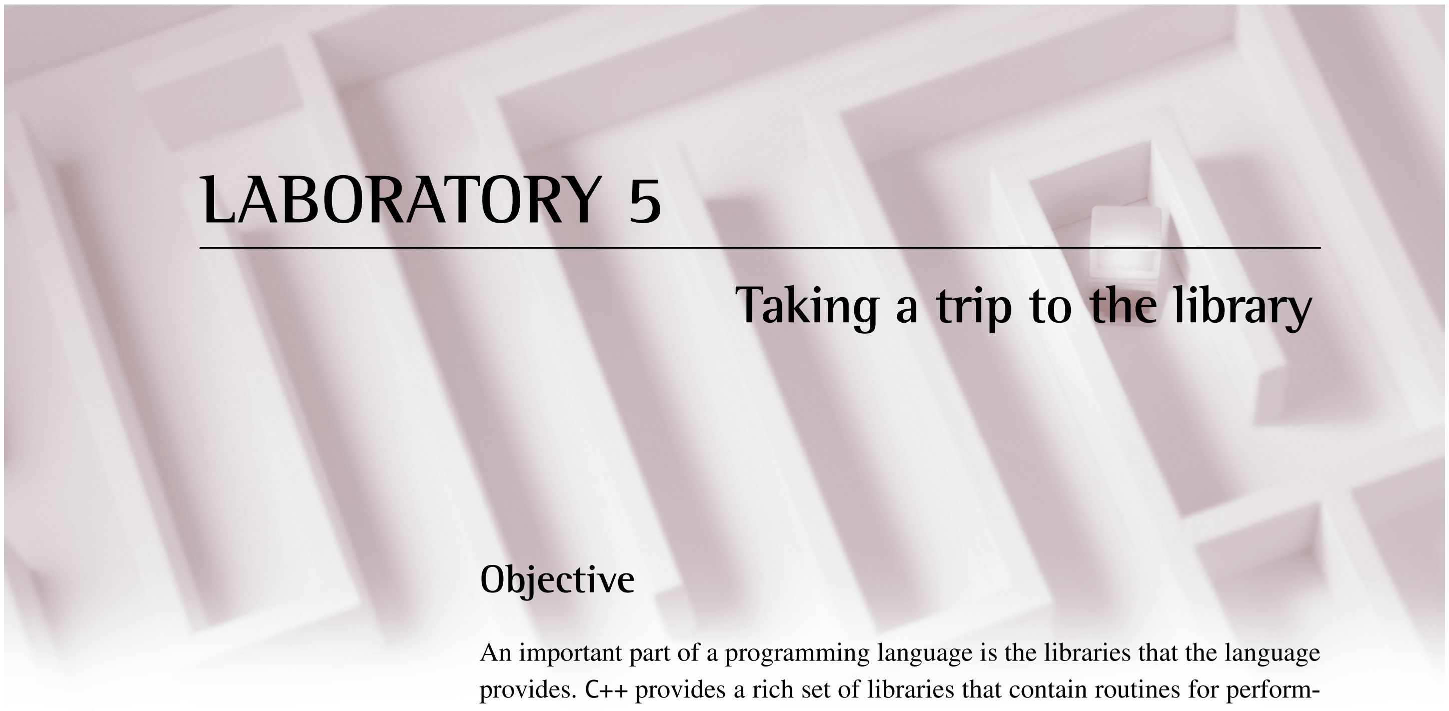


LABORATORY 5

Taking a trip to the library

Objective

An important part of a programming language is the libraries that the language provides. C++ provides a rich set of libraries that contain routines for performing a variety of tasks. One of the first things a professional programmer does when learning a new language is to become familiar with its various libraries. The purpose of this laboratory is to familiarize you with some of the most helpful C++ libraries and utility functions.

Key Concepts

- `string` class
- Character strings
- Using member functions

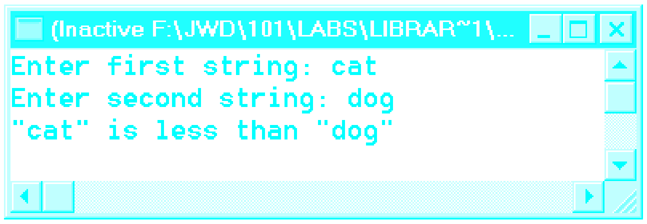
5.1 GETTING STARTED

- Using the procedures in the introductory laboratory handout, create the working directory `\cpplab` on the appropriate disk drive and obtain a copy of self-extracting archive `lab05.exe`. The copy should be placed in the `cpplab` directory. Execute the copy to extract the files necessary for this laboratory.
- Many of the activities that are performed in the laboratory can be done in groups but you should work the exercises yourself.

