

# Define Variable Types in C++

**Unit:** Programming

**Problem Area:** Use C++ in Game Development

**Lesson:** Define Variable Types in C++

■ **Student Learning Objectives.** Instruction in this lesson should result in students achieving the following objectives:

- 1 List variable types used in C++.**
- 2 List the guidelines for naming variables in C++.**
- 3 Write variable declaration statements in C++.**

■ **Resources.** The following resources may be useful in teaching this lesson:

“C++ Tutorials,” *Programming Tutorials.com*. Accessed April 25, 2009. <<http://www.programmingtutorials.com/cplusplus.aspx>>.

“Getting Started with C/C++,” *Cprogramming.com*. Accessed May 11, 2009. <<http://www.cprogramming.com/begin.html>>.

Laramee, Francois, and Erik Yuzwa. *Learn C++ by Making Games*. Charles River Media, 2007.

Soulie, Juan. “Variables. Data Types,” *Cplusplus.com*. Accessed May 11, 2009. <<http://www.cplusplus.com/doc/tutorial/variables/>>.



## ■ **Equipment, Tools, Supplies, and Facilities.**

- ✓ Overhead or PowerPoint projector
- ✓ Visual(s) from accompanying master(s)
- ✓ Copies of sample test, lab sheet(s), and/or other items designed for duplication
- ✓ Materials listed on duplicated items
- ✓ Computers with a C++ programming environment installed (e.g., Visual Studio or Bloodshed)
- ✓ Printers and Internet access
- ✓ Classroom resource and reference materials
- ✓ Whiteboard or flipchart

## ■ **Key Terms.** The following terms are presented in this lesson (shown in bold italics):

- ▶ bool
- ▶ char
- ▶ declaration statement
- ▶ double
- ▶ float
- ▶ global variables
- ▶ identifier
- ▶ int
- ▶ keyword
- ▶ local variables
- ▶ short int
- ▶ string
- ▶ unsigned int
- ▶ unsigned short int
- ▶ variable

## ■ **Interest Approach.** Use an interest approach that will prepare the students for the lesson. Teachers often develop approaches for their unique class and student situation. A possible approach is included here.

*Have students make a list of all of the things they use on a daily basis at school (e.g., a locker, a backpack, and writing utensils).*

*Next, have students write the specific location in which to find the items, along with a nickname for each location. For instance, if the student keeps writing utensils in the left backpack pocket, he or she might write “pencil, left pocket of backpack, nickname = thing-a-ma-bob.”*

When the lists are completed, have the students work in pairs. Taking turns, have each student prompt his or her partner to retrieve (or guess) one of the items on the list by giving only a description of the item and its location nickname. Student pairs are not allowed to use the name of the item and may only use the location nickname, not the specific location description.

Following this activity, have a class discussion about the frustrations and challenges the students encountered. Emphasize the importance of meaningful names.

## SUMMARY OF CONTENT AND TEACHING STRATEGIES

**Objective 1:** List variable types used in C++.

**Anticipated Problem:** What variable types are used in C++?

- I. Variable types—In programming, a **variable** is a place in memory used to store data; it can be visualized as a bucket. Just as a child can store sand, water, or rocks in a bucket, a programmer can store data and then change the data that has been stored in a variable. Unlike a child’s bucket, a variable data type is declared and only the data type declared can be stored in that variable. Therefore, the data stored in a variable can change, but the type of data must be the same type declared when the variable is created.
  - A. An **int** is a variable type used to store integers (whole numbers) that range in value from negative 2,147,483,648 to positive 2,147,483,647.
  - B. A **short int** is a variable type used to store integers that range in value from negative 32,768 to positive 32,767. A short int uses less storage space in memory. As a result, if a programmer knows that values stored will always be within the range of a short int, a short int variable should be used to save system resources.
  - C. If a programmer needs to store only positive integer numbers larger than 2,147,483,647, an unsigned int may work. An **unsigned int** is a variable type that stores positive integers from zero to 4,294,967,295. It doubles the range of numbers allowed by an int on the positive side of zero and does not allow numbers on the negative side of zero to be stored.
  - D. An **unsigned short int** is a variable type used to store integers that range in value from zero to 65,535. It doubles the range of numbers allowed by a short int on the positive side of zero and does not allow numbers on the negative side of zero to be stored.

