Lab Manual: Vector Demonstration

Instructor: Muhammad Waseem

Course: Data Structures and Algorithms (DSA) Lab

Duration: 3 Hours
Languages: C++ & Python
Lab Objective:

- To understand the implementation and working of dynamic arrays ('vector' in C++ and 'list' in Python).
- To demonstrate how vectors dynamically resize in C++ and Python.

Part 1: Vectors in C++

Theory

In C++, a `vector` is a part of the Standard Template Library (STL) and is used to implement dynamic arrays that can resize themselves as needed. Vectors are similar to arrays, but they can grow and shrink dynamically.

Code Example

```
#include <iostream>
#include <vector>
using namespace std;
int main()
{
    vector<int> vec;

    // Adding elements to the vector
    vec.push_back(10);
    vec.push_back(20);
    vec.push_back(30);
    vec.push_back(40);

    // Displaying elements of the vector
    cout << "Vector elements: ";
    for (int i = 0; i < vec.size(); ++i)
    {
        cout << vec[i] << " ";
    }
```

```
    // Showing the current size and capacity
    cout << "\nSize: " << vec.size() << ", Capacity: " << vec.
       capacity() << endl;

    // Adding more elements to show dynamic resizing
    vec.push_back(50);
    cout << "After adding one more element:\n";
    cout << "Size: " << vec.size() << ", Capacity: " << vec.
       capacity() << endl;

    return 0;
}
```

Explanation

- `push_back`: Adds a new element to the end of the vector, resizing if necessary.
- `size()`: Returns the number of elements in the vector.
- `capacity()`: Shows the number of elements the vector can hold before needing to allocate more memory.

Output

```
Vector elements: 10 20 30 40
Size: 4, Capacity: 4
After adding one more element:
Size: 5, Capacity: 8
```

—

Part 2: Lists in Python

Theory

In Python, the `list` is a dynamic array-like structure that adjusts its size when elements are added or removed. While Python lists are simpler to use than C++ vectors, they are functionally equivalent in this context.

Code Example

```
# Initializing an empty list
my_list = []

# Adding elements to the list
my_list.append(10)
my_list.append(20)
my_list.append(30)
```

```
my_list.append(40)

# Displaying elements of the list
print("List elements:", my_list)

# Python doesn't expose capacity, but we can check the length
print("Size:", len(my_list))

# Adding one more element to demonstrate dynamic resizing
my_list.append(50)
print("After adding one more element:")
print("List elements:", my_list)
print("Size:", len(my_list))
```

Explanation

- `append()`: Adds a new element to the end of the list.
- `len()`: Returns the number of elements in the list.

Output

```
List elements: [10, 20, 30, 40]
Size: 4
After adding one more element:
List elements: [10, 20, 30, 40, 50]
Size: 5
```

—

Discussion Questions

1. How does the memory allocation strategy differ between C++ vectors and Python lists?
2. What are the advantages of dynamic arrays over static arrays?
3. In what scenarios might the automatic resizing of a vector be inefficient?

Conclusion

In this lab, we explored how dynamic arrays (`vector` in C++ and `list` in Python) function, how they resize themselves, and their basic operations. You should now understand the similarities and differences between how these structures manage memory and handle dynamic resizing.

Problems

C++ Problems

1. **Problem 1:** Write a program that initializes a vector of strings and allows the user to add or remove strings. The program should also display the size and capacity of the vector after each operation.

2. **Problem 2:** Write a program that inserts 100 integers into a vector. For each insertion, print the size and capacity of the vector. What do you observe about the capacity growth pattern?

3. **Problem 3:** Write a program to search for a specific integer in a vector. If found, print the index of the integer. If not, indicate that the integer is not present.

4. **Problem 4:** Implement a program that performs the following operations on a vector of integers:

   (a) Reverse the elements.

   (b) Sort the elements in ascending order.

   (c) Remove duplicates.

5. **Problem 5:** Write a program to create a 2D vector (a vector of vectors) representing a matrix. Implement functions to add rows, add columns, and display the matrix. Also, implement a function to transpose the matrix.

6. **Problem 6:** Create a class AutoGrowingArray with the following functionalities as follow:

```
AutoGrowingArray();
// Initially the array should be of Size 0 and capacity 1,
   then increment every time by 1.
// AutoGrow Array by increasing the size by 1 at each
   insertion
T operator [](int index) const;
T& operator [](int index);
friend ostream& operator << (ostream& out,const
   AutoGrowArray & Other);
~AutoGrowingArray();
void PushBack(T Value);
```

7. **Problem 7:** Create a class Vector with the following functionalities as follow:

```
Vector();
void PushBack(T Value);
// // Initially the array should be of Size 0 and Capacity
   1 and then increment every time by 1 and if the capacity
   is full i.e.
Size==Capacity it should resize by the factor of 2. This is
    the same implementation which is given in STL library
   called
#include <vector> in C++
```

```
T operator [](int index)const;
T& operator [](int index);
friend ostream& operator << (ostream& out,const Vector &
   Other);
~Vector();
```

8. **Problem 8:** Create a class ArrayList with the following functionalities as follow:

```
ArrayList();
void PushBack(T Value);
// // Initially the array should be of Size 0 and Capacity
   2 and then increment Size every time by 1 and if the
   capacity is full
i.e. Size==Capacity it should resize by the factor of 1.5.
   This is the same implementation which is given in Java
   generics
library called Arraylist.
T operator [](int index)const;
T& operator [](int index);
friend ostream& operator << (ostream& out,const ArrayList &
    Other);
~ArrayList();
```

9. **Experiment:** Using this function creates a 2 MB file. And Use the above three implementations and load both the files (please NEVER output the loaded data onto the screen as it will be two many characters, and console Write them down in another output file with the following names "OutputGA.txt", "OutputVector.txt". "OutputArraylist.txt", and measure the time taken in seconds for each experiment using time(0) function.

```
void CreateRandomFile(string fn, int Size, int RN=100)
{
    srand(time(0));
    ofstream Writer(fn);
    for (int i = 0; i < Size * 1024 * 1024; i++)
    {
        Writer << rand() % RN << " ";
    }
}
```

   This experiment must be done and screenshots should be attached in the submission.

Python Problems

1. **Problem 9:** Create a list that holds the names of students in your class. Write a Python script to add a student, remove a student, and display the list of students.

2. **Problem 10:** Write a program that appends integers from 1 to 100 to a list. Track the time it takes to append each integer. Does the time increase as the list grows?

3. **Problem 11:** Write a program to search for a specific string in a list of student names. Print the index of the name if found, and a message if the name is not present.

4. **Problem 12:** Implement a program that performs the following operations on a list of integers:

   (a) Remove all negative numbers.

   (b) Find the maximum and minimum values.

   (c) Compute the average of the elements.

5. **Problem 13:** Write a Python program to create a 2D list (a list of lists) representing a grid. Implement functions to add rows, add columns, and display the grid. Also, implement a function to find the sum of all elements in the grid.

LeetCode Questions related to vector arrays

Solve the following problems on leetcode. Each question carries 10 points.

1. https://leetcode.com/problems/how-many-numbers-are-smaller-than-the-current-number/

2. https://leetcode.com/problems/remove-element/

3. https://leetcode.com/problems/search-a-2d-matrix/

4. https://leetcode.com/problems/remove-duplicates-from-sorted-array/

5. https://leetcode.com/problems/maximum-subarray/

6. https://leetcode.com/problems/first-bad-version/

7. https://leetcode.com/problems/first-missing-positive/description/