GTU Department of Computer Engineering
CSE 222/505 - Spring 2022
Homework 6 Report

Ömer Kaan Uslu
1801042642

1. **SYSTEM REQUIREMENTS**

1.Software Specification

Operating System : Windows 10 , macOS Catalina

Front End : Eclipse, Sublime Text

Rear End : Oracle SQL

Design Tool : UML

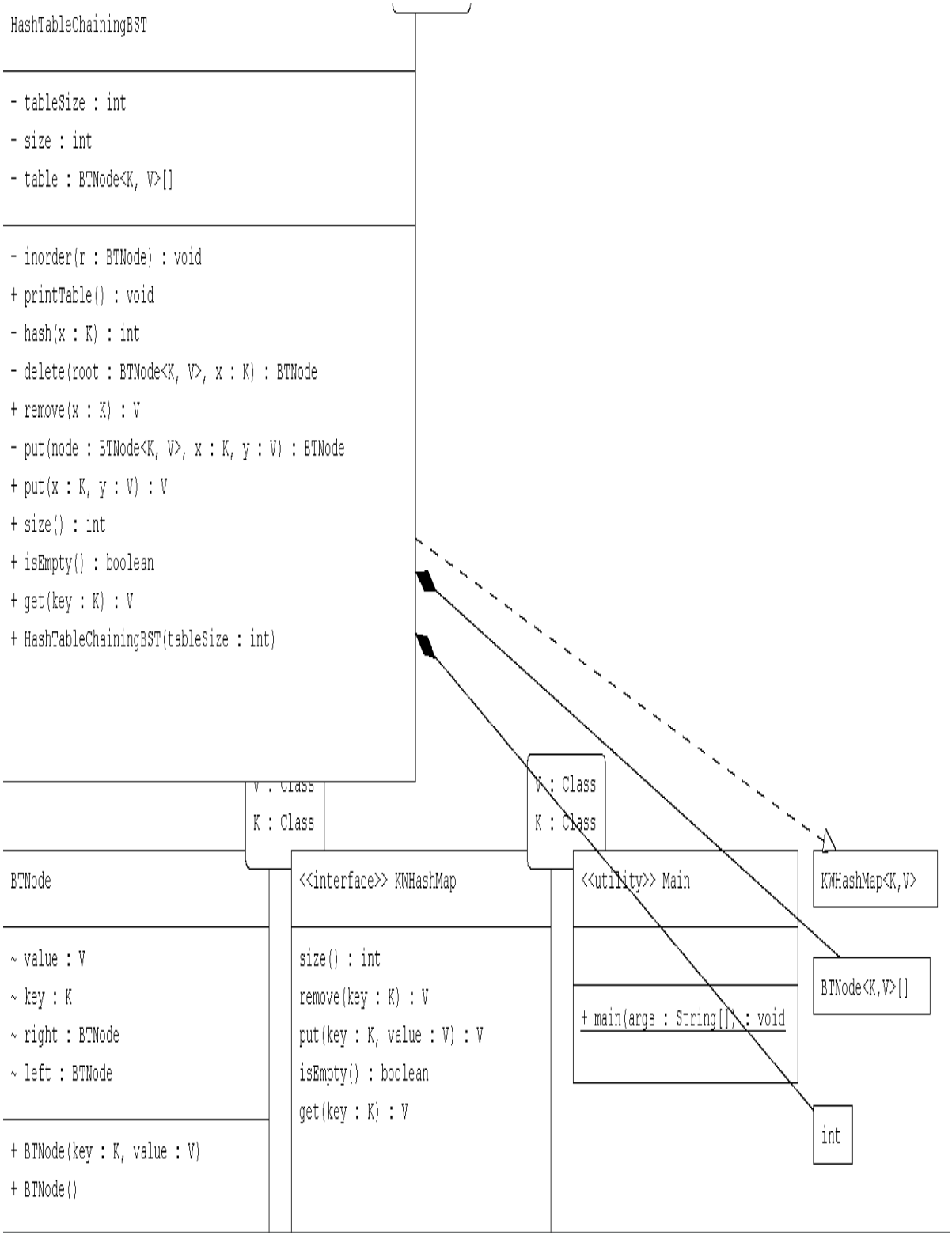
2.Hardware Specification

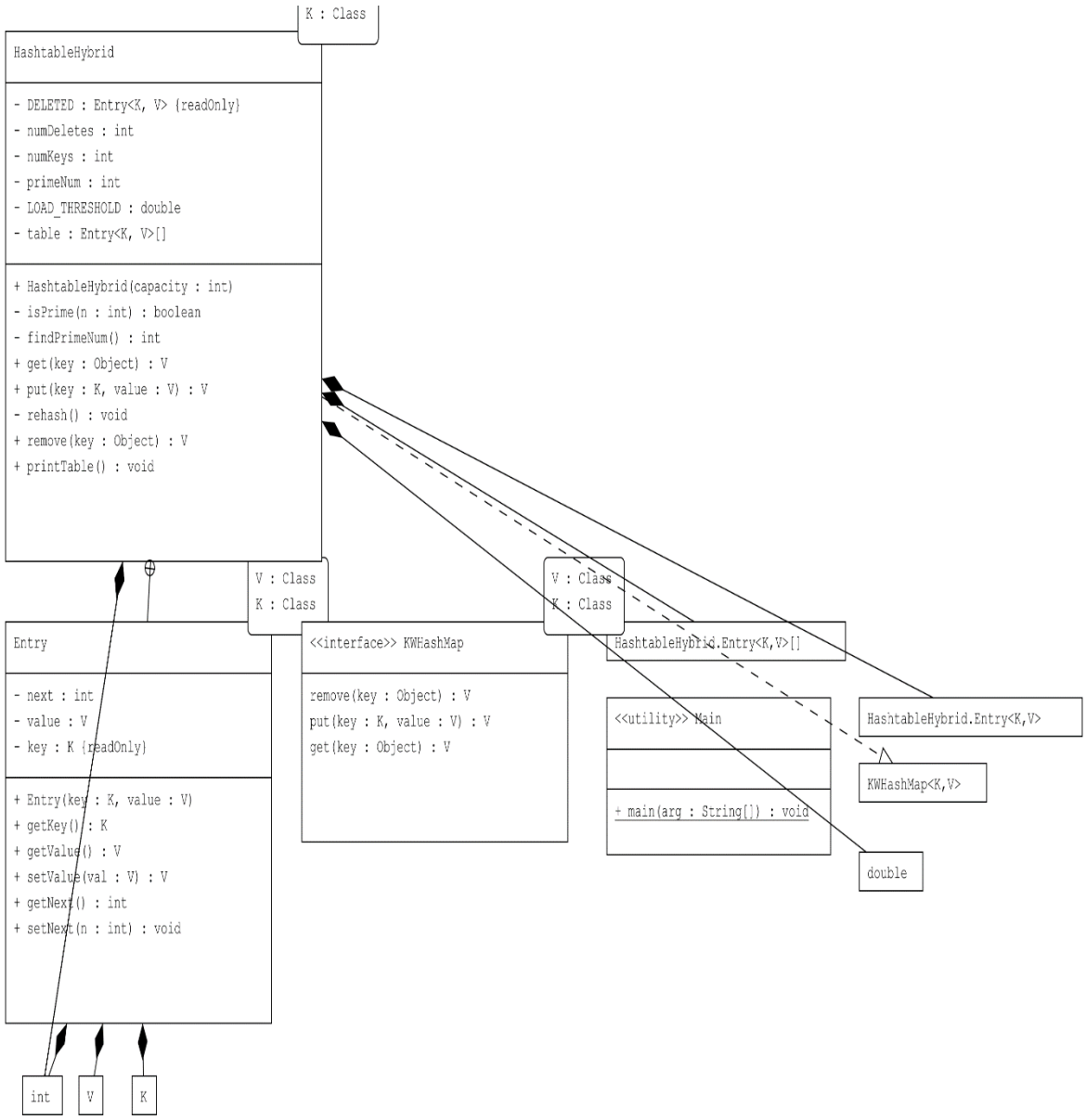
Processor : x86 processor

RAM : 512 MB or greater

Hard Disk : 20 GB or greater

2. USE CASE AND CLASS DIAGRAMS