- E. A **float** is a variable type used to store large numbers that require decimal point precision. A float is precise up to seven significant digits. The maximum size of a float variable is  $3.4 \times 10^{38}$ .
- F. A **double** is a variable type used to store large numbers that require decimal point precision. A double is precise up to 15 significant digits. The maximum size of a double variable is  $1.798 \times 10^{308}$ .
- G. A **char** is a variable type used to store any one of the 256 character values. Technically, a char variable can store more than one character, but those characters cannot be accessed individually unless they are stored as a string. In most cases, it is best to limit data to one character when using the char variable type.
- H. A **string** is a variable type used to store a series of printable characters.
- I. A **bool** is a variable type used to store a true or false value. The number zero is stored in a bool variable to represent false. The number one is stored to represent a true value.

Use VM–A to reinforce the different variable types that can be used in a C++ program. Then assign LS–A.

**Objective 2:** List the guidelines for naming variables in C++.

**Anticipated Problem:** What are the guidelines for naming variables in C++?

- II. C++ identifiers—Every variable in C++ must be given a name or an identifier. An **identifier** is a name given to a variable in C++. The identifier name is used to store or retrieve data stored in the memory location to which it refers.
  - A. Rules for naming identifiers
    1. The name must begin with a letter of the alphabet or an underscore. It cannot begin with a number.
    2. The name may only contain numbers, letters, and underscores.
    3. The name may not be a C++ **keyword** (a reserved word that has special meaning in C++).
  - B. Guidelines for naming identifiers
    1. It is necessary to choose descriptive names to ensure that the use of the identifier is clear. For example, the following names clearly identify the data that will be stored:
      - a. player1\_score
      - b. player2\_score

2. Consistent multi-word identifier names must be used. If a variable name has two words, one of two standards should be used. Then it is necessary to stick with it.
  - a. Standard one uses an underscore between words. For instance:  
player1\_score.
  - b. Standard two uses a lowercase letter for the first word and upper case letters for each additional word or words. For example: player1Score.
3. It is best to follow traditions set by programmers.
  - a. Identifier names should begin with a lowercase letter.
  - b. An identifier should not begin with an underscore.
4. Identifier name lengths should be kept at 15 characters or less. Identifier names can be between 1 and 255 characters. Lengths of 15 characters or less improve the portability of code to other platforms.

*Display VM–B to stimulate a discussion. Then assign LS–B to have students practice naming identifiers.*

**Objective 3:** Write variable declaration statements in C++.

**Anticipated Problem:** How are variable declaration statements written in C++?

### III. Declaring variables

- A. A **declaration statement** is used to create a variable and to assign a data type and identifier name to the variable. Components of a variable declaration statement are:
  1. Variable type
  2. Identifier
  3. Initialization—Initializing a variable is optional, but it is highly recommended. When the computer assigns a place in memory to a variable, there is sometimes leftover data from a previous program or process (garbage) sitting in that storage location. If a variable is not initialized, it takes on the value of the garbage in the location assigned to it. This can result in unexpected output or nasty bugs in the program. It is good practice to initialize a variable to the value of zero or space.
  4. Semicolon—A semicolon signals the end of the declaration statement in C++ just as a period signals the end of a sentence in written English.
- B. Statement locations
  1. **Global variables** are variables declared outside of any block or function in the C++ program.
  2. **Local variables** are variables declared within a C++ function. Once program execution has left the function, the variable is destroyed and its identifier is forgotten by the program.

Use VM–C to reinforce the concept of declaring variables using declaration statements. Use VM–D to emphasize the importance of initialization. Assign LS–C.

