C S 272/463 Introduction to Data Structures

Lab 11: Heap, Search, Hash

I. Learning objectives

Objectives 4, 6, 7, 8, 9.

II. Requirements

Q1. (30 points) Implement the following methods for a min heap (i.e., the top element of the heap is the smallest element) by using the `ArrayList` java class to store the heap elements. Put all your methods to MinHeap.java.

(1a) (10 points) Add a new element e into the heap.

```
public void add(int e)
```

(1b) (10 points) Get and remove the top element of the heap.

```
public int remove()
```

(1c) (5 points) Get the top element of the heap.

```
public int top()
```

(1d) (5 points) Put your test cases to the main method. You need to design test cases to test your program thoroughly. If your test cases cannot cover some important conditions, points may be deducted.

Q2. (10 points) Please design the binary search function to search an element e in an array A. Assume that all the elements in array A are useful elements, and the values in A are ordered in ascending order. Put your test cases to the main method. You need to design test cases to test your program thoroughly. If your test cases cannot cover some important conditions, points may be deducted. Put all your method and test code to search.java.

```
public static int binarySearch (int[] A, int e)
```

Q3. (10 points) Write a method that finds the `k` nearest elements to a `target` in a sorted integer ArrayList. `k` and `target` are given positive integers. The method needs to make use of the binary search technique. For example,

Input: [11, 13, 14, 17, 20, 23, 27], k = 4, target = 15

Output: [11, 13, 14, 17]

Input: [12, 13, 14, 15, 16, 17], k = 3, target = 11

Output: [12, 13, 14]

Input: [12, 13, 14, 15, 16, 17], k = 2, target = 18

Output: [16, 17]

```
public static ArrayList<Integer> findKNearest(ArrayList<Integer> A, int k, int target)
```

What is the time complexity of your method?

Q4. (30 points) Please design a hash table EmployeeTable to implement the open-address hashing data structure that we discussed in class. EmployeeTable is used to store information of employees. The employee structure is the data structure you implemented in lab 2. For this lab, each employee should also have an int variable employee_no, which is a unique identifying value of an employee. Implement the following methods for this class and put the code to EmployeeTable.java.

(4a) (5 points) Please include proper instance variables in EmployeeTable and include proper constructors.

(4b) (5 points) Please design the hash function to be the hash code of the employee no % size_of_array_for_keys

(4c) (5 points) Add a new employee e into the hash table.

```
public void put(Employee e)
```

(4d) (5 points) Remove a given employee with employee id emp_no from the hash table. Return false if an employee with emp_no does not exist in the hash table; Otherwise, remove it and return true.

```
public boolean remove(int emp_no)
```

(4e) (5 points) Find the index of the given employee id emp_no. Return the index of the employee in the array if the employee with the given employee no exists in the hash table; otherwise, return -1.

```
public int search(int emp_no)
```

(4f) (5 points) Put your test cases to the main method. You need to design test cases to test your program thoroughly. If your test cases cannot cover some important conditions, points may be deducted.

Q5. (20 points) The HashMap java class implements a hash table. Please implement a dictionary program using HashMap. The user inputs a word, and then the word's definition is displayed. You'll need to use HashMap with a string as both the key and the element types. For your reference, an example to use HashMap can be found from https://beginnersbook.com/2013/12/hashmap-in-java-with-example/. Please put the words in example_words.txt to a HashMap structure for testing your program. In example_words.txt, each line contains a word and its definition that are separated by a tab.

III. Note

- Specifications for all your classes and methods:

  Please properly explain (1) the functionality of the methods, (2) the parameters, (3) the return values, (4) the pre-conditions if there is any;

  Please use inline comments, meaningful variable names, indentation, formatting, and whitespace throughout your program to improve its readability.

- You can (but are not required to) design and implement other facilitating methods (E.g., other get and set methods, toString method) to finish the implementation of the required methods.

IV. Submission

Submit through canvas a zipped file containing your (1) java file(s) (not .class files).

V. Grading Criteria

1. The score allocation is beside the questions.
2. Please make sure that you test your code thoroughly by considering all possible test cases. Your code may be tested using more test cases.