3. **PROBLEM SOLUTION APPROACH**

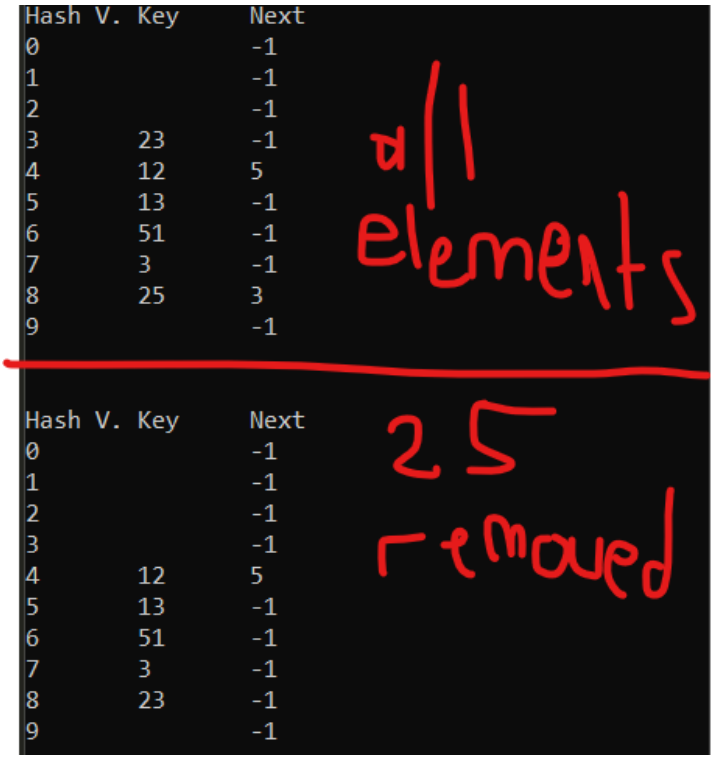
In first question, in the first part, I just put BST instead of linked list to do chaining method. Nothing different. Remove part has to be done with bst. It went like that.
In second part, actually it is no difference, it is same coalesced method. But hashing is different. And probe has to be done with same loop. Instead of incrementing i in basic coalesced hashmap, this version increments i in the formula. Nothing different.

