CIS 251 • C++ Programming
Laboratory Assignment 4: Loops

In this laboratory assignment you will learn how to use loop statements to obtain the revenue and expenses for a company to determine its annual profit or loss.

Background

The company for whom you are writing this program has gathered information about the revenue for each of twelve months in its fiscal year. It also has a report listing the annual expenses for each of several departments, the number of which is unknown.

You will need two loops in this program, and each loop will determine the **sum** of a series of values from input (section 3.4). The first loop, dealing with the revenue for each month, will be a **count-controlled loop** (the number of iterations is predetermined). For the second loop, the number of expenses is unknown, so a good option would be to design this loop as a **sentinel-controlled loop**, where a non-data value (e.g., a negative or zero value) allows the user to stop the loop.

After the input loops are finished, you will determine the profit or loss by subtracting the sum of the expenses from the sum of the revenue. In order to display the output properly, you will need a decision structure, such as an **if-else-if**, to display the value as a profit or loss (if the number is positive, it's a profit; if it's negative, it's a loss; if it's zero, the company broke even).

Step 1: Launch Visual Studio and Set Up an Empty C++ Project with Code

Open Visual Studio from the Start Menu. Close the start page if you wish to minimize the visual clutter on the screen. From the **File** menu, select **New** > **Project…** and make the following selections in the dialog:

1. Installed Templates / Project Type: **Visual C++** (may appear under **Other Languages**)
2. Subtype: **Win32**
3. Template: **Win32 Console Application**
4. Project Name and Location as desired (usually **Documents > Visual Studio 20xx > Projects**)
5. **Uncheck** the checkbox marked **Create directory for solution**

In the **Win32 Application Wizard**, click **Next** to proceed to **Application Settings**, select the checkbox labeled **Empty project**, and click **Finish**.

In the **Solution Explorer** (select it from the **View** menu if you don't see it), right-click the **Source Files** folder, select the **Add** submenu, and select **New Item…** from the submenu. In the **Add New Item** dialog:

1. Category: **Code**
2. File Type / Template: **C++ File (.cpp)**
3. File Name: **FLast_Lab04.cpp** (your first initial, your last name, an underscore, and Lab04)
4. **Browse…** to navigate to your flash drive (if you save your file to a laboratory or classroom computer instead, be sure to delete the file once you've turned in your submission)

You should now see an empty code window on your screen.

Step 2: Begin Your Code

The first element you should provide in your code is a set of comments identifying yourself and the purpose of your program. You may enter this as a single block comment or as a set of line comments. Include your name, the course number, the laboratory assignment number, the date (optional), and a brief description of the program's contents. Place each of these items on a separate line. (Review the Laboratory Assignment 2 instructions for an example of these comments.)

As you continue your code, you may go ahead and add comments to the statements or groups of statements that you believe deserve additional explanation. You may also wait until after you have written your entire program to add comments.

It's a good idea to go ahead and set up the basic structure of your program as well:

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

int main()
{
    (Leave a blank line here for the space where you'll enter the statements for your program)
    return 0;
}
```

Step 3: Declare Your Variables and Constant

The following table describes the variables needed in your program:

Description	Data Form	Initial Value
Month counter	Whole number (from 1 to 12)	1
Revenue for one month	Real number	From input
Sum of revenue for entire year	Real number	0
Expenses for one department	Real number	From input
Sum of expenses for entire year	Real number	0
Profit or loss	Real number	Calculated

Add a **declaration** for each variable. You can declare all the variables of the same type in a single statement, or you can use one statement for each variable declaration. Remember: variable names cannot contain spaces.

Some of these variables need to start with an initial value. You may use assignment statements to set these values, or you may **initialize** the variables within the declarations (end of section 2.1).

HINT: Whenever a program calculates the sum of a series of values from input, the sum variable needs to start at zero. If a program calculates a product instead of a sum, the variable should start at one.

Step 4: Obtain Input from the User

At this point, the program needs to **prompt** the user to enter the twelve months of revenue using a **loop**, with an **input** statement allowing the user to enter the revenue for each month (assume the values in bold are entered by the user):

```
Company Profit / Loss Calculator

Revenue:
Please enter revenue for month 1:  12982.56
Please enter revenue for month 2:  20367.71
Please enter revenue for month 3:  18401.22
```

The other eight prompts are omitted for brevity but are nonetheless required.

