

6.4 Functions with branches/loops

Example: Auction website fee calculator

A function's block of statements may include branches, loops, and other statements. The following example uses a function to compute the amount that an online auction/sales website charges a customer who sells an item online.

Figure 6.4.1: Function example: Determining fees given an item selling price for an auction website.

```
Enter item selling price (Ex:  
65.00): 9.95  
eBay fee: $1.7935  
  
...  
  
Enter item selling price (Ex:  
65.00): 40  
eBay fee: $5.7  
  
...  
  
Enter item selling price (Ex:  
65.00): 100  
eBay fee: $9.5  
  
...  
  
Enter item selling price (Ex:  
65.00): 500.15  
eBay fee: $29.5075  
  
...  
  
Enter item selling price (Ex:  
65.00): 2000  
eBay fee: $74.5
```

```

#include <iostream>
using namespace std;

/* Returns fee charged by ebay.com given the selling
   price of fixed-price books, movies, music, or
   video-games.
   Fee is $0.50 to list plus a % of the selling
   price:
   13% for $50.00 or less
   plus 5% for $50.01 to $1000.00
   plus 2% for $1000.01 or more
   Source: http://pages.ebay.com/help/sell/fees.html,
   2012.

   Note: double variables often are not used for
   dollars/cents,
   but here the dollar fraction may extend past two
   decimal places.
*/

// Function determines eBay price given item selling
// price
double EbayFee(double sellPrice) {
    const double BASE_LIST_FEE    = 0.50; // Listing
    Fee
    const double PERC_50_OR_LESS   = 0.13; // % $50 or
    less
    const double PERC_50_TO_1000   = 0.05; // %
    $50.01..$1000.00
    const double PERC_1000_OR_MORE = 0.02; // %
    $1000.01 or more
    double feeTotal;                //
    Resulting eBay fee

    feeTotal = BASE_LIST_FEE;

    // Determine additional fee based on selling price
    if (sellPrice <= 50.00) { // $50.00 or lower
        feeTotal = feeTotal + (sellPrice *
    PERC_50_OR_LESS);
    }
    else if (sellPrice <= 1000.00) { //
    $50.01..$1000.00
        feeTotal = feeTotal + (50 * PERC_50_OR_LESS )
        + ((sellPrice - 50) * PERC_50_TO_1000);
    }
    else { // $1000.01 and higher
        feeTotal = feeTotal + (50 * PERC_50_OR_LESS)
        + ((1000 - 50) * PERC_50_TO_1000)
        + ((sellPrice - 1000) * PERC_1000_OR_MORE);
    }

    return feeTotal;
}

int main() {
    double sellingPrice; // User defined selling
    price

    cout << "Enter item selling price (Ex: 65.00): ";
    cin >> sellingPrice;

    cout << "eBay fee: $" << EbayFee(sellingPrice) <<
    endl;

    return 0;
}

```

[Feedback?](#)**PARTICIPATION
ACTIVITY**

6.4.1: Analyzing the eBay fee calculator.

- 1) For any call to EbayFee() function, how many assignment statements for the variable `feeTotal` will execute?

Check[Show answer](#)

- 2) What does EbayFee() function return if its argument is 0.0 (show your answer in the form `#.##`)?

Check[Show answer](#)

- 3) What does EbayFee() function return if its argument is 100.00 (show your answer in the form `#.##`)?

Check[Show answer](#)[Feedback?](#)

Example: Least-common multiple calculator

The following is another example with user-defined functions. The functions keep `main()`'s behavior readable and understandable.

