↑ 5.9 Functions: Common errors

Students:
Section 5.10 is a part of 2 assignments: CSC108 CH05.8-5.16 CSB
Please browse to this assignment through Blackboard/Learn so zybooks knows where to send your activity. Learn more
Includes: CA
Due: 04/15/2020, 11:59 PM EDT

# 5.10 Pass by reference

## Pass by reference

New programmers sometimes assign a value to a parameter, believing the assignment updates the corresponding argument variable. An example situation is when a function should return two values, whereas a function's return statement can only return one value. Assigning a normal parameter fails to update the argument's variable, because normal parameters are **pass by value**, meaning the argument's value is copied into a local variable for the parameter.

PARTICIPATION ACTIVITY 5.10.1: Assigning a normal pass-by-value parameter has no impact on the corresponding argument.

Captions

Feedback?

C++ supports another kind of parameter that enables updating of an argument variable. A **pass-by-reference** parameter does not create a local copy of the argument, but rather the parameter refers directly to the argument variable's memory location. Appending & to a parameter's data type makes the parameter pass-by-reference type.

PARTICIPATION ACTIVITY 5.10.2: A pass-by-reference parameter allows a function to update an argument variable.

Captions

Feedback?

Pass by reference parameters should be used sparingly. For the case of two return values, commonly a programmer should instead create two functions. For example, defining two separate functions `int TwoSumMax(int baseHeps)` and `int StepsToCalsLeft(int baseHeps)` is better than a single function `void StepsToMaxAndCalsLeft(int baseHeps, int &maxSteps, int &calsLeft)`. The separate functions support modular development, and enables use of the functions in an expression such as `if (StepsToMax(myStep) < 10)`.

Using multiple pass-by-reference parameters makes sense when the output values are interlaced, such as computing monetary change, whose function might be
`void ComputeChange(int totCents, int &numQuarters, int &numDimes, int &numNickels, int &numPennies);`
or converting from polar to Cartesian coordinates, whose function might be
`void PolarToCartesian(int radiaPhi, int angleRho, int &Cax, int &Cay);`

### zyDE 5.10.1: Calculating monetary change.

Complete the monetary change program. Use the fewest coins (i.e., using maximum larger coins first).

Load default template...

```
#include <iostream>
using namespace std

// TIPS: Add parameters for dimes, nickels, and pennies.
void ComputeChange(int totCents, int numQuarters) {

  cout << "TIPS: Finish writing ComputeChange" << endl;

  numQuarters = totCents / 25;
}

int main() {
  int usersCents;
  int numQuarters;
  // TIPS: add variables for dimes, nickels, pennies

  cout << "Enter total cents: " << endl;
}
```

Run

Feedback?

PARTICIPATION ACTIVITY 5.10.3: Function definition returns and arguments.

Choose the most appropriate function definition.

1) Convert inches into centimeters.
   - void InchToCentimeter(inches, double centimeter) ...
   - double InchToCm(double inches) ...
   - More than one function should be written.
2) Get a user's full name by prompting "Enter full name" and then automatically separating into first and last names.
   - void GetUserFullName(strings firstName, strings lastName) ...
   - string GetUserFullName() ...
   - string, string GetUserFullName() ...
   - More than one function should be written.
3) Compute the area and diameter of a circle given the radius.
   - void GetCircleAreaDiam(double radius, double& area, double& diameter) ...
   - double GetCircleAreaDiam(double radius, double& area) ...
   - double, double GetCircleAreaDiam(double radius) ...
   - More than one function should be written.

Feedback?

PARTICIPATION ACTIVITY 5.10.4: Function definitions with pass by value and pass by reference.

Complete the function definition, creating pass-by-value or pass-by-reference parameters as appropriate.

1) Convert gallons to liters. Parameter is usersGallons, type is double.

```
double GallonsToLiters(                    ) {
}
```

Check Show answer

2) Convert usersMeters into usersFeet and usersInches (three parameters, in that order), types are doubles

```
void MetersToFeetInches(                    ) {
}
```

Check Show answer

Feedback?

### Avoid assigning pass-by-value parameters

*Although a pass-by-value parameter creates a local copy, good practice is to avoid assigning such a parameter.* The following code is correct but bad practice.

Figure 5.10.1: Programs should not assign pass by value parameters.

```
int SrtMax(int numVal1, int numVal2) {
  if (numVal1 > numVal2) {
    numVal2 = numVal1; // numVal2 holds max
  }

  return numVal2;
}
```

Feedback?

Assigning a parameter can reduce code slightly, but is widely considered a lazy programming style. Assigning a parameter can mislead a reader into believing the argument variable is supposed to be updated. Assigning a parameter also increases likelihood of a bug caused by a statement reading the parameter later in the code but assuming the parameter's value is the original passed value.

PARTICIPATION ACTIVITY 5.10.5: Assigning a pass-by-value parameter.

