Data Structures – Spring 2023
Project 6
Due: May 3rd 2023 at 11:59 PM

Objective:
The goal of this project is to learn about hash functions from the standard template library.

Description:
Hash maps are one of the most commonly used data structures in computer science for storing and retrieving data in constant time. They are implemented using an array, where the index of the array represents the key and the value is stored in the corresponding array element. The hash function is used to map the key to its corresponding index in the array.

The map and unordered map are two of the most used hash map data structures in C++. The main difference between these two data structures is that the map is a sorted associative container that stores elements in a binary tree, while the unordered map is an unsorted associative container that uses a hash table to store elements.

The map is a red-black tree implementation, and it is typically used when elements need to be accessed in a specific order. Since the elements are stored in a binary tree, the time complexity for inserting and deleting an element is O(log n). This makes the map ideal for use cases where elements need to be sorted or accessed in a specific order.

On the other hand, the unordered map is implemented using a hash table, which makes it ideal for cases where fast insertions and deletions are required. The time complexity for inserting and deleting an element in an unordered map is typically O(1) on average, which is faster than the map. However, the order of the elements in an unordered map is not guaranteed since the elements are not stored in a specific order.

For your project, you are going to compare the performance of inserting and deleting elements in the map and unordered map data structures. You will measure the time taken to insert and delete a fixed number of elements/values (in this project, it is going to be a bunch of names of type strings) in both data structures and compare their performance. Additionally, you are going to analyze the results and provide insights as a single page report comparing the performance between map and unordered map by varying the number of operations (insertions + deletions/removal).

You are required to write your own hash function to find the key for each insertion/removal. Use chaining to handle collisions (hence the value of the map and unordered structures are a vector of strings). You must use the same hash function for both the data structures to make the comparison apples to apples. A good hash function is simple to understand and has less collisions. Please explain your hash function in your report as well.

Boiler plate code is given that goes has a bunch of libraries and some code to get you started.

Redirected Input file:

```
fileInput1.txt
```

The redirected input file just has one line in it which will be the name of the file to open to read in the commands. You open the fileInput txt file with the help of the ifstream library.
This is done so that you can open and close the file twice, to time the run of the map and unordered map. All commands will be in the fileInput txt file.

FileInput txt:

```
I Erin
I Adam
I Elsa
R Adam
R Erin
.
.
.
```

The fileInput txt file will have the commands given in the above format. There are insertions of names (strings) and removal of the same. You can assume that the names that are going to be removed will be present already. These insertions and removal have to be done on one data structure at a time (map and unordered map). You will have to open this file once to read the contents and create the data structure using map. After which you will close it and open it again to do build the data structure with unordered map.

Redirected input:

Redirected input provides you a way to send a file to the standard input of a program without typing it using the keyboard. To use redirected input in Visual Studio environment (on windows), follow these steps: After you have opened or created a new project, on the menu go to project, project properties, expand configuration properties until you see Debugging, on the right you will see a set of options, and in the command arguments type <"input filename". The < sign is for redirected input and the input filename is the name of the input file (including the path if not in the working directory).

If you use macOS or linux (or windows using powershell), you may use the terminal to compile and run your program using g++. This is the compiler that we use in the autograder on GradeScope (you may need to install g++ on your computer). Once you installed g++, you may compile your program using the following command:
g++ project6.cpp -o p6

You may then run the program with the redirected input using the following command:
./p6 < input1.txt

For powershell,
Get-Content input1.txt | ./p6

Make sure your program and your input file are in the same folder.

You may also use Google CoLab as an alternative and a sample file is given along with the project. This is an ipynb file which has to be opened in Google CoLab.

A simple program that reads the input file (one number) and displays everything that was read is given below.

```
#include<iostream>
using namespace std;

int main()
{
  int num = 0;
  cout << "Hello World" << endl;
  cin >> num;
  cout << num;
  return 0;
}
```

Sample output file for corresponding input files will be released. The input1.txt file given is a simple input file that will help you understand the project, more complicated ones will be released later and used for grading.

Report and timing:

You are required to measure the time taken to create both the data structures. You will create multiple input files of different lengths (more insertions and removals of names) to test both the structures. You may test with worst and best cases while inserting. You can create some file inputs and share it on the discussion board for others to test the same as well. I will use a couple of these test cases that are posted on the discussion board for grading in GradeScope as well.

```
// start timer
clock_t start1 = clock();

// read and perform the operations

// end timer
clock_t end1 = clock();
```

You will have to submit a single page report along side the source code comparing the performance between the two data structures for various input sizes (total operations = insertions + removal). You may draw a single chart/graph comparing the time and the number of operations for both the structures.

Submission:

Submission will be through GradeScope. Your program will be autograded with test cases and then hand graded to check for documentation and if you have followed the specifications given.

You may submit how many ever times to check if your program passes the test cases provided. Test cases will be released at the beginning and later other test cases will be released while grading. For the autograder to work, the program you upload must be named as **project6.cpp**.

Your final submission must contain your source file named **project6.cpp** and a one page report with experiment and plot named **project6_report.pdf**. You access GradeScope using the tab on the left in our course page on canvas.

Constraints:

1. In this project, the only header you will use the header files that are given to you in the boiler plate code. Using other or excess header files will be subject to a heavy grade penalty for the project.
2. None of the projects is a group project. Consulting with other members of this class on programming projects is strictly not allowed and plagiarism charges will be imposed on students who do not follow this.
    - Please review academic integrity policies in the modules.

How to ask for help:

1. Check FAQ discussion board of the project first.
2. Email the TAs with your precise questions.
3. Upload your well commented and clean program to CodePost.
    a. This is a website which is used to share code with your TA only.
    b. You will upload your program and I can view it and comment on it.
    c. Here is the invite link to our class for the summer.
    d. Once you join the class on CodePost, you can upload your program to the Project 1 assignment.

        Note: Your program will **not** be auto graded at CodePost, this is just for when you want to ask a question and a place where I can look at your program and comment on it. CodePost is great for live feedback. GradeScope is the place where you should submit and where your program will be autograded.