DSA 2023 - Homework 1

Due Date: 05-Sep-2023

Task 1 - Create a class AutoGrowingArray with the following functionalities as follow:

AutoGrowingArray(); **10 points**

// Initially the array should be of Size 0 and 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);

- Drive the time complexity of the **AutoGrowingArray** implementation prove it is $O(N^2)$ for N operations of adding a value inside.
- Generate and Load 1, 2, 3 and 4MB Data into **AutoGrowingArray** and Predict how much time it will take for loading 1GB and 1TB.

Task 2: Create a class Vector with the following functionalities as follow:

Vector(); **15 points**

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();

- Drive the time complexity of the Vector implementation prove it is O(N) for N operations of adding a value inside.
- Generate and Load 10, 20, 30 and 40MB Data into Vector and Predict how much time it will take for loading 1GB and 1TB.

Task 3: Create a class ArrayList with the following functionalities as follow: **5 points**

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();

Drive the time complexity of the ArrayList implementation prove it is O(N) for N operations of adding a value inside.

- Generate and Load 10, 20, 30 and 40MB Data into **ArrayList** and Predict how much time it will take for loading 1GB and 1TB.

Do the following for the EXPERIMENT part of the above 3 problems

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 there will be too many elements in the output 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.

Task 4- 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/search-a-2d-matrix-ii/
5. https://leetcode.com/problems/remove-duplicates-from-sorted-array/
6. https://leetcode.com/problems/remove-duplicates-from-sorted-array-ii/
7. https://leetcode.com/problems/maximum-subarray/
8. https://leetcode.com/problems/maximal-rectangle/
9. https://leetcode.com/problems/first-bad-version/
10. https://leetcode.com/problems/first-missing-positive/