CIS 251 • C++ Programming
Laboratory Assignment 10: Pointers

In this laboratory assignment you will learn how to use C++ syntax to create a dynamic array, populate it with values from input, display its values in reverse order, and delete it when finished.

Background

Your goal is to write a program that displays a series of whole numbers from input in reverse order. Your program will prompt the user for the number of values in this list, so you can use the user's input as the size for a dynamic array declared after this prompt.

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

Open Visual Studio from the Start Menu. Start a new project of installed template / type **Visual C++**, subtype **Win32**, template **Win32 Console Application**, project name and location as desired (usually the Visual Studio **Projects** folder), and **Create directory for solution** box **unchecked**. In the **Win32 Application Wizard**, go to **Application Settings**, **check** the box for **Empty project**, and click **Finish**.

In the **Solution Explorer**, right-click **Source Files**, and select **Add > New Item…** from the shortcut menu. Select the category **Code** and the file type / template **C++ File (.cpp)**. Enter the file name **FLast_Lab10.cpp** (with your **first initial** and **last name** replacing **F** and **Last**, respectively). Click **Browse…** to select the location where you want to save your code (flash drive preferred).

Step 2: Begin Your Code

Add a single block comment or a set of line comments including 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. 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, or you may wait until after you have written your entire program to add comments.

Next, enter the standard items that are part of your program: the `include` and `using` directives needed for console I/O, the header for the `main` function, the opening curly bracket, a blank line after the bracket, the `return` statement, and the closing curly bracket.

Step 3: Declare Your Variables

The following table describes the variables needed in your `main` function (one per table row):

Description	Data Form	Initial Values
Array size	Whole number	From user input
Values entered	Pointer to a whole number	From user input
Subscript variable	Whole number	Assigned before loop

Add a declaration for each variable to the `main` function.

Do not allocate memory for the pointer yet. You'll need to do that after the user indicates how many values are to be entered in the list. Simply declare the pointer variable itself.

The textbook mentions that you can use the key word `typedef` to create a special type specifically for pointers to variables of a particular data type. You may do so in this assignment, but it is not required.

Step 4: Prompt the User for the Size and Create the Dynamic Array

Prompt the user for the number of values to be entered. As long as the user enters a non-positive value (0 or less), the program should display an error message and re-prompt the user for the size of the list.

Once the user has entered a positive size, use the key word `new` to assign a dynamic array of that size to the pointer variable.

Do not dereference the pointer when assigning the dynamic array to it. The inclusion of the asterisk in the declaration (step 3) is sufficient to indicate that it is a pointer; using an asterisk here would attempt to dereference a pointer variable that has yet to be assigned an address.

Step 5: Prompt the User for the Values

Use a loop to prompt the user for a value for each element in the dynamic array.

You can use square brackets after a pointer variable to access the element at each subscript just as you would with an ordinary array; no asterisk is required. Each subscript for this array must be within the bounds for an array of this size.

Step 6: Display the Values in Reverse Order

Use another loop to display the values the user entered in reverse order as compared to how the user entered them originally.

The program is not displaying the values in descending order (from greatest to least). You are not required to sort the values; simply display them in the order opposite to which they were entered.

A loop dealing with an array does not have to start with the first element; it can start at the highest subscript, with the counter decreasing by 1 during each iteration and stopping at the lowest subscript.

Step 7: Release the Memory for the Dynamic Array

When you reserve memory dynamically using a pointer variable, you should be sure to release that memory once you are done with it. Use the keyword `delete` to free up the memory allocated for the pointer variable.

You must use empty square brackets after the keyword `delete` if you are deallocating memory for a dynamic array; without the square brackets, the compiler assumes you are deallocating memory for a single dynamic variable.

Step 8: Build Your Project and Run the Executable

Build your project. If there are syntax errors, double-click each error description to get an idea of where the error is. Once the build succeeds, select **Start Without Debugging** from the Debug menu. Run the program several times, specifying different list sizes and verifying that the program operates as expected.

Example Run (your formatting and spacing may vary; assume the user enters the values in **bold**):

```
How many values would you like to enter? -2
Size must be positive.
How many values would you like to enter? 3

Enter value #1: 10
Enter value #2: 5
Enter value #3: 19

Here are the values you entered in reverse order:
Value #3: 19
Value #2: 5
Value #1: 10
```

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 pertain to pointers and dynamic arrays). 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_Lab10.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.

If you do not click the Submit button, your attachment will not be saved!