GLOBAL EDITION

Starting Out with C++

From Control Structures through Objects

NINTH EDITION

Tony Gaddis

Pearson

Digital Resources for Students

Your new textbook provides 12-month access to digital resources that may include VideoNotes (step-by-step video tutorials on programming concepts), source code, web chapters, quizzes, and more. Refer to the preface in the textbook for a detailed list of resources.

Follow the instructions below to register for the Companion Website for Tony Gaddis' *Starting Out with C++ from Control Structures through Objects*, Ninth Edition, Global Edition.

1. Go to www.pearsonglobaleditions.com/gaddis
2. Enter the title of your textbook or browse by author name.
3. Click Companion Website.
4. Click Register and follow the on-screen instructions to create a login name and password.

ISSGDS-SPADA-BAUTH-SKEAN-FIERI-BALES

Use the login name and password you created during registration to start using the online resources that accompany your textbook.

IMPORTANT:

This prepaid subscription does not include access to MyProgrammingLab, which is available at www.myprogramminglab.com for purchase.

This access code can only be used once. This subscription is valid for 12 months upon activation and is not transferable.

For technical support go to https://support.pearson.com/getsupport

Program Output

```
Testing an adult membership...
Calling the showFees function with arguments 40 and 10.
The total charges are $400

Testing a senior citizen membership...
Calling the showFees function with arguments 30 and 10.
The total charges are $300

Testing a child membership...
Calling the showFees function with arguments 20 and 10.
The total charges are $200
```

As shown in Program 6-30, a driver can be used to thoroughly test a function. It can repeatedly call the function with different test values as arguments. When the function performs as desired, it can be placed into the actual program it will be part of.

Case Study: See High Adventure Travel Agency Part 1 Case Study on the Computer Science Portal at www.pearsonglobaleditions.com/gaddis.

Review Questions and Exercises

Short Answer

1. Why do local variables lose their values between calls to the function in which they are defined?
2. What is the difference between an argument and a parameter variable?
3. Where do you define parameter variables?
4. If you are writing a function that accepts an argument and you want to make sure the function cannot change the value of the argument, what do you do?
5. When a function accepts multiple arguments, does it matter in what order the arguments are passed?
6. How do you return a value from a function?
7. What is the advantage of breaking your application's code into several small procedures?
8. How would a `static` local variable be useful?
9. Give an example where passing an argument by reference would be useful.

Fill-in-the-Blank

10. In the __________ approach, a large problem is broken up into several smaller problems that can be easily solved.
11. The data type of the variable returned from a function is called __________.
12. Either a function's __________ or its __________ must precede all calls to the function.
13. The __________ function is automatically invoked when execution of a program starts.

14. Special variables that hold copies of function arguments are called _________.
15. When only a copy of an argument is passed to a function, it is said to be passed by _________.
16. A(n) _________ eliminates the need to place a function definition before all calls to the function.
17. A(n) _________ variable is defined inside a function and is not accessible outside the function.
18. _________ variables are defined outside all functions and are accessible to any function within their scope.
19. _________ variables provide an easy way to share large amounts of data among all the functions in a program.
20. When the _________ statement is encountered within a function, the control goes back to the statement that called the function.
21. If a function has a local variable with the same name as a global variable, only the _________ variable can be seen by the function.
22. _________ local variables retain their value between function calls.
23. A _________ is a named constant available to every function in a program.
24. _________ arguments are passed to parameters automatically if no argument is provided in the function call.
25. When a function uses a mixture of parameters with and without default arguments, the parameters with default arguments must be defined _________.
26. The value of a default argument must be a(n) _________.
27. When used as parameters, _________ variables allow a function to access the parameter's original argument.
28. Reference variables are defined like regular variables, except there is a(n) _________ in front of the name.
29. Only _________ can be passed as a reference to a function.
30. Functions that have the same name but different parameter lists are called _________
31. A _________ program is used to test whether a function is working properly.

