EECS 560 Lab – Implementation of Hash Table

Prof.: Dr. Shontz, GTA: Chiranjeevi Pippalla, Prashanthi Mallojula

100 Points

Due date:
11:59pm, Monday 02/10/2020 for Tuesday labs
11:59pm, Wednesday 02/12/2020 for Thursday labs

Purpose:
The purpose of this lab is to implement a hash table with separate chaining using a singly linked list in C++.

General Requirements:
In this lab, you will insert the records of soccer players along with their international goal counts into a hash table. Each record consists of the player name followed by their goal count. The player name should be of type string, and the goal count should be of type integer. The goal count of the player is considered as the key here. Using the goal count as the key, you are required to generate the hash value or the index of the hash table at which the record has to be placed. You are to read in the records from a data.txt file. **There shouldn't be any duplicate records inserted into the hash table.**

Let the index of the hash table be determined by the hash function (index = hashfunction(goalcount, bucketsize)) where the hashfunction(goalcount, bucketsize) is defined using the formula

hash_value = (goalcount % bucketsize),

where goalcount indicates the number of goals. For example, consider the goalcount to be 50. Also consider the bucketsize to be 10. Then the hash value will be computed as:

hash_value = 50 % 10.

The hash value is equal to 0. So the record with goalcount and respective player name will be inserted into the hash table at the index zero.

Finally, the function should return the index of the array at which the respective linked list is located. The index will be computed according to the formula index = (goalcount % bucketsize), where bucketsize will be passed as a parameter to the function. Here bucketsize is the total number of buckets, denoted by m in the upcoming formulas.

For example, the index for the goal count 72 will be 72 % 5 = 2, where 5 is the bucket size. That means, the record with a goal count of 72 will be inserted in the linked list starting at array index 2.

The file from which you should read the records is called 'data.txt'. You may hard code the file name if you wish. After building the structure, your program should ask the user to choose one of the options below:

EECS 560 Lab – Implementation of Hash Table

Prof.: Dr. Shontz, GTA: Chiranjeevi Pippalla, Prashanthi Mallojula

Operations:

The hash table should implement an appropriate constructor and destructor. The rest of the methods should be implemented as follows:

- AddPlayer(x): Should insert x into the hash table *when it is not found*. **All of the keys must be integers**. The number of goals field should be of **type integer**. Insertion must be done at the end of the chain. If you plan to use the Singly Linked List that you designed in your previous lab, make additional changes required in the insert function of the linked list such that the insertion of records will be from the end. Check the outputs given below to see the insertion sequences. The output will be "*x was added successfully*" when x is inserted; otherwise, the output will be "*x was not added successfully*". **Duplicate records are not allowed.**

- RemovePlayer(x): Should remove the record x from the hash table. The output should be either "Player x has been removed from the hashtable." or "No record found".

- PrintPlayersList: Should print out all the records of the hash table. **Each chain must be separated by an arrow and end with an endl**. **It should output from bucket 0 to bucket m-1 in the format:**

  Bucket 0: Element list
  Bucket 1: Element list
  ...
  Bucket m-1: Element list

- Hashfunction(goalcount, bucketsize) – Should compute the hash value of the index of the hash table using the goalcount field. Then take the modulus of the goalcount with bucket_size.

- PlayerWithGoalCountEqualTo(g) – Should output the list of players with a particular goal count defined by g. For example, if g is 72, the output should print the list of player names with the goal count of 72.

- PlayerWithNumGoalsGreaterThan(h) – Should output the list of players with goal count greater than or equal to h. For example, if h is 70, the output should print the list of player names with the goal count greater than or equal to 70.

- PlayerWithNumGoalsLessThan(i) – Should output the list of players with goal count less than or equal to i. For example, if g is 65, the output should print the list of player names with the goal count less than or equal to 65.

- Exit – Should exit the program.