- **Review/Summary.** Use the student learning objectives to summarize the lesson. Have students explain the content associated with each objective. Student responses can be used in determining which objectives need to be reviewed or taught from a different angle.
- **Application.** Use the included visual masters and lab sheets to apply the information presented in the lesson.
- **Evaluation.** Evaluation should focus on student achievement of the objectives for the lesson. Various techniques can be used, such as student performance on the application activities. A sample written test is provided.

## ■ **Answers to Sample Test:**

### **Part One: Matching**

1. f
2. c
3. e
4. a
5. b
6. d

### **Part Two: Short Answer**

1. Global variables are declared outside of function blocks. Therefore, they are available throughout the entire program. Once execution of the program moves out of the function block in which the local variables are declared, local variable identifier names are forgotten by the program.
2. Using identifier name lengths of 15 characters or less improves the portability of the code to other computer platforms.

### **Part Three: Completion**

1. declaration statement
2. bool
3. string
4. initialize
5. keywords
6. memory

# Define Variable Types in C++

## ► Part One: Matching

**Instructions:** Match the term with the correct definition.

- |              |                 |
|--------------|-----------------|
| a. short int | d. float        |
| b. variable  | e. char         |
| c. double    | f. unsigned int |

- \_\_\_\_ 1. A variable type that does not allow for negative numbers
- \_\_\_\_ 2. A variable type used to store large numbers that require decimal point precision; precise up to 15 significant digits
- \_\_\_\_ 3. A variable type used to store any one of the 256 character values
- \_\_\_\_ 4. A variable type used to store integers that range in value from negative 32,768 to positive 32,767
- \_\_\_\_ 5. A place in memory used to store data
- \_\_\_\_ 6. A variable type used to store large numbers that require decimal point precision; precise up to seven significant digits

## ► Part Two: Short Answer

**Instructions:** Answer the following.

- 1. Compare and contrast global and local variables.



2. Why is it a good idea to use identifier name lengths of 15 characters or less?

► **Part Three: Completion**

**Instructions:** Provide the word or words to complete the following statements.

1. A(n) \_\_\_\_\_ is used to declare a variable, its identifier, and its data type.
2. A variable data type used to store only a true or a false value is known as a(n) \_\_\_\_\_.
3. When more than one character needs to be stored in a continual series, it is best to declare a(n) \_\_\_\_\_ type variable.
4. It is considered best practice to \_\_\_\_\_ a variable upon declaration so any leftover garbage in memory does not cause errors in the program.
5. Reserved words in C++ that should not be used as identifier names are \_\_\_\_\_.
6. The computer's operating system allocates space in \_\_\_\_\_ for use by a variable upon declaration of that variable.

# VARIABLES IN C++



**Data Type:**  
**int**

Whole numbers

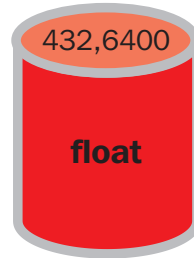
-2,147,483,648 to  
+2,147,483,647



**Data Type:**  
**short int**

Whole numbers

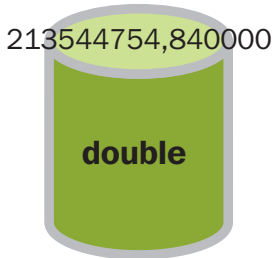
-32,768 to  
+32,767



**Data Type:**  
**float**

Decimal numbers

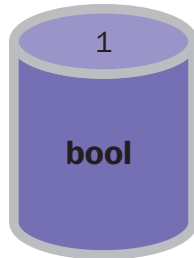
Precise up to 7 significant digits



**Data Type:**  
**double**

Decimal numbers

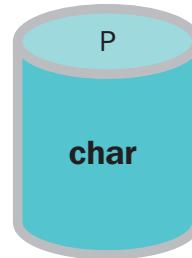
Precise up to 15 significant digits



**Data Type:**  
**bool**

True or False

1 = true (on)