5.2 STRINGS

One of the important libraries or classes provided by C++ is the `string` class. Whereas the fundamental type `char` stores a single character, the `string` class enables you to create objects that hold a character sequence. We typically call this character sequence a string.

- Start up your compiler and open the file `string1.cpp`.
- Build and run `string1.cpp`. (You will need to create a default workspace).
- The `string` library enables you to compare strings the same way that you compare numbers. That is, you can use the equality operators == and `!=`, as well as the relational operators <, >, <=, and >=. For example, two strings are equal if they contain exactly the same characters. One string is less than a second string if the first string comes before the second string lexicographically (alphabetically). For example, the string `"cat"` is less than the string `"dog"`.
- Modify `string1.cpp` so that it accepts two strings as input from the user. The program should determine whether its two objects represent identical strings or whether the first input occurs lexicographically before or after the second input. When you run your program, its output should look like the following:



```
Enter first string: cat
Enter second string: dog
"cat" is less than "dog"
```

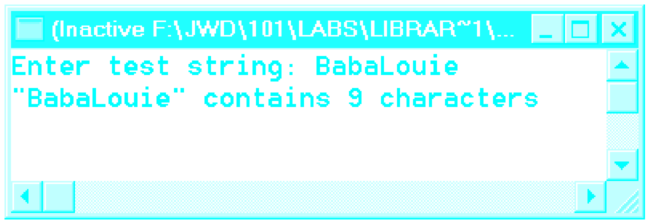
- Once you have your program working, try the following strings as input and record the results.

String 1	String 2	Result
computer	computers	
wizard	master	
wizard	Master	
data	daTa	

- Show your results to your laboratory instructor and explain each outcome. ✓
- Close the current workspace.
- The class string has many useful capabilities. For example, a string object can indicate how many characters it contains. If you had a string object named `TestString`, the code

  ```
  cout << TestString.size() << endl;
  ```

  would display the number of characters in `TestString`.
- To see this code work, open the program `string2.cpp`. Add code so that program displays the number of characters in a string. The output of your program should look like the following:



```
Enter test string: BabaLouie
"BabaLouie" contains 9 characters
```

- After making the required additions to `string2.cpp`, create a default workspace and run the program and use the following strings as input:

Input String	Result
computer-literate	
wizard	
dog and pony	
Onward!	

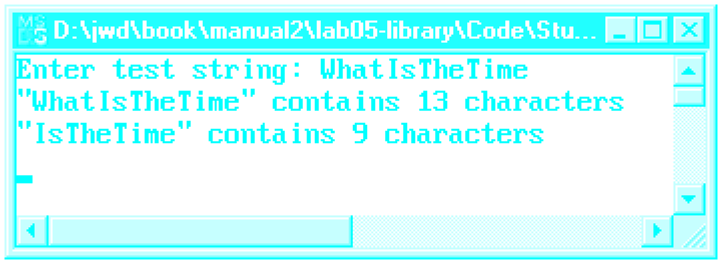
- Were the answers what you expected? Explain the results to your laboratory instructor. ✓

5.3 STRING MEMBER FUNCTIONS

A good question is, How do people find out about the various library routines that are available? Fortunately, modern program development environments make this easy. For example, to learn more about the capabilities of class `string`, you can use the on-line help.

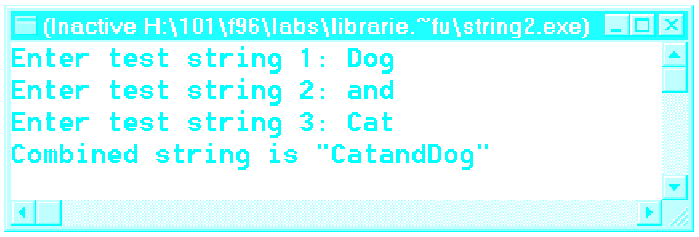
- A simple way to find out more about this class is to double-click on the word `string` in the program `string2.cpp` to highlight the word. Press `F1`. In the help window that appears display “string C/C++ Languages and C++ Libraries”. The window displayed indicates that `string` is specialization of the class `basic_string`. To see the description of `basic_string`, click on the hyperlink.