Figure 6.4.2: User-defined functions make `main()` easy to understand.

```
#include <iostream>
#include <cmath>
```

```

using namespace std;

// Function prompts user to enter positive non-zero
number
int GetPositiveNumber() {
    int userNum;

    userNum = 0;

    while (userNum <= 0) {
        cout << "Enter a positive number (>0): " << endl;
        cin >> userNum;

        if (userNum <= 0) {
            cout << "Invalid number." << endl;
        }
    }

    return userNum;
}

// Function returns greatest common divisor of two inputs
int FindGCD(int aVal, int bVal) {
    int numA;
    int numB;

    numA = aVal;
    numB = bVal;

    while (numA != numB) { // Euclid's algorithm
        if (numB > numA) {
            numB = numB - numA;
        }
        else {
            numA = numA - numB;
        }
    }

    return numA;
}

// Function returns least common multiple of two inputs
int FindLCM(int aVal, int bVal) {
    int lcmVal;

    lcmVal = abs(aVal * bVal) / FindGCD(aVal, bVal);

    return lcmVal;
}

int main() {
    int usrNumA;
    int usrNumB;
    int lcmResult;

    cout << "Enter value for first input" << endl;
    usrNumA = GetPositiveNumber();

    cout << endl << "Enter value for second input" <<
endl;
    usrNumB = GetPositiveNumber();

    lcmResult = FindLCM(usrNumA, usrNumB);

    cout << endl << "Least common multiple of " << usrNumA
        << " and " << usrNumB << " is " << lcmResult <<
endl;

    return 0;
}

```

Enter value for first input
Enter a positive number (>0):
13

Enter value for second input
Enter a positive number (>0):
7

Least common multiple of 13 and 7 is
91

}

[Feedback?](#)**PARTICIPATION
ACTIVITY**

6.4.2: Analyzing the least common multiple program.



- 1) Other than main(), which user-defined function calls another user-defined function? Just write the function name.

**Check**[Show answer](#)

- 2) How many user-defined function calls exist in the program code?

**Check**[Show answer](#)[Feedback?](#)**CHALLENGE
ACTIVITY**

6.4.1: Function with branch: Popcorn.



Complete function PrintPopcornTime(), with int parameter bagOunces, and void return type. If bagOunces is less than 2, print "Too small". If greater than 10, print "Too large". Otherwise, compute and print 6 * bagOunces followed by " seconds". End with a newline. Example output for ounces = 7:

42 seconds

```
5
6  /* Your solution goes here */
7  if(bagOunces<2){
8      cout << "Too small";
9  }else if (bagOunces > 10){
10     cout << "Too large";
11 }else{
```

```
12     cout << 6*bagOunces << " seconds";  
13 }  
14 cout<< endl;  
15  
16 }  
17  
18 int main() {  
19     int userOunces;  
20  
21     cin >> userOunces;  
22     PrintPopcornTime(userOunces);  
23  
24     return 0;  
25 }
```

Run

✓ All tests passed

✓ Testing with input 7

Your output

42 seconds

✓ Testing with input 1

Your output

Too small

✓ Testing with input 2

Your output

12 seconds

✓ Testing with input 11

Your output

Too large

[Feedback?](#)**CHALLENGE
ACTIVITY**

6.4.2: Function with loop: Shampoo.



Write a function `PrintShampooInstructions()`, with `int` parameter `numCycles`, and `void` return type. If `numCycles` is less than 1, print "Too few.". If more than 4, print "Too many.". Else, print "N: Lather and rinse." `numCycles` times, where N is the cycle number, followed by "Done.". End with a newline. Example output with input 2:

1: Lather and rinse.
2: Lather and rinse.
Done.

Hint: Declare and use a loop variable.

```
5 void PrintShampooInstructions(int numCycles){  
6     if(numCycles<1){  
7         cout << "Too few." << endl;  
8     }else if (numCycles > 4){  
9         cout << "Too many." << endl;  
10    }else{  
11        for(int i =1; i <= numCycles; i++){  
12            cout << i << ": Lather and rinse." << endl;  
13        }  
14        cout << "Done." << endl;  
15    }  
16 }  
17 }  
18  
19 int main() {  
20     int userCycles;  
21  
22     cin >> userCycles;  
23     PrintShampooInstructions(userCycles);  
24  
25     return 0;  
26 }
```

Run

✓ All tests passed

✓ Testing with input 0

Your output

Too few.

✓ Testing whether your code had a loop

Yours

Had a loop

✓ Testing with input 2

Your output

1: Lather and rinse.
2: Lather and rinse.
Done.

✓ Testing with input 4

1: Lather and rinse.
2: Lather and rinse.

Your output

```
3: Lather and rinse.  
4: Lather and rinse.  
Done.
```

✓ Testing with input 5

Your output

```
Too many.
```

✓ Testing with input -1

Your output

```
Too few.
```

[Feedback?](#)