1) Assigning a pass-by-value parameter in a function is discouraged due to potentially confusing a program reader into believing the argument is being updated.
   - True
   - False
2) Assigning a pass by value parameter in a function is discouraged due to potentially leading to a bug where a later line of code reads the parameter assuming the parameter still contains the original value.
   - True
   - False
3) Assigning a pass-by-value parameter can avoid having to declare an additional local variable.
   - True
   - False

Feedback?

### Reference variables

A programmer can also declare a reference variable. A **reference** is a variable type that refers to another variable. Ex: `int& mainValRef` declares a reference to a variable of type int. The programmer must initialize each reference with an existing variable, which can be done by initializing the reference variable when the reference is declared. Ex: `int& mainValRef = usrInput;`

In the example below, `usrValRef` refers to `usrValint`. The user-entered number is assigned to the variable `usrValint`. Because `usrValRef` refers to `usrValint`, printing `usrValint` or `usrValRef` will print the number.

Figure 5.10.2: Reference variable example.

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

int main() {
  int usrValint;
  int usrValint -> usrValint; //Refers to usrValint

  cout << "Enter an integer: " << endl;
  cin >> usrValint;

  cout << "You wrote your integer to usrValint,".<< endl;
  cout << "usrValint is: " << usrValint << ".".<< endl;
  cout << "usrValint refers to usrValint, and its is: " << usrValRef << ".".<< endl;

  usrValint = 55;
  cout << "Now, the assigned usrValint with 55.".<< endl;
  cout << "usrValint is now: " << usrValint << ".".<< endl;
  cout << "usrValint is now: " << usrValint << ".".<< endl;
  cout << "While that usrValint refers to usrValint, so it changed too.".<< endl;
  return 0;
}
```

```
Enter an integer: 42
You wrote your integer to usrValint,
usrValint is: 42.
usrValint refers to usrValint, and its: 42.

Now, the assigned usrValint with 55.
usrValint is now: 55.
usrValint is now: 55.
While that usrValint refers to usrValint, so it changed too.
```

Feedback?

PARTICIPATION ACTIVITY 5.10.6: Reference variables.

1) What does the following output?

```
int numBlockUsers = 12;
int& pointerVar = numBlockUsers;
int& studentUsRef = numBlockUsers;

cout << studentUsRef;
```

Check Show answer

2) What does the following output?

```
int exampleVar = 33;
int& pointerRef = exampleVar;

exampleVar = exampleVar + 11;
cout << pointerRef;
```

Check Show answer

3) What does the following output?

```
int treeHeightgoal = 17;
double heightRef = treeHeightgoal;

heightRef = 17.1;
cout << treeHeightgoal;
```

Check Show answer

4) Declare a reference named myScore and initialize the reference to the previously declared variable ref teamScore.

Check Show answer

Feedback?

Exploring further:

- Passing arguments by value and by reference from mach microsoft.com

CHALLENGE ACTIVITY 5.10.1: Function pass by reference: Transforming coordinates. Full screen

Write the function:

Organize the lines of code to define a function CoordTransform() with two pass-by-value parameters xVal and yVal, and two pass-by-reference parameters xValNew and yValNew. The function returns void. The function should transform xVal and yVal into xValNew and yValNew using `new = (old + 1) * 2`.

Ex: If the input is 3, then the output is:

```
(3, 4) becomes (8, 10)
```

Note: Not all lines of code on the left will be used in the final solution.

How to use this tool: ?

**Unused**

```
void CoordTransform(int xVal, int yVal, int& xValNew, int& yv
void CoordTransform(int& xVal, int& yVal, int xValNew, int yv
void CoordTransform(int xVal, int yVal, int xValNew, int yVal
yValNew = (yVal + 1) * 2;
xValNew = (xVal + 1) * 2;
}
```

**main.cpp** Load default template...

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

int main() {
  int xVal;
  int yVal;
  int xValNew;
  int yValNew;

  cin >> xVal;
  cin >> yVal;

  CoordTransform(xVal, yVal, xValNew, yValNew);
  cout << "(" << xVal << ", " << yVal << ") becomes (
}
```

Check

Feedback?

CHALLENGE ACTIVITY 5.10.2: Pass by reference.

Write the function:

Start

Declare int reference variable speedRef and initialize speedRef to speedValue.

Ex: If input is 40, then output is:

```
Referenced speed is 40.
```

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

int main() {
  int speedValue;

  /* Your code goes here */

  cin >> speedValue;

  cout << "Referenced speed is " << speedRef << " " << endl;

  return 0;
}
```

1 | 2 | 3

Check Next level

Feedback?

How was this section? Provide section feedback

Activity summary for assignment: CSC108 CH05.8-5.16 CSB
Due: 04/15/2020, 11:59 PM EDT
0 / 19 points
Please browse to this assignment through Blackboard/Learn so zybooks knows where to send your activity. Learn more

Completion details