Algorithm Workbench

32. Examine the following function header, and then write two different examples to call the function:

```
double absolute ( double number );
```

33. The following statement calls a function named average. The average function returns the average of all the arguments passed to the function. Write the function.

```
avg = average ( num1, num2, num3 );
```

34. A program contains the following function:

```
double velocity(double initial, double acceleration, double distance)
{
   return initial * initial + 2 * acceleration * distance;
}
```

Write a statement that passes the values 10.0, 50.5, and 100 to this function and assigns its return value to the variable `final_velocity`.

35. Write a function `isValid` that accepts an argument, grade. The function returns true if grade is within the range A through F; otherwise it returns false.

36. A program contains the following function:

```
void display(int arg1, double arg2, char arg3)
{
   cout  << "Here are the values: "
         << arg1 << " " << arg2 << " "
         << arg3 << endl;
}
```

Write a statement that calls the procedure and passes the following variables to it:

```
int age;
double income;
char initial;
```

37. Write a function named `getNumber` that uses a reference parameter variable to accept an integer argument. The function should prompt the user to enter a number in the range of 1 through 100. The input should be validated and stored in the parameter variable.

True or False

38. T F Functions should be given names that reflect their purpose.

39. T F Function headers are terminated with a semicolon.

40. T F A function may have more than one return statement.

41. T F If other functions are defined before `main`, the program still starts executing at function `main`.

42. T F When a function terminates, it always branches back to `main`, regardless of where it was called from.

43. T F Arguments are passed to the function parameters in the order they appear in the function call.

44. T F The scope of a parameter is limited to the function that uses it.

45. T F Changes to a function parameter always affect the original argument as well.

46. T F In a function prototype, the names of the parameter variables may be left out.

47. T F The lifetime of a local variable is throughout the execution of the program.

48. T F A local variable cannot have the same name as a global constant.

49. T F Static local variables are not destroyed when a function returns.

50. T F All static local variables are initialized to −1 by default.

51. T F Initialization of static local variables only happens once, regardless of how many times the function in which they are defined is called.

52. T F When a function with default arguments is called and an argument is left out, all arguments that come after it must be left out as well.

53. T F It is not possible for a function to have some parameters with default arguments and some without.

54. T F The `exit` function can only be called from `main`.

55. T F Stubs and drivers are tools that are used for testing and upgrading long programs that generally do not have any functions or sub-procedures in them.

Find the Errors

Each of the following functions has errors. Locate as many errors as you can.

56.
```
int product ( int num1; int num2; int num3 )
{
   return num1 * num2 * num3;
}
```

57.
```
double athird ( double number )
{
   int value = number / 3;

   return value;
}
```

58.
```
void area(int length = 30, int width)
{
   return length * width;
}
```

59.
```
void getValue(int value&)
{
   cout << "Enter a value: ";
   cin >> value&;
}
```

60. *(Overloaded functions)*
```
int getValue()
{
   int inputValue;
   cout << "Enter an integer: ";
   cin >> inputValue;
   return inputValue;
}

double getValue()
{
   double inputValue;
   cout << "Enter a floating-point number: ";
   cin >> inputValue;
   return inputValue;
}
```

Programming Challenges

1. Markup

VideoNote
Solving the Markup Problem

Write a program that asks the user to enter an item's wholesale cost and its markup percentage. It should then display the item's retail price. For example:

- If an item's wholesale cost is 5.00 and its markup percentage is 100 percent, then the item's retail price is 10.00.

- If an item's wholesale cost is 5.00 and its markup percentage is 50 percent, then the item's retail price is 7.50.

The program should have a function named `calculateRetail` that receives the wholesale cost and the markup percentage as arguments and returns the retail price of the item.

Input Validation: Do not accept negative values for either the wholesale cost of the item or the markup percentage.

