CSE 30 Fall 2016 Homework 2
Due at 11:59pm 30th November, 2016

Type your answers in a text file and submit it in CatCourses.
You can also write your answers on papers and scan them into image files for submission (Please do not take pictures of your answers with a camera. There are scanners in the library available for free scans.)

How to answer questions. Please be sure to:

- **read the problems and questions **carefully**!!!!**
- check answers - when possible
- lay out your solutions as neatly as possible in the space provided
- write down the reasoning used to arrive at the answer - when possible
  unsupported wrong answers may not receive any credit.

Each question is followed by space or lines to write your answers.

Points. The total number of points on this Homework is **100**.

Question	Score	Total
1		10
2		10
3		15
4		15
5		15
6		15
7		20
Total		**100**

Problem 1. [10 pts] Given a stack of objects of type *char*, write the **output** and the **stack contents** at the return of each function call. The stack is initially **empty**. (*Assume pop() doesn't return anything*)

Function call	**Output**	**Stack Contents** (bottom -> top)
push('C')		C
size()	1	C
top()	C	C
pop()	C	
empty()	True	
push('S')		S
push('E')		S, E
pop()	E	S
top()	S	S
empty()	False	S

Problem 2. [10 pts] Given a queue of objects of type *int*, write the **output** and the **queue contents** at the return of each function call. The queue is initially **empty**. (Assume dequeue() doesn't return anything)

Function call	**Output**	**Queue Contents** (rear -> front)
size()	0	
enqueue(1)		1
front()	1	1
dequeue()	1	
enqueue(3)		3
enqueue(5)		3, 5
empty()	False	3, 5
dequeue()	3	5
front()	5	5
empty()	False	5

Problem 3. [15 pts] Given the following tree, answer the following questions.

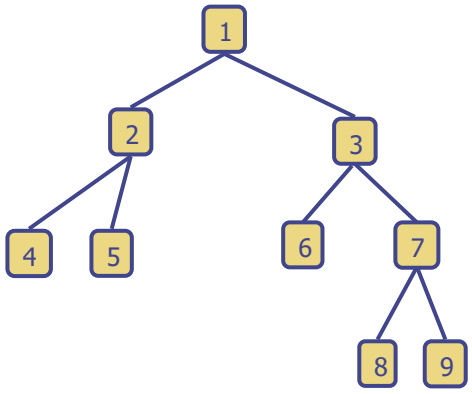
3.a [1 pts] Is it a binary tree? Why? Yes, because each parent has two childs.

3.b [2 pts] What are the **depth** and the **height** of the node containing 7? depth: 2 height: 3

3.c [5 pts] Write the **postorder** traversal: 4, 5, 2, 6, 8, 9, 7, 3, 1

3.d [5 pts] Write the **preorder** traversal: 1, 2, 4, 5, 3, 6, 7, 8, 9

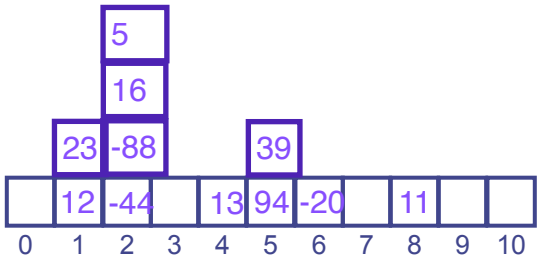
3.e [2 pts] Write all internal nodes: 1, 2, 3, 7



Problem 4. [15 pts] Given:

- the hash function: $h(x) = |3x - 2| \bmod M$
- a bucket array of capacity N
- a set of objects with keys: 12, -44, 13, -88, 23, 94, 11, 39, -20, 16, 5 (to input from left to right)

4.a [5 pts] Write the hash table where $M=N=11$ and collisions are handled using **separate chaining**.

