“Straight-Line” Programs in C++
and Simple Arithmetic Expressions

1. Log in to your account and start up the Bloodshed Dev C/C++ System as described in Lab No. 1. Click on File → New → Source File.

2. Type the following C++ program (we shall use this as a template for the exercises that follow). This is slightly different from what we have been using in class. It includes a “hack” that will keep the Windows panel open after the execution of the program (if, in fact, it compiles and executes). [If there are problems with this template, alert your instructor.]

3. The first exercise seeks to clarify the effect of the following assignment statement presented during our last meeting: \( k++=k-1; \)

   In the section headed by the line comment “//my declarations here” enter the following declaration:

   ```
   int k = 0;
   ```

   [Do not forget the semicolon at the end of the statements.] And in the section headed by the line comment “//my statements here” enter the following statements:

   ```
   cout << "k = " << k << endl;
   k+=++k-1;
   cout << "k = " << k << endl;
   ```
The complete program should look like the following:

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

int main() {
    // my declarations here
    int k = 0;
    
    // my statements here
    cout << "k = " << k << endl;
    k = k - 1;
    cout << "k = " << k << endl;
    system("PAUSE");
    return 0;
}
```

Save and then compile the program as instructed in Lab No. 1. Run the resulting executable by clicking on Execute and then Run. [This is different from the instructions in Lab No. 1 – we are able to do this and not have the Windows panel disappear immediately because of the “hack” statement as described earlier.] The Windows panel should prompt the user with the following message before it disappears:

Press any key to continue . . .

Note the output of the run below:

Try to trace the evolution of the variable `k` in the code. What can you conclude about the readability of this syntactic form?

4. Next, we shall write simple code to empirically determine the results of Problems 18-20 on p. 75 of the textbook.

   For each of the problems, you should begin with a fresh version of the template (see No. 2 above).
For No. 18, in the section headed by the line comment “//my declarations here” enter the following declaration:

```cpp
float value_1(5.78263);
```

In the section headed by the line comment “//my statements here” enter the following statement:

```cpp
cout << "value_1 = " << value_1 << endl;
```

Compile and then run the resulting program and note the resulting output.

---

For No. 19, in the section headed by the line comment “//my declarations here” enter the following declaration:

```cpp
double value_4(66.45832);
```

In the section headed by the line comment “//my statements here” enter the following statement:

```cpp
cout << scientific << "value_4 = " << value_4 << endl;
```

Compile and then run the resulting program and note the resulting output.

---

For No. 20, in the section headed by the line comment “//my declarations here” enter the following declaration:

```cpp
int value_5(7750);
```

In the section headed by the line comment “//my statements here” enter the following statement:

```cpp
cout << "value_5 = " << fixed << value_5 << endl;
```

Compile and then run the resulting program and note the resulting output.

Consult your textbook (cf. p. 48) as to the meaning of the additional elements in the output statements of No. 19 and No. 20 above.

5. To further illustrate the simple formatting features available in C++, compile and run the code found on pp. 48 & 49 of the textbook. You can avoid typing in the code by accessing the text of the program through the CSCI 1205 course webpage (http://www.cs.uno.edu/~adlai/1205). The link should be under the Miscellaneous Notes section of the website. The code is also given below just in case you have no access to a browser at the time you are performing these lab exercises:
Compare your results with the textbook’s (and mine, given below).

6. Finally, test the code we developed in class last meeting. Recall that it was meant to solve Programming Problem No. 21 on p. 75 of the textbook. To test a program involves constructing a “test suite” which is a collection of input data, together with the expected output that should be produced by the input data. These are then used in the testing and the actual results compared with the expected values. If the code is correct, all output should be as expected.
Recall that the problem involved writing a C++ program that converts miles to kilometers. The program would accept a value assumed to be in miles and then produce the equivalent value in kilometers. The code we came up with was similar to the following:

```cpp
/*
 * A simple program that accepts values in miles
 * and converts this to kilometers (displaying the
 * result with a simple message.
 */

#include <iostream>
using namespace std;

const double MI_TO_KM_FACTOR=1.6093440;

int main() {  
    //Declare program variables.
    double valueInMiles(0.0), valueInKilometers(0.0);
    
    //Prompt and accept input value (in miles)
    cout << "Enter value (in miles): ";
    cin >> valueInMiles;
    
    //Compute the equivalent in kilometers
    valueInKilometers = valueInMiles * MI_TO_KM_FACTOR;
    
    //Display the result
    cout << valueInMiles << " miles is equivalent to "
    << valueInKilometers << " kilometers." << endl;
    system("PAUSE");
    return 0;
}
```

You can avoid typing in the code just like in the previous exercise by accessing the program through the class website in the Miscellaneous Notes section.

Our test suite consists of the following pairs of “input-expected_output” pairs. Your task is to execute the code above and enter the actual values obtained at the running of the program in the third column of the table below:

<table>
<thead>
<tr>
<th>Input Value (miles)</th>
<th>Expected Output (kilometers)</th>
<th>Actual Output (from run)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.609</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td>-1.609</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>160.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>40.225</td>
<td></td>
<td></td>
</tr>
<tr>
<td>186289</td>
<td>299739.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>