Hash Tables

(These questions refer to examples from chapter 12 of the course notes.)

From chapter 12, this is a description of Amy's hash function assuming an initial table size of 6:

"Amy uses an interesting fact. If she selects the third letter of each name, treating the letter as a number from 0 to 25, and then Mods the number by 6, each name yields a different number"

1. When Alan wishes to join the circle of six friends, why can't Amy simply increase the size of the table to seven?

The third letter of Alan is “a”, Alan should be put into the first index position.

2. Amy's club has grown, and now includes the following members:

Abel Abigail Abraham Ada

Adam Adrian Adrienne Agnes

Albert Alex Alfred Alice

a. Find what value would be computed by Amy's hash function for each member of the group, BEFORE modding by the table size.

Abel:4	Abigail:8	Abraham:17	Ada:0
Adam:0	Adrian:17	Adrienne:17	Agnes:13

Albert:1	Alex:4	Alfred:5	Alice:8	

b. Now, assume we use Amy's hash function and assign each member to a bucket by simply modding the hash value (obtained from part a) by the number of buckets. Determine how many elements would be assigned to each bucket (assume hashing with chaining) for a hash table of size 6. Do the same for a hash table of size 13.

Hush Table of size 6

0: Ada, Adam
1: Agnes, Albert
2: Abigail, Alice
3:
4: Abel, Alex
5: Abraham, Adrian, Adrienne, Alfred

Hush Table of size 13

0: Ada, Adam, Agenes
1: Albert
2:
3:
4: Abel, Abraham, Adrian, Adrienne, Alex
5: Alfred
6:
7:
8: Abigail, Alice
9:
10:
11:

12:

c. What are the load factors of these two tables?

Hash Table of size 6: Load Factor = 12 / 6 = 2

Hash Table of size 13: Load Factor = 12 / 13 = 0.9231

3. In searching for a good hash function over the set of integer values, one student thought he could use the following:

int index = (int) cos(value); // Cosine of 'value'

What was wrong with this choice?

Cos(value) maybe small than 0.

4. Can you come up with a perfect hash function for the names of days of the week? The names of the months of the year? Assume a table size of 10 for days of the week and 15 for names of the months. In case you cannot find any perfect hash functions, we will accept solutions that produce a small number of collisions (< 3).

. Days of the week: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday
. Monday: (13 * 0) % 10 = 0
. Tuesday: (4 * 1) % 10 = 4
. Wednesday: (3 * 2) % 10 = 6
. Thursday: (20 * 3) % 10 = 0
. Friday: (8 * 4) % 10 = 2
. Saturday: (19 * 5) % 10 = 5
. Sunday: (13 * 6) % 10 = 8
.

. Months of the year: January, February, March, April, May, June, July, August, September, October, November, December

. January: (13 * 1) % 15 = 13
. February: (1 * 2) % 15 = 2
. March: (17 * 3) % 15 = 6
. April: (17 * 4) % 15 = 8
. May: (24 * 5) % 15 = 0
. June: (13 * 6) % 15 = 3
. July: (11 * 7) % 15 = 2
. August: (6 * 8) % 15 = 3
. September: (15 * 9) % 15 = 0
. October: (19 * 10) % 15 = 10
. November: (21 * 11) % 15 = 6
. December (2 * 12) % 15 = 9

5. The function containsKey() can be used to see if a dictionary contains a given key. How could you determine if a dictionary contains a given value? What is the complexity of your procedure?

The best case is O(1), the worse case is O(n), the average case is O(1+).

Graphs

(These questions refer to examples from chapter 13 of the course notes.)

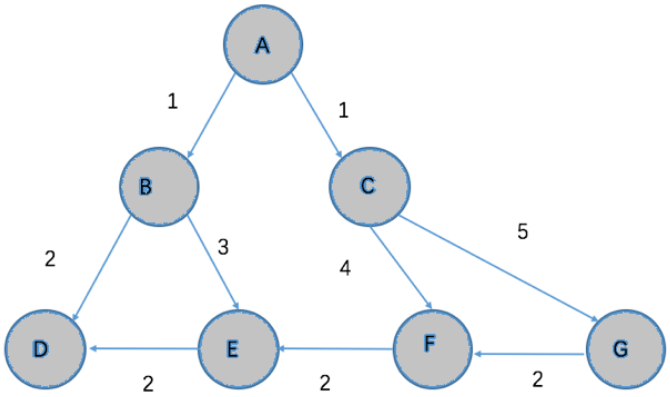
6. Describe the following graph as both an adjacency matrix and an edge list: graph

	1	2	3	4	5	6	7	8
1	?	1		1				
2		?	1					
3			?		1	1		
4				?	1			
5					?			
6						?	1	1
7					1		?	
8								?

