Name: Kaivan Shah

Section: 503

Csce 221: Data Structures and Algorithm

Programming Assignment Report

September 12, 2014

Part I
: Program Description

We need to create a vector in C++ called My_vec which can take charachter values and perform certain kind of actions using the functions defined. We need to declare/define those functions of the vector in its header file. Some of the functionsof the vector are get_size(); , insert_at_rank(); , get_max_index(); and so on. Then we need to create a generic version of My_vec such that it can hold any type of values. This can be achieved by using templates.

Part II
:Purpose of the assignment

The purpose of this assignment is to learn Data Structures and Algorithms. Also we learn to build a vector of our own from scratch. This vector will be able to perform the desired actions which we would allow it do. Also, the purpose of the assignment is to be able to write a generic program such that it could be used for any datatype(char, int, double etc.).

Part III
:Data Structures Description

- Theoretical Definition - A data structure is a theoretical concept of representing data and operations on them which can be implemented in any programming language (C, C++, Java, etc.) . It is a systematic way of organizing input data and specifying operations (algorithms) which can be performed on this data (e.g., add, delete, search).
- Real implementation - Data structure can be used to make a class of

students which can take their roll number, names and grades. Then taking in the values from the user and sorting them alphabetically according to their names or sorting them according to their grades and so on.

- Analysis of worst case and best case scenario for a vector- Worst case scenario would happen when we insert an element at a position say 1000 which would make the vector go crazy to increase its size exponentially. Not only that the vector would also have to undergo so many checks and then everytime it prints out it would have to print all the 1000 elements which would include the null values. The best case scenario would be when the user keeps inserting at a rank smaller than the size. That way it would have no probelm with increasing the size exponentially or so.

Part IV
:Instructions to compile and run the program

The data members and functions of the class are defined in the header file My_vec.h. The file My_vec.cpp is empty and main.cpp is used as a test run to check the how the vector works. We also need a makefile. Compile your program on Linux in the following way.

1.We open the folder where all the files are saved.
2.Type 'make' to compile the all the files mentioned in the make file.
3. Run the program by executing './main.

Part V
:Input and Output specifications

Input specification: They are to be given in the main.cpp file. Where we call the functions and pass the value through them. We use cout and overload the operator << to print the elements, size and so on for the vector.

Output specs: We can use the vector for Char_datatype, double_datatype and any other as we are using templates to print char, double etc..

Part VI
:Logical Exceptions

When we use insert_at_rank(); function to insert an element at the rank mentioned while calling the function. The error that I received was when inserting

at a rank bigger than size but less than capacity. So I just decided to enter null values until the rank and then insert the element at that position.

For char datatype, one could pass a int value and still work. But logically it would be incorrect.

Part VII

:C++ object oriented or generic programming features, C++11 features

In main we create objects of My_vec then use the data members and functions of the class through the object created. We use templates so that the program can basically be used for any datatype we want to. If we make an object of int datatype it can use the functions defined in the class on the integers. C+11 feature is the use of the vector. We essentially created our own vector named My_vec which will be arrays that can change size.

Part VIII

:Testing Results

Results:

Char datatype

The element(s) in v after insert are: B, The size of the vector v: 1

The element(s) in v after insert are: A, B, The size of the vector v: 2

The element(s) in v after insert are: A, B, , , , , , , , , D, The size of the vector v: 11

The element(s) in v after remove are: A, B, , , , , , , , D, The size of the vector v: 10

The element(s) in v after replace are: A, B, E, , , , , , , D, The size of the vector v: 10

The element(s) of v1 after copy from v are: A, B, E, , , , , , , D, The size of the vector v1: 10

The element(s) of v1 after replace are: Y, B, E, , , , , , , D,

The element(s) v2 after insert are: K, The size of the vector v2: 1

The element(s) v2 after copy from v1 are: Y, The size of the vector v2: 1

The element(s) v after inserting elements: L, Z, K, G, B, Y, The size of the vector v2 : 6

The max index of v2 is : 1 The element at the max index of v2 is : Z

The sorted max for v2 is: B, G, K, L, Y, Z, The elements of the vector v2 are: B, G, K, L, Y, Z, , , , , , , , , S, The size of the vector v2: 15

Int datatype

The element(s) vi after insert are: 1, The size of the vector vi: 1

The element(s) vi after insert are: 2, 1, The size of the vector vi: 2

The element(s) vi after insert are: 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 3,

The size of the vector vi: 11

The element(s) vi after remove are: 2, 1, 0, 0, 0, 0, 0, 0, 0, 3, The size of the vector vi: 10

The element(s) vi after replace are: 2, 1, 4, 0, 0, 0, 0, 0, 0, 3, The size of the vector vi: 10

The element(s) of vi1 after copy from v: 2, 1, 4, 0, 0, 0, 0, 0, 0, 3, The size of the vector vi1: 10

The element(s) of v1 after the replace: 2, 1, 5, 0, 0, 0, 0, 0, 0, 3,

The element(s) vi2 after insert are: 1, The size of the vector vi2: 1

The element(s) v2 after copy from v1: 2, The size of the vector v2: 1

The element(s) v after insert are: 6, 7, 9, 4, 8, 2, The size of the vector v2 : 6

The max index of vi2 is : 2 The element at the max index of v2 is : 9

The sorted max for v2 is: 2, 4, 6, 7, 8, 9, The elements of the vector v2 after replace are: 2, 4, 6, 7, 8, 9, 0, 0, 0, 0, 0, 0, 0, 0, 10, The size of the vector v2: 15

Int datatype

The element(s) vd after insert are: 1.5, The size of the vector vd: 1

The element(s) vd after insert are: 2.8, 1.5, The size of the vector vd: 2

The element(s) vd after insert are: 2.8, 1.5, 0, 0, 0, 0, 0, 0, 0, 0, 3.34,

The size of the vector vd: 11

The element(s) vd after remove are: 2.8, 0, 0, 0, 0, 0, 0, 0, 0, 3.34, The size of the vector vd: 10

The element(s) vd after replace are: 2.8, 0, 4.89, 0, 0, 0, 0, 0, 0, 3.34, The size of the vector vd: 10

The element(s) of vd1 after copy from vd: 2.8, 0, 4.89, 0, 0, 0, 0, 0, 0, 3.34, The size of the vector vd1: 10

The element(s) of vd1 after the replace: 2.8, 0, 5.2, 0, 0, 0, 0, 0, 0, 3.34,

The element(s) vd2 after insert are: 1.11, The size of the vector vd2: 1

The element(s) v2 after copy from vd1: 2.8, The size of the vector vd2: 1

The element(s) v after inserting elements: 6.6, 7.8, 9.1, 4.5, 8.3, 2.8, The size of the vector v2 : 6

The max index of vd2 is : 2 The element at the max index of vd2 is : 9.1

The sorted max for vd2 is: 2.8, 4.5, 6.6, 7.8, 8.3, 9.1, The elements of the vector vd2 after replace are: 2.8, 4.5, 6.6, 7.8, 8.3, 9.1, The size of the vector vd2: 6