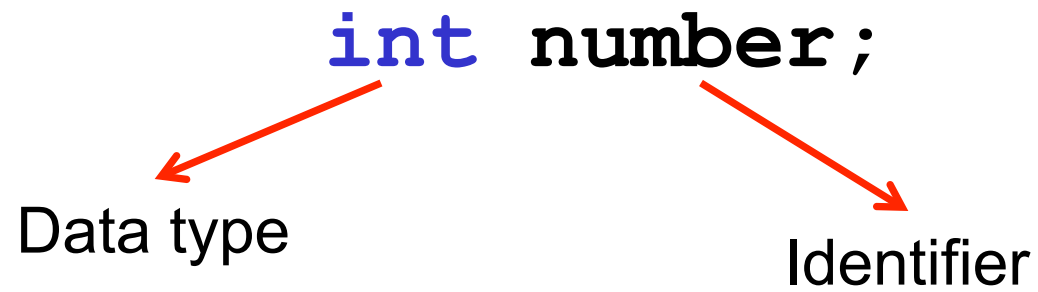
```
cout << "Enter two numbers: ";
```

```
cin >> num1 >> num2;
```

```
cout << num1 << endl;
```

```
cout << num2 << endl;
```

Variable Definition



Identifiers

- 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*

page 41

Identifiers

- Identifiers are case sensitive

```
int totalCost;
```

```
int TotalCost;
```

- Use meaningful variable names

```
width
```

```
W
```

Identifiers

- Which of the following declarations are invalid and why?
 - a. `char Letter1;`
 - b. `char 1letter;`
 - c. `double inches, kms;`
 - d. `double inches*num;`
 - e. `int joe's;`
 - f. `Int cent_per_inch;`
 - g. `double two-dimensional;`
 - h. `char hello;`
 - i. `int return;`
 - j. `size int;`

Data types

- A **data type** defines:
 - how the computer **interprets** data in memory

Integers

- 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`, 2353 is an `int`
- Operations?

char

- The **char** data type is used to store single characters (letters, digits, special characters)
 - ASCII
- Character literals are enclosed in **single** quotes
- Examples of character literals are: `'A'` , `'a'` , `'*'` , `'2'` , `'$'`

Program

```
// page 48, program 2-13
#include <iostream>

using namespace std;

int main()
{
    char letter;

    letter = 'A';
    cout << letter << endl;
    letter = 'B';
    cout << letter << endl;
    return 0;
}
```

string Class

- `string` is used to store a list of characters
- Need to include the preprocessor directive
 - `#include <string>`
 - why?

string Questions

- Q How do we declare a variable of type string?
- Q How do we assign a value to the variable?
- Q How do we output a string literal and a string variable?

Floating-Point Data Types

- **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;`

bool Data Type

- **bool**: boolean
- Variables of type **bool** can be either **true** or **false**
 - They cannot be any other value

- Example

```
bool bValue;  
bValue = true;  
cout << bValue << endl;  
bValue = false;  
cout << bValue << endl;
```

Summary

- In today's lecture we covered
 - How data that is used by a program can be declared and stored
- We have covered sections 2.4-2.9 and 2.11 of your textbook