6.7 Functions: Common errors

A common error is to copy-and-paste code among functions but then not complete all necessary modifications to the pasted code. For example, a programmer might have developed and tested a function to convert a temperature value in Celsius to Fahrenheit, and then copied and modified the original function into a new function to convert Fahrenheit to Celsius as shown:

Figure 6.7.1: Copy-paste common error: Pasted code not properly modified. Find error on the right.

```
double Cel2Fah(double celVal) {
   double convTmp;
   double fahVal;

   convTmp = (9.0 / 5.0) * celVal;
   fahVal = convTmp + 32;

   return fahVal;
}
```

```
double Fah2Cel(double fahVal) {
   double convTmp;
   double celVal;

   convTmp = fahVal - 32;
   celVal = convTmp * (5.0 / 9.0);

   return fahVal;
}
```

Feedback?

The programmer forgot to change the return statement to return celVal rather than fahVal. Copying-and-pasting code is a common and useful time-saver, and can reduce errors by starting with known-correct code. Our advice is that when you copy-paste code, be extremely vigilant in making all necessary modifications. Just as the awareness that dark alleys or wet roads may be dangerous can cause you to vigilantly observe your surroundings or drive carefully, the awareness that copying-and-pasting is a common source of errors, may cause you to more vigilantly ensure you modify a pasted function correctly.

PARTICIPATION ACTIVITY 6.7.1: Copy-pasted sum-of-squares code.

Original parameters were num1, num2, num3. Original code was:

```
int sum;

sum = (num1 * num1) + (num2 * num2) + (num3 * num3);

return sum;
```

New parameters are num1, num2, num3, num4. Find the error in the copy-pasted new code below.

1)

```
int sum;

sum = (num1 * num1) + (num2 * num2) +
(num3 * num3) + (num3 * num4);

return sum;
```

Feedback?

Another common error is to return the wrong variable, such as typing `return convTmp;` instead of fahVal or celVal. The function will work and sometimes even return the correct value.

Failing to return a value for a function is another common error. If execution reaches the end of a function's statements, the function automatically returns. For a function with a void return type, such an automatic return poses no problem, although some programmers recommend including a return statement for clarity. But for a function defined to return a value, the returned value is undefined; the value could be anything. For example, the user-defined function below lacks a return statement:

Figure 6.7.2: Missing return statement common error: Program may sometimes work, leading to hard-to-find bug.

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

int StepsToFeet(int baseSteps) {
   const int FEET_PER_STEP = 3;  // Unit conversion
   int feetTot;                  // Corresponding feet to steps

   feetTot = baseSteps * FEET_PER_STEP;
}

int main() {
   int stepsInput;            // User defined steps
   int feetTot;               // Corresponding feet to steps

   // Prompt user for input
   cout << "Enter number of steps walked: ";
   cin >> stepsInput;

   // Call functions to convert steps to feet and calories
   feetTot = StepsToFeet(stepsInput);
   cout << "Feet: " << feetTot << endl;

   return 0;
}
```

```
Enter number of steps walked: 1000
Feet: 3000
```

Feedback?

Sometimes a function with a missing return statement (or just `return;`) still returns the correct value. The reason is that the compiler uses a memory location to return a value to the calling expression. That location may have also been used by the compiler to store a local variable of that function. If that local variable happens to be the item that was supposed to be returned, the value in that location is the correct return value. But a later seemingly unrelated change to a function, like defining a new variable, may cause the compiler to use different memory locations, and the function suddenly no longer returns the correct value, leading to a bewildered programmer.

PARTICIPATION ACTIVITY 6.7.2: Common function errors.

Find the error in the function's code.

1)
```
int ComputeSumOfSquares(int num1, int
num2) {
  int sum;

  sum = (num1 * num1) +
(num2 * num2);

  return;
}
```

2)
```
int ComputeEquation1(int num, int val,
int k) {
  int sum;

  sum = (num * val) + (k * val);

  return num;
}
```

Feedback?

PARTICIPATION ACTIVITY 6.7.3: Common function errors.

1) Forgetting to return a value from a function is a common error.
   - True
   - False

2) Copying-and-pasting code can lead

to common errors if all necessary changes are not made to the pasted code.

- True
- False

3) Returning the incorrect variable from a function is a common error.

- True
- False

4) Is this function correct for squaring an integer?

```
int sqr(int a) {
   int t;
   t = a * a;
}
```

- Yes
- No

5) Is this function correct for squaring an integer?

```
int sqr(int a) {
   int t;
   t = a * a;
   return a;
}
```

- Yes
- No

Feedback?

CHALLENGE ACTIVITY 6.7.1: Function errors: Copying one function to create another.

Using the CelsiusToKelvin function as a guide, create a new function, changing the name to KelvinToCelsius, and modifying the function accordingly.

```
2 using namespace std;
3
4 double CelsiusToKelvin(double valueCelsius) {
5    double valueKelvin;
6
7    valueKelvin = valueCelsius + 273.15;
8
9    return valueKelvin;
```

```
10 }
11
12 /* Your solution goes here  */
13 double KelvinToCelsius(double valueKelvin) {
14    double valueCelsius;
15
16    valueCelsius = valueKelvin - 273.15;
17
18    return valueCelsius;
19 }
20
21
22 int main() {
```

Run ✓ All tests passed

✓ Testing with Kelvin input of 283.15

Your value `10`

✓ Testing with Kelvin input of 0.0

Your value `-273.15`

Feedback?