University of Computer and Emerging Sciences

Laboratory Manual

for

Data Structures Lab

Course Instructor	Ma'am Abeeda Akram
Lab Instructors	Sohaib Ahmad Ammara Nasir
Section	3G
Semester	Fall 2022

Department of Computer Science

FAST-NU, Lahore, Pakistan

Course Policies

1. After 5 minutes you will be marked late.
2. 80% attendance.
3. Discussion is not allowed during the lab.
4. Quizzes will be announced and unannounced.
5. No makeup for missed quiz.
6. Mobile phone usage within class will result in you being marked absent.
7. Labs have to be evaluated within the lab time, any submissions afterwards will not be accepted.

NOTE

Zero Tolerance for Plagiarism. Penalty will be given in accordance with the severity of plagiarism. This also includes forwarding the case to the DC Committee.

Objective of this lab:

After performing this lab, below mentioned concepts would be revised

1. Pointers
2. Templates

Instructions:

• Make a separate project for each task.
• Indent your code properly.
• Use meaningful variable and function names. Follow the naming conventions.
• Use meaningful prompt lines and labels for all input/output.
• Make sure that there are NO dangling pointers or memory leaks in your program.

<u>Question 1:</u>

Implement a class called myVector that stores integers in it. You have to implement the following members:

1. Private data member **arrayPtr** to integer array
2. Private data member **capacity** of array
3. Private data member that specifies the actual elements i.e.**current size**(elements added to the array by the user (initially it should be set to zero))
4. Default constructor (initially it should allocate an array of size 2. You will have to set the other private members too. Initialize the array to zero)
5. Constructor with int parameter (the int parameter specifies the capacity of the array). The constructor should initialize the entire array to zero.
6. Copy constructor (should make a deep copy)
7. Destructor (if the array is allocated then deallocate it)
8. Get function for total capacity…(think of other get functions)
9. + operator that takes an integer variable as parameter. This method should add one integer to the array and increase the number of current elements. If the allocated size is not sufficient then increase the capacity of the array by double. Copy the old array in the new one (including the new element) and delete the old array. b=
10. – operator that takes no parameters. This method should delete the last added value from the array and change the current size. (note no memory allocation or deallocation will take place here)
11. + operator that takes any Vector object as argument. What should be the exact type of that parameter? This operator should return a new array which has all the elements of the current object (lhs) and the array passed as parameter (rhs).
12. = operator which should make a deep copy. Make sure if lval is already allocated then it should be deallocated and allocated again with the size of the rval
13. [] operator which should place the value of an element at index in the parameter. If the index is out of bound then return false otherwise return true.
14. >>operator which takes as input the elements of myVector.
15. <<operator which prints the elements of myVector.

Provide a sufficient main program that tests all of the above functions. (including destructors too).

Question 2:
Convert the class myVector, which is implemented in Question 1, in a generic template class that can store any data value in it.

Question 3:
Design a specialized template class for myVector which is implemented in Question 2, to handle (char*) data type.

Reference Material

Vector:

Vectors are sequence containers representing arrays that can change in size.

Just like arrays, vectors use contiguous storage locations for their elements, which means that their elements can also be accessed using offsets on regular pointers to its elements, and just as efficiently as in arrays. But unlike arrays, their size can change dynamically, with their storage being handled automatically by the container.

Internally, vectors use a dynamically allocated array to store their elements. This array may need to be reallocated in order to grow in size when new elements are inserted, which implies allocating a new array and moving all elements to it. This is a relatively expensive task in terms of processing time, and thus, vectors do not reallocate each time an element is added to the container.

Links:

https://www.geeksforgeeks.org/vector-in-cpp-stl/
https://cplusplus.com/reference/vector/vector/