Expected Results:

data.txt records:

Chhetri: 72, Ronaldo: 99, Chitalu: 84, John: 70, Hassan: 70, Messi: 70, Neymar: 61, Robert: 61, Radhi: 62

The following outputs are just for illustrative purposes only.

The bucket size should be greater than 5. For the following examples, the bucket size is considered as for illustrative purpose.

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>Input: 1
Enter the record to be inserted:
>Pele: 77
>Output: Record is successfully inserted.

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>3

0: -> John 70 -> Hassan 70 -> Messi 70
1: -> Neymar 61 -> Robert 61
2: -> Chhetri 72 -> Radhi 62 -> Pele 77
3:
4: -> Chitalu 84 -> Ronaldo 99

Please choose one of the following commands:
1- AddPlayer

2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>Input: 1
Enter the record to be inserted:
>Keane: 68
>Output: Record is successfully inserted.

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>3
0: -> John 70 -> Hassan 70 -> Messi 70
1: -> Neymar 61 -> Robert 61
2: -> Chhetri 72 -> Radhi 62 -> Pele 77
3: -> Keane 68
4: -> Chitalu 84 -> Ronaldo 99

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>2
Enter a record with required goals to be removed:
>100
>No record found

Please choose one of the following commands:

EECS 560 Lab – Implementation of Hash Table

Prof.: Dr. Shontz, GTA: Chiranjeevi Pippalla, Prashanthi Mallojula

1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>2
Enter a record with required goals to be removed:
>62
>Player Radhi has been removed from the hash table.

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>3

0: -> John 70 -> Hassan 70 -> Messi 70
1: -> Neymar 61 -> Robert 61
2: -> Chhetri 72 -> Pele 77
3: -> Keane 68
4: -> Chitalu 84 -> Ronaldo 99

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>4
Enter the goal count:
>70
>John, Hassan, Messi

EECS 560 Lab – Implementation of Hash Table

Prof.: Dr. Shontz, GTA: Chiranjeevi Pippalla, Prashanthi Mallojula

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>5
Enter the goal count:
>75
>Pele, Chitalu, Ronaldo

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>5
>Enter the goal count:
>100
>No players found

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>6
>Enter the goal count:
>60
>No players found

Please choose one of the following commands:

1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>6
>Enter the goal count:
>70
>Neymar, Robert, Keane, John, Hassan, Messi

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>4
>Enter the goal count:
>72
>Chhetri

Please choose one of the following commands:
1- AddPlayer
2- RemovePlayer
3- PrintPlayersList
4- PlayerWithGoalCountEqualTo(g)
5- PlayerWithNumGoalsGreaterThan(h)
6- PlayerWithNumGoalsLessThan(i)
7- Exit

>7

>Bye Bye!

Execution Instructions:

GDB/DDD can be used to debug your code. Make sure to use Valgrind to execute your code to check for any memory leaks/segmentation faults.

EECS 560 Lab – Implementation of Hash Table

Prof.: Dr. Shontz, GTA: Chiranjeevi Pippalla, Prashanthi Mallojula

Grading rubric:

- Full grade: Program should execute without any issues with all the options executed and with no memory leaks.
- Points will be taken for any execution errors, such as memory leaks, segmentation/program abort issues and missing handling of invalid cases.
- Programs that are compiled but do not execute will earn in the range of 0 to 50% of the possible points. Your grade will be determined based on the program design and the options implemented in the code.

Submission instructions:

- All files, i.e., the source files and Makefile, should be zipped in a folder.
- Include a ReadMe.txt if your code requires any special instructions to run.
- The naming convention of the folder should be LastName_Lab2.zip (or .tar or .rar or .gz).
- Email it to your respective lab instructor: chiranjeevi.pippalla@ku.edu (Chiru) or prashanthi.mallojula@ku.edu (Prashanthi) with subject line EECS 560 Lab2.
- Your program should compile and run on the **Linux machines** in **Eaton 1005D using g++.**