**Data Type:**  
**char**

Typically used for one of  
256 single characters



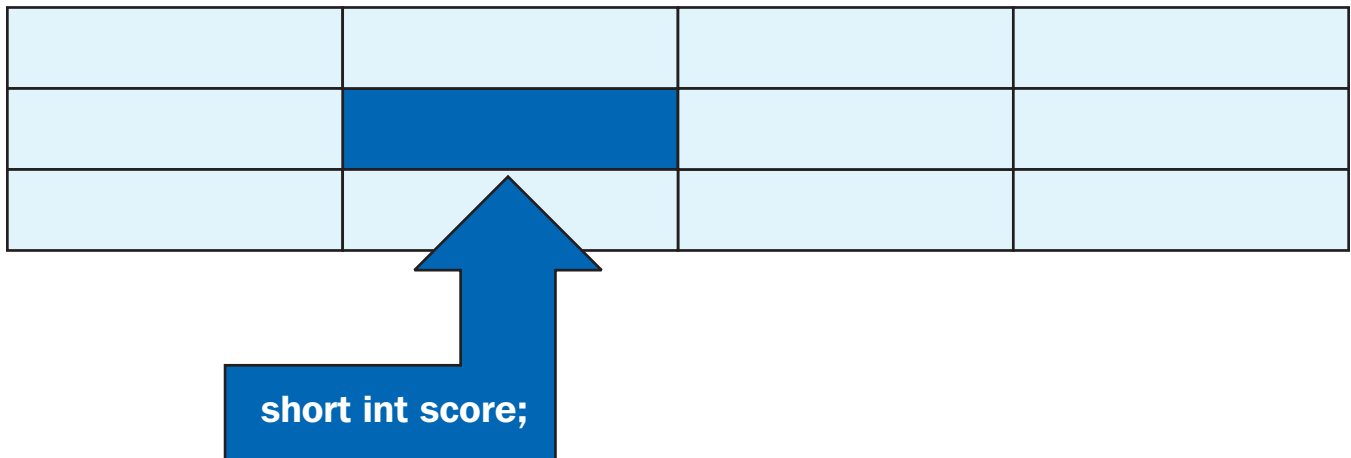
**Data Type:**  
**string**

Series of printable  
characters

# IDENTIFIERS, VARIABLES, AND MEMORY

A variable is a location in memory used to store data. Identifiers are used to name that place in the computer's memory.

## Computer Memory (RAM)



This program statement identifies a location in memory as “score.” Because the type is declared as an int, the data stored there must have a value between  $-32,768$  and  $32,767$ . The name “score” becomes the identifier for the variable location that will store the data representing the score of the game. The computer's operating system takes care of allocating a specific location in memory to the variable.

# SAMPLE DECLARATION STATEMENTS

---

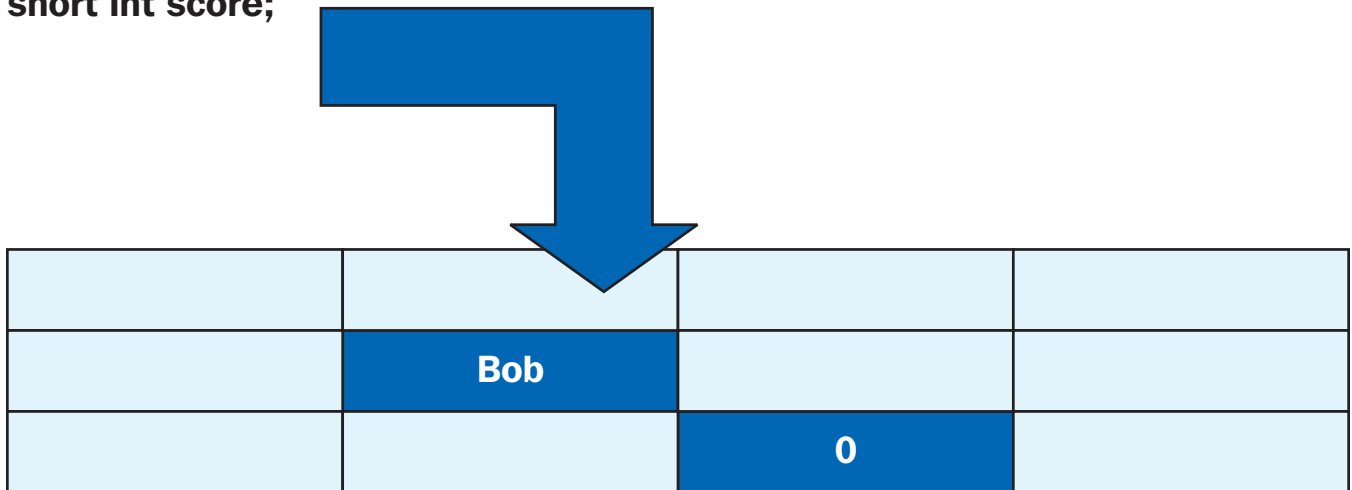
- ◆ `int score;`
  - Identifies a variable as “score” with a data type of integer
- ◆ `float total_pay;`
  - Identifies a variable as “total\_pay” with a data type of float
- ◆ `char initial;`
  - Identifies a variable as “initial” with a data type of character
- ◆ `bool go_left;`
  - Identifies a variable as “go\_left” with a data type of Boolean
- ◆ `short int health=5;`
  - Identifies an integer variable as “health” with an initial value of 5
- ◆ `String player1=“Hulk”;`
  - Identifies a string variable as “player1” with an initial value of “Hulk”

