Northeastern University
College of Engineering
Department of Electrical & Computer Engineering

EECE 2560: Fundamentals of Engineering Algorithms

Spring 2020 - Homework 5

Instructions

- For programming problems:
  1. Your code must be well commented by explaining what the lines of your program do. Have at least one comment for every 4 lines of code.
  2. You are not allowed to use any advanced C++ library unless it is clearly allowed by the problem. For example, you cannot use a library function to sort a list of data if the problem is asking you to implement an algorithm to sort the list.
  3. At the beginning of your source code files write your full name, students ID, and any special compiling/running instruction (if any).
  4. Test your code on the COE Linux server before submitting it:
     a. If your program does not compile in the COE server due to incompatible text encoding format, then before uploading your source code file to the server make sure it is saved with Encoding Unicode (UTF-8). In visual studio, Save As -> Click on the arrow next to Save -> Save with Encoding -> Yes -> Unicode (UTF-8) -> Ok
     b. Compile using g++ -std=c++11 <filename>

- Submit the following to the homework assignment page on Blackboard:
  1. Your homework report developed by a word processor (no handwritten or drawn contents are acceptable) and submitted as one PDF file. The report includes the following (depending on the assignment contents):
     a. Answers to the non-programming problems.
     b. A summary of your approach to solve the programming problems.
     c. A summary of the skills you acquired and challenges you faced by implementing the programs.
     d. Your recommendations of extension to the programming problems.
     e. The screen shots of the sample run(s) of your program(s)
  2. Your well-commented programs source code files (i.e., the .cc or .cpp files).

  Do NOT submit any files (e.g., the PDF report file and the source code files) as a compressed (zipped) package. Rather, upload each file individually.

__Note__: Yon can submit multiple attempts for this homework, however, only your last submitted attempt will be graded.

Problem 1 (30 Points)

Assume we have a hash table consisting of *m* = 11 slots, and suppose a nonnegative integer key *k* is hashed into the table using the hash function *h1*(*k*), which is defined as follows:

```
int h1(int key) {
  int x = (key + 5) * (key + 5);
  x = x / 16;
  x = x + key;
  x = x % 11;
  return x;
}
```

The sequence of 12 integer key values listed below are to be inserted in the table, in the order given. Suppose that collisions are resolved by using linear probing. Show the probe sequence (the sequence of slots to be probed until an empty one, if any, is available) for each key. In addition, show the final contents of the hash table after all keys are inserted.

Key Values	Probe Sequence
43	
23	
1	
0	
15	
31	
4	
7	
11	
3	
5	
9	

	Final Hash Table Contents
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

Problem 2 (30 Points)

Assume that a heap is implemented using an array called **`items`** where the root of the heap is stored in **`items[0]`**. Provide the mathematical proof of the following formulas that calculate the indices of the children and parent of any node **`items[`i`]`**.

- Left child of **`items[`i`]`** is **`items[2×`i` + 1]`**
- Right child of **`items[`i`]`** is **`items[2×`i` + 2]`**
- Parent of **`items[`i`]`** is **`items[`$\lfloor(i-1)/2\rfloor$`]`**

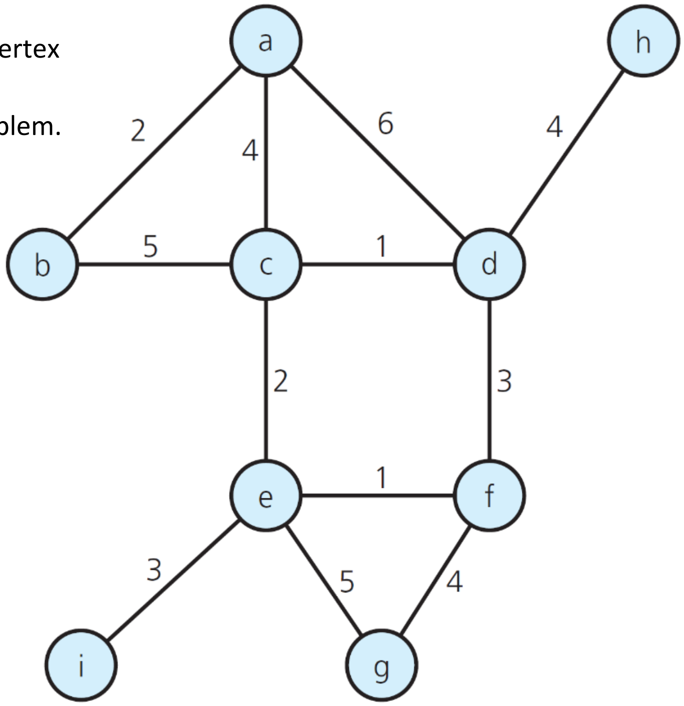
Problem 3 (40 Points)

For the following graph and beginning at vertex **a**, draw the following three trees:

3. Its BFS traversal tree (show the progress of the queue contents as described by the algorithm given in the lecture).
4. Its DFS traversal tree (show the progress of the stack contents as described by the algorithm given in the lecture).
5. Its minimum spanning tree (show that both Prim and Kruskal's algorithms result in minimum spanning trees with the same cost. Show the progress of applying both algorithms).

Notes:

- Make sure to break any ties by selecting the vertex whose letter comes alphabetically first.
- No programming code is required for this problem.



Extra Credit Problem 1 (10 Points)

In homework 4, you implemented, tested, and analyzed the following sorting algorithms: Selection, Bubble, Insertion, Merge, and Quick. Add to the C++ program you implemented the required code to implement the Heap Sort algorithm. Test the Heap Sort algorithm the same way you tested the other algorithms. Regenerate the two graphs for the number of moves and comparisons needed by the **six** algorithms for the three cases: best, average, and worst. Comment on the performance of the Heap Sort algorithm as compared to the other five algorithms.

Extra Credit Problem 2 (10 Points)

In homework 4, you implemented an event-driven simulation of a bank customer service. In that simulation we used a sorted list to implement the priority queue, which maintains the simulation events. However, using a heap to implement that priority queue results in a more time-efficient implementation. Make the necessary modifications in the PQueue class to use heap instead of a sorted list.