2. Rectangle Area—Complete the Program

If you have downloaded this book's source code, you will find a partially written program named AreaRectangle.cpp in the Chapter 06 folder. Your job is to complete the program. When it is complete, the program will ask the user to enter the width and length of a rectangle, then display the rectangle's area. The program calls the following functions, which have not been written:

- `getLength`—This function should ask the user to enter the rectangle's length then return that value as a `double`.
- `getWidth`—This function should ask the user to enter the rectangle's width then return that value as a `double`.
- `getArea`—This function should accept the rectangle's length and width as arguments and return the rectangle's area. The area is calculated by multiplying the length by the width.
- `displayData`—This function should accept the rectangle's length, width, and area as arguments and display them in an appropriate message on the screen.

3. Winning Division

Write a program that determines which of a company's four divisions (Northeast, Southeast, Northwest, and Southwest) had the greatest sales for a quarter. It should include the following two functions, which are called by `main`:

- `double getSales()` is passed the name of a division. It asks the user for a division's quarterly sales figure, validates the input, then returns it. It should be called once for each division.
- `void findHighest()` is passed the four sales totals. It determines which is the largest and prints the name of the high-grossing division, along with its sales figure.

Input Validation: Do not accept dollar amounts less than $0.00.

4. Safest Driving Area

Write a program that determines which of five geographic regions within a major city (north, south, east, west, and central) had the fewest reported automobile accidents last year. It should have the following two functions, which are called by `main`:

- `int getNumAccidents()` is passed the name of a region. It asks the user for the number of automobile accidents reported in that region during the last year, validates the input, then returns it. It should be called once for each city region.
- `void findLowest()` is passed the five accident totals. It determines which is the smallest and prints the name of the region, along with its accident figure.

Input Validation: Do not accept an accident number that is less than 0.

5. Falling Distance

When an object is falling because of gravity, the following formula can be used to determine the distance the object falls in a specific time period:

$$d = {}^{1}\!/_{2}gt^2$$

The variables in the formula are as follows: d is the distance in meters, g is 9.8, and t is the amount of time, in seconds, that the object has been falling.

Write a function named `fallingDistance` that accepts an object's falling time (in seconds) as an argument. The function should return the distance, in meters, that the object has fallen during that time interval. Write a program that demonstrates the function by calling it in a loop that passes the values 1 through 10 as arguments and displays the return value.

6. Throwing Distance

When throwing an object, the distance the object can reach is defined by the initial speed v and the angle α with which the object is thrown. The actual formula for the distance D is as follows:

$$D = \frac{v^2 .\ \sin(2\alpha)}{g}$$

The gravity constant g is defined as 9.81 m/s^2.

Write a function named `throwingDistance` that accepts the initial speed and the angle (in degrees) as arguments, and returns the throwing distance. Demonstrate the function by calling it in a program that asks the user to enter a value for the speed and print out all angles from 0 to 90° with steps of 5°.

7. Weight on Another Planet

The weight of any object on a planet depends on both the radius and the mass of the planet itself, but the formula becomes very easy if you start with the object's weight on Earth. For example,

- for Venus, multiply the weight on Earth with 0.905
- for Mars, multiply with 0.3787
- for Jupiter, multiply with 2.53

Write a function named `calculateWeight` that accepts the name of the planet and an object's weight on Earth as arguments, and returns the weight on that planet. Demonstrate the function by calling it in a program that asks the user to enter values for the planet and the weight.

8. Coin Toss

Write a function named `coinToss` that simulates the tossing of a coin. When you call the function, it should generate a random number in the range of 1 through 2. If the random number is 1, the function should display "heads." If the random number is 2, the function should display "tails." Demonstrate the function in a program that asks the user how many times the coin should be tossed, then simulates the tossing of the coin that number of times.

9. Present Value

Suppose you want to deposit a certain amount of money into a savings account and then leave it alone to draw interest for the next 10 years. At the end of 10 years, you would like to have $10,000 in the account. How much do you need to deposit today to