In second question, I researched about finding max and min in array algorithm and come up Tournament Method for it. I just aplly rules of Tournament Method. It is very similar to binary search.

4. **TEST CASES**

Test Case No	Test Scenario	Test Steps	Test Data	Expected Result	Actual result	Pass/Fail
1	Add item to HashTable BST	Run driver code	7	7 is added	As expected	Pass
2	Remove item from HashTableBST	Run driver code	7	7 is removed	As expected	Pass
3	Add item to HashTable BST	Run driver code	7	7 is added	As expected	Pass
4	Remove item from HashTableBST	Run driver code	7	7 is added	As expected	Pass
5	Sort with Merge Sort	Run driver code	Random array	Array sorted	As expected	Pass
6	Sort with Quick Sort	Run driver code	Random array	Array sorted	As expected	Pass
7	Sort with New Sort	Run driver code	Random array	Array sorted	As expected	Pass

5. **RUNNING AND RESULTS**



```
--------HASH TABLE CHAINING BST EMPRICAL EXPERIMENT---------
--------100 ARRAY WITH 100 CAPACITY.---------

Hash V. Key

1       301,301 301,301
4       204,204
6       6,6 906,906
7       7,7 207,207 407,407
8       208,208
9       309,309 709,709
10      610,610 910,910
12      612,612 712,712
16      916,916
18      18,18
19      419,419
21      21,21
22      622,622
23      23,23 223,223 323,323
24      24,24 224,224 724,724
26      526,526
27      227,227 227,227
28      428,428 528,528
32      932,932
```

```
------------------------------------------------------------
--------HASH TABLE CHAINING BST EMPRICAL EXPERIMENT---------
--------100 ARRAY WITH 1000 CAPACITY.---------
------------------------------------------------------------
--------HASH TABLE CHAINING BST EMPRICAL EXPERIMENT---------
--------100 ARRAY WITH 10000 CAPACITY.---------
------------------------------------------------------------
  HASH TABLE CHAINING BST EMPRICAL EXPERIMENT(100) in nanoSeconds:  100704600

 HASH TABLE CHAINING BST EMPRICAL EXPERIMENT(1000) in nanoSeconds:  176890400

HASH TABLE CHAINING BST EMPRICAL EXPERIMENT(10000) in nanoSeconds:  2851178200
```

```
C:\Users\90555\Desktop\q2>java Main.java
MergeSort with size 100 Average time: 12849
QuickSort with size 100 Average time: 9171
NewSort with size 100 Average time: 51643
MergeSort with size 1000 Average time: 106718
QuickSort with size 1000 Average time: 180317
NewSort with size 1000 Average time: 1836704
MergeSort with size 10000 Average time: 1024179
QuickSort with size 10000 Average time: 44421396
NewSort with size 10000 Average time: 177651600
```

	Quick Sort	Merge Sort	New Sort
Theortical	T(n)=T(n-1)+Theta(n)	2T(n/2)+Theta(n)	nT(n/2)+n
Emprical(100) (10^-9s)	9171	12849	51643
Emprical(1000) (10^-9s)	180317	106718	1836704
Emprical(10000) (10^-9s)	44421396	1024179	177651600

Q2)

a) Coalesced Hashing is a very efficient way of storing elements in internal memory. Hash function hashes element to find a proper place to store it. If it is filled, then filled place linked with new place which is empty.
In searching, coalesced hashing is very advantageous. It hashes the element and searches it's next till hash does not match element. If the chains are short, it is very efficient over 2 algorithms.

In deletion, coalesced hashing is not preferable.Because there may be a lots of operation for link elements through back, it has no advantage over standart chaining method.

b) Double hashing is collision resolution technique. It uses 2 different hash function and unite this 2. It has advantage that, it require less comparisons. And in smaller size hash tables, it is very advantegous.
As elements in table increases, performance decreases. Because traversal may be harder than other hashing types.