```
Please enter revenue for month 12:  35909.12
```

Include the loop control variable (your month counter) in your output statement to label each value. As you read the input values from the user, add each one to the variable storing the total revenue for the year (section 3.4).

Next, your program should **prompt** the user for the expenses for several departments using a second **loop**. Provide the user some means of exiting the loop when there are no more expenses to enter:

```
Expenses:
Please enter an expense (or a negative value to exit):  34971.22
Please enter an expense (or a negative value to exit):  9568.81
Please enter an expense (or a negative value to exit):  10450.76
```

The number of expenses to be entered is unknown. The user indicates when there are no more values to enter.

```
Please enter an expense (or a negative value to exit):  -1
```

You may choose to use a **sentinel-controlled loop** or an **ask-before-iterating loop** (section 3.4). Remember to add each input value to the variable storing the total expenses for the year.

Step 5: Perform the Calculation

There is a single calculation in this program:

profit or loss = sum of revenue for the year – sum of expenses for the year

Use the variables you declared at the top of `main` to perform this calculation.

WARNING: If the spellings of the variables in the calculation statement do not match the spellings used earlier in your program, the code will not compile! Be consistent in your spelling!

Step 6: Display the Results with a Decision

Before you display the resulting profit or loss, write the three statements that tell the console to display decimal values in fixed notation, with a required decimal point, and two digits after the decimal point.

If the company made a profit (the profit or loss variable is positive), **display** an appropriate message:

`Your profit for the year is $`*profit or loss variable*

If the company lost money (the profit or loss variable is negative), **display** a different message (add a **negative sign** before the profit / loss variable to remove the negative sign from the output):

`Your loss for the year is $`*negated profit or loss variable*

Finally, **if** the company broke even (the profit or loss is zero), **display** an appropriate message:

```
You broke even this year.
```

WARNING: Do not enter a Boolean expression after a plain `else`! You only include a Boolean expression after an `if` or an `else-if`. The `else` automatically handles the condition in which the earlier Boolean expressions are `false`.

Step 7: Build Your Project

Select the item **Build Solution** from the **Build** menu. If the **Output** area below your code indicates that there are syntax errors, double-click each line that describes a syntax error to see the statement that contains the error (or, in some cases, the statement after the error).

Step 8: Run the Executable

Once you receive a message in the Output area that the build succeeded, select **Start Without Debugging** from the **Debug** menu, and enter some sample data for each choice in the console. You will need to select the Start Without Debugging command several times to test the program with different data sets.

HINT: Use "easy" numbers when testing your program. If you can do the math in your head, it's easier to verify that the results are correct. Make sure you vary the numbers that you use in different sets.

Step 9: Add Inline Comments and Submit Your Code

Make sure that your code has **three or more inline comments** near the statements that you believe warrant the additional description (especially those that deal with loops). These can be simple line comments at the end of a line of code that needs explanation, or block comments where each precedes a series of statements described in the comment.

In Blackboard, return to the Assignments page for the current assignment. Under the **Submission** section, click **Add Attachments**. Click **My Computer**; then, click the first **Browse…** button. Navigate to where you saved the source code file **FLast_Lab04.cpp** (with your name), select it, and then click **OK**. Verify that your file appears in the Submission section, and click the **Submit** button at the bottom of the page.

WARNING: If you do not click the Submit button, your attachment will not be saved! Once you click the Submit button, your assignment should move from the Inbox to the Submitted tab.