  There is some complicated stuff here, but don’t panic. Browse the rest of the help file and read the description of the functions.
- After examining the description of `basic_string` describe what the following member functions do:
  - `append()`
  - `empty()`
  - `erase()`
- Modify the program `string2.cpp` so that the first 4 characters of `TestString` are removed. The output of your modified program should look like the following:



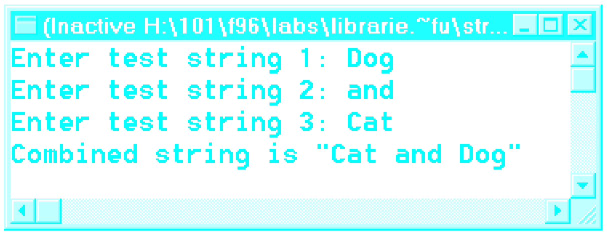
```
Enter test string: WhatIsTheTime
"WhatIsTheTime" contains 13 characters
"IsTheTime" contains 9 characters
```

- We can create a new string by appending one string to another. Modify `string2.cpp` so that it reads in three strings: `TestString1`, `TestString2`, and `TestString3`. Using the function `append()`, your program should create a new string that is the concatenation of `TestString3`, `TestString2`, and `TestString1` in that order. The following is a sample run.



```
Enter test string 1: Dog
Enter test string 2: and
Enter test string 3: Cat
Combined string is "CatandDog"
```

- However, what you really want is the following:



```
Enter test string 1: Dog
Enter test string 2: and
Enter test string 3: Cat
Combined string is "Cat and Dog"
```

✎ - Modify the program so that it produces output similar to the above. Demonstrate your program to your laboratory instructor. ✓

- You probably used the `append` member functions to create the required string. It turns out that C++ has overloaded the + operator so that it can operate on strings. Use the on-help and read about how the + operator is applied to strings. Revise `string2.cpp` to use the + operator instead of the member functions `append()`. Show your revised program to your laboratory instructor. ✓ ✎

- Close the current workspace.

5.4 WORD COUNT

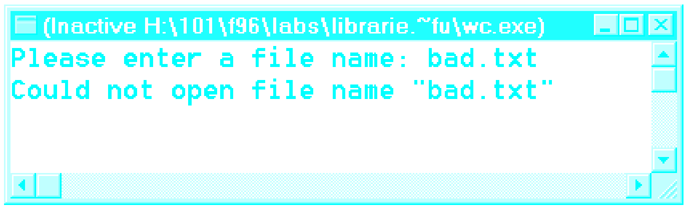
- Open the file `wc.cpp`. This file is the word count program. This handy program has a problem. It always reads its data from the same file. Examine the program and determine the name of this file.

- Modify the program so that it prompts the user for the name of the file to process and then extracts the name into a string. You can then open that file. Add code to the appropriate place in `wc.cpp` to prompt the use for a filename and to open that file. Create a default workspace and try to compile and execute the modified program. One compilation error that you may get is that the parameter to the `ifstream` constructor is not correct. The `ifstream` class expects the filename to be a conventional type `C` character string, not a `string` object. You need to change the name to a conventional string object. The class string has a member function to do so. Use on-line help to find the right function. If you are having trouble finding the right member function, ask your laboratory instructor for help.

✎ - Modify your program appropriately and make sure it compiles. Use the file `joke.txt` to demonstrate to your laboratory instructor that your program runs correctly. ✓

- What do you think happens if program cannot open the filename that is supplied? To find out, run your program and give it a filename that is not on your disk.

- Whenever you open a file, always checking to make sure that the operation was successful is a good idea. Otherwise, you can get erroneous results. To check whether a file was opened properly, you test the input stream object. If the stream object evaluates to `true`, the file was opened successfully. If the stream object evaluates to `false`, something went wrong and the file was not opened. Modify `wc.cpp` so that if it extracts a filename that it cannot open, it informs the user and exits. The following screen capture shows how your modified program should behave when it receives a filename that does not exist. ✓



```
Please enter a file name: bad.txt
Could not open file name "bad.txt"
```

5.5 FINISHING UP

- Copy any files you wish to keep to your own drive.
- Delete the directory `\cpplab`.
- Hand in your check-off sheet.