NOTE: It is considered best practice to initialize variables upon declaration.

# INITIALIZING DATA IN DECLARATION STATEMENTS

Memory has leftover data that is not overwritten, as the declaration statement does not initialize the variable location in memory. This can cause unexpected results, errors, and bugs.

**short int score;**



**short int score = 0**

The variable is initialized to a value of zero (0) that writes over any data that may have been leftover by a previous process.

# Determine Variable Types for Data Used in C++

---

## Purpose

The purpose of this activity is to reinforce the variable types available for use in C++.

## Objective

Determine the appropriate variable types for specific data storage requirements.

## Materials

- ◆ writing utensil
- ◆ eraser or Wite-Out®

## Procedure

1. Write the name of the appropriate variable type that should be used to store the data given. The first answer is given for you.

- \_\_\_\_\_ int a. 3298
- \_\_\_\_\_ b. z
- \_\_\_\_\_ c. - 31,904
- \_\_\_\_\_ d. 3.1475
- \_\_\_\_\_ e. true (1)
- \_\_\_\_\_ f. 69,200
- \_\_\_\_\_ g. - 41,984
- \_\_\_\_\_ h. Bob Jones
- \_\_\_\_\_ i. 42.743
- \_\_\_\_\_ j. life is good

2. Participate in a class discussion of the appropriate variable type for each item.
3. Turn in your lab sheet to your instructor.

# Determine Variable Types for Data Used in C++

---

- a. short int
- b. char
- c. short int
- d. float
- e. bool
- f. int
- g. int
- h. string
- i. float
- j. string

# Naming Identifiers

## Purpose

The purpose of this activity is to reinforce the rules and guidelines for identifying variables.

## Objective

Use naming rules and guidelines to write accurate and appropriate identifier names.

## Materials

- ◆ writing utensil
- ◆ notes, if permitted by instructor

## Procedure

1. Create an identifier name for each variable description given below. The first one is shown.

- \_\_\_gross pay\_\_\_ a. Identifier for a variable used to store gross pay that is calculated by multiplying the number of hours worked by the hourly pay.
- \_\_\_\_\_ b. Identifier for a variable used to store an individual's middle initial.
- \_\_\_\_\_ c. Identifier for a variable used to store the health of a game player when there are four players in the game.
- \_\_\_\_\_ d. Identifier for a variable used to accumulate a score for player three when there are four players in the game.
- \_\_\_\_\_ e. Identifier for a variable used to accumulate the computer's score when the program is for a game where the computer plays against a single player.