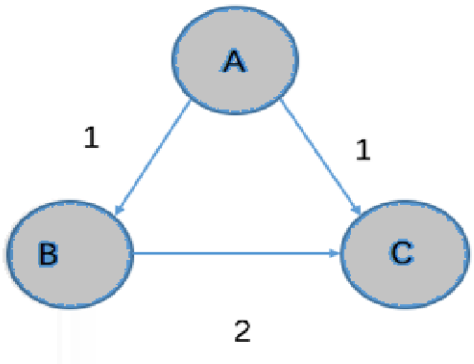
Edge List
1: {2, 4}
2: {3}
3: {5, 6}
4: {5}
5: { }
6: {7, 8}
7: {5}
8: { }

7. Construct a graph in which a depth first search will uncover a solution (discover reachability from one vertex to another) in fewer steps than will a breadth first search. You may need to specify an order in which neighbor vertices are visited. Construct another graph in which a breadth-first search will uncover a solution in fewer steps.

(a) The DFS



(b) The BFS



8. Complete Worksheet 41 (2 simulations). Show the content of the stack, queue, and the set of reachable nodes.

DFS

Iteration	Stack (T-B)	Reachable
0	1	{}
1	2 6	1
2	3 7 6	1 2
3	4 8 7 6	1 2 3
4	5 9 8 7 6	1 2 3 4
5	10 9 8 7 6	1 2 3 4 5
6	15 9 8 7 6	1 2 3 4 5 10
7	20 14 9 8 7 6	1 2 3 4 5 10 15
8	19 14 9 8 7 6	1 2 3 4 5 10 15 20
9	24 1 2 3 4 5 10 15 20 19	1 2 3 4 5 10 15 20 19

10	25 1 2 3 4 5 10 15 20 19	1 2 3 4 5 10 15 20 19 24
11	{}	1 2 3 4 5 10 15 20 19 25

BFS

Also with the clockwise

Iteration	Queue(F--B)	Reachable
0	1	{}
1	2 6	1
2	6 3 7	1 2
3	3 7 11	1 2 6
4	7 11 4 8	1 2 6 3
5	11 4 8	1 2 6 3 7
6	4 8 12 16	1 2 6 3 7 11
7	8 12 16 5 9	1 2 6 3 7 11 4
8	12 16 5 9 13	1 2 6 3 7 11 4 8
9	16 5 9 13 17	1 2 6 3 7 11 4 8 12
10	5 9 13 17 21	1 2 6 3 7 11 4 8 12 16
11	9 13 17 21 10	1 2 6 3 7 11 4 8 12 16 5
12	13 17 21 10 14	1 2 6 3 7 11 4 8 12 16 5 9
13	17 21 10 14 18	1 2 6 3 7 11 4 8 12 16 5 9 13

9. Complete Worksheet 42 (1 simulation). Show the content of the priority queue and the cities visited at each step.

Iteration	Rqueue	Reachable with cost
0	Pensacola: 0	{}
1	Phoenix:5	Pensacola: 0
2	Pueblo:8 Peoria:9	Phoenix:5

	Pittsburgh:15	
3	Peoria:9 Pierre:11 Pittsburgh:15	Pueblo:8
4	Pierre:11 Pueblo:12 Pittsburgh:14 Pittsburgh:15	Peoria:9
5	Pueblo:12 Pendleton:13 Pittsburgh:14 Pittsburgh:15	Pierre:11
6	Pendleton:13 Pittsburgh:14 Pittsburgh:15	--
7	Pittsburgh:14 Pittsburgh:15 Phoenix:17 Pueblo:21	Pendleton:13
8	Pittsburgh:15 Phoenix:17 Pensacola:18 Pueblo:21	Pittsburgh:14
9	Phoenix:17 Pensacola:18 Pueblo:21	--
10	Pensacola:18 Pueblo:21	--
11	Pueblo:21	--
12	{}	--

10. Why is it important that Dijkstra's algorithm stores intermediate results in a priority queue, rather than in an ordinary stack or queue?

The value with smallest distance is at the top of the queue.

11. How much space (in big-O notation) does an edge-list representation of a graph require?

. O(V+E)

12. For a graph with V vertices, how much space (in big-O notation) will an adjacency matrix require?

· $O(V^2)$

13. Suppose you have a graph representing a maze that is infinite in size, but there is a finite path from the start to the finish. Is a depth first search guaranteed to find the path? Is a breadth-first search? Explain why or why not.

. Breadth First Search can always find the path. Because Breadth First Search check all of the paths with different length one by one.