- \_\_\_\_\_ f. A computer game allows a player to choose his or her own player name. Name an identifier that will be used to store this data.
- \_\_\_\_\_ g. Identifier used to store and accumulate the number of swords gathered by the warrior prince.
- \_\_\_\_\_ h. Identifier used to keep track of how many ducks the hunter has shot down in a hunting game.
- \_\_\_\_\_ i. Identifier used to keep track of how many deer the hunter has shot and killed in a hunting game.
- \_\_\_\_\_ j. Identifier used to keep track of how many bass over 17" have been caught in a fishing game that keeps track of three categories of bass:
  - Category 1: Under 12"
  - Category 2: Between 11" and 17"
  - Category 3: Over 16" (HINT: Think through this logic! There is no overlap in the sizes here!)

2. Participate in a class discussion of the identifier for each item.
3. Turn in your lab sheet to your instructor.

## **Naming Identifiers**

---

The answers below are examples of many possibilities. A name could be considered accurate and appropriate as long as the name follows the naming rules and the standard guidelines.

- a. grossPay, gross\_pay
- b. middleInit, middle\_init, middle\_initial, initial
- c. p2\_health, player2\_health, p2Health
- d. p3\_score, player3\_score, p3Score
- e. computerScore, computer\_score, comp\_score
- f. playerName, player\_name, name
- g. swords, swords\_accum, warrior\_swords
- h. ducks, ducks\_shot, ducks\_accum, ducks\_hunter
- i. deer, deer\_shot, deer\_accum, deer\_hunter
- j. bass\_over16, bass17, large\_bass, bassLarge

# Declaration Statements

## Purpose

The purpose of this activity is to provide practical experience in writing C++ variable declaration statements.

## Objective

Write C++ declaration statements for specific programming problems.

## Materials

- ◆ writing utensil
- ◆ paper

## Procedure

Read each of the following programming scenarios, and write the declaration statements for all variables required. The solution for the Programming Scenario is shown.

## Programming Scenarios

- A. In this game, every time the player eats a heart, 10 points is awarded to that player. A diamond is worth 20 points to the player. Each star is worth 50 points. The player begins with three lives. The game ends when the player loses all three lives. When the game ends, a summary screen will appear listing how many hearts, diamonds, and stars were eaten as well as the total score for the player. For example, the declaration statements would be:

### Solution:

```
short int hearts=0; (one is added each time the player eats a heart)
```

```
short int diamonds=0; (one is added each time the player eats a diamond)
```



short int stars=0; (one is added each time the player eats a star)

short int lives=3; (one is deleted each time the player loses a life)

int total\_score=0; (accumulates the score as hearts, diamonds, and stars are eaten)

- B. In this single-player game, the player shoots down enemy spaceships. Each micro ship is worth 10 points. Each master ship is worth 25 points. If a spaceship shoots and hits the player with a laser, the player loses a life. The player begins the game with five lives. If the player reaches 10,000 points, the game is over and the player wins. If the player loses all five lives before reaching 10,000 points, the aliens win. The player earns an additional life for every 2,500 points earned.

**Solution:**

- C. In this game, the player is asked to provide a name for his or her character in the game. The program will refer to the player by name throughout game play. The game is blackjack, and it is computer (dealer) against one player. Each time a card is dealt, the value of each card dealt along with the accumulated total for the current hand is displayed on the screen next to the player's name. The player can choose to be dealt another card or to hold. The player cannot be dealt more than five cards. The dealer's cards and totals are also displayed.

**Solution:**

2. Participate in a class discussion of declaration statements for each programming scenario.
3. Turn in your lab sheet to your instructor.

# Declaration Statements

---

The identifier names used in the declarations below are examples of many possible names. A name could be considered accurate and appropriate as long as the name follows the naming rules and the standard guidelines. Consistency in naming is important. However, the declarations listed represent the number of variable declaration statements needed to meet the program requirements.

## Programming Scenario B:

```
short int value_micro_ship=10;
short int value_macro_ship=25;
short int player_lives=5;
short int score_to_win=10000;
short int extra_life1=2500;
short int extra_life2=5000;
short int extra_life3=7500;
```

## Programming Scenario C:

Scenario C could easily be accomplished using an array. See lessons on single and multi-dimensional arrays.