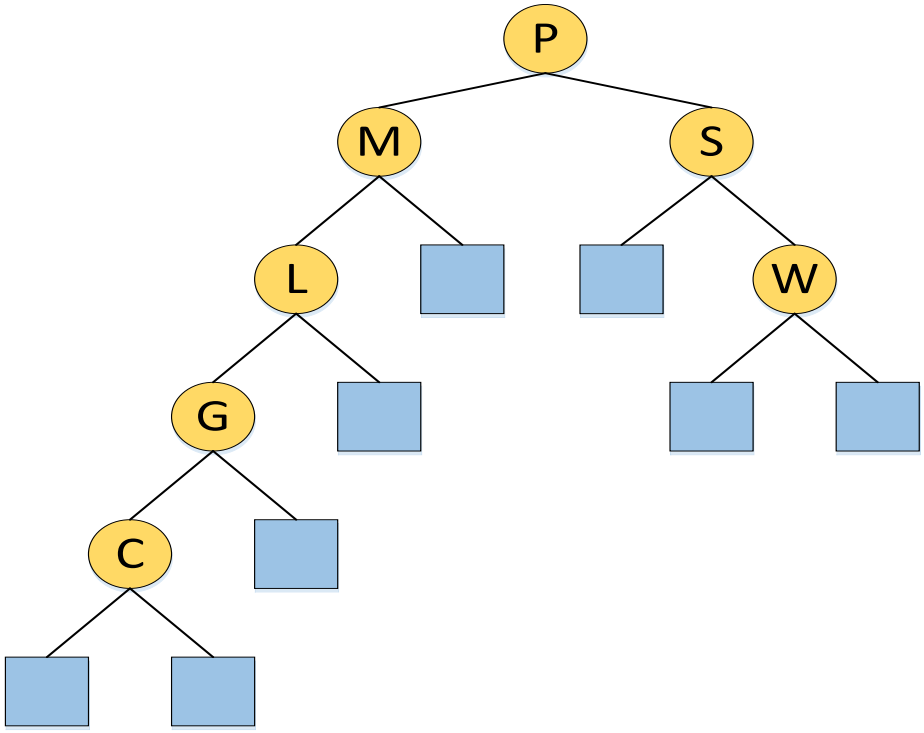


4.b [5 pts] Write the hash table where $M=N=11$ and collisions are handled using **linear probing**.

4.c [5 pts] Would a size N for the bucket array exist, such that no collisions would happen with the hash function hash function $h(x) = |3x - 2| \bmod 11$ *and the keys above?*

No, the same hash function makes it the same collision

Problem 5. [15 pts] Given the following binary search tree, with char keys and int elements



5.a [1 pts] What is the height of the tree? 5

5.b [1 pts] What's the key at the root of the tree? P

5.c [1 pts] How many leaves are in the tree? 8

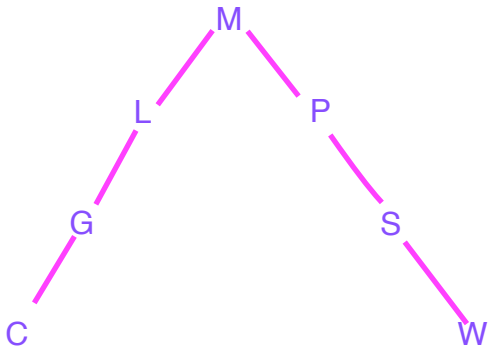
5.d [2 pts] What's the key of the sibling of the node with key S? M

5.e [2 pts] How many steps to find the key H? 6

5.f [2 pts] What is the minimum memory required if we want to store this tree with an array? 8

5.g [6 pts] Draw or describe rotation(s) and result to balance the tree, preserving inorder search:

Right Rotation:

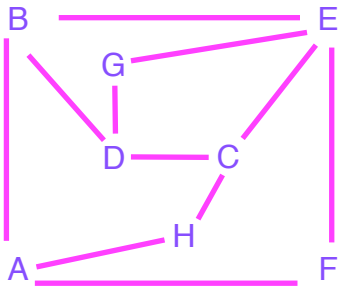


Problem 6. [15 pts] Let G be an undirected graph, where the vertices are all letters A-H

with adjacent vertices as listed in the following table:

VERTEX	ADJACENT VERTICES
A	B, F, H
B	A, D, E
C	D, E, H
D	B, C, G
E	B, C, F, G
F	A, E
G	D, E
H	A, C

6.a [5 pts] Draw the graph G.



6.b [5 pts] Write the sequence of vertices of G using a **DFS** traversal, starting at A:

A ,B,D,C,E,F,G,H

6.c [5 pts] Write the sequence of vertices of G using a **BFS** traversal, starting at A:

A ,B,F,H,D,E,C,G

Problem 7. [20 pts] Given the following class definition:

```
class Node {
   private:
      char element;
      Node* next;
   public:
      Node(char e){element=e, next=NULL;};
      Node(char e, Node* N){element=e;next=N;};

      void set_element(char e) {element=e;}
      char read_element() {return element;}
      void set_next(Node* N) {next=N;}

      char& ref_element() {return element;}
      Node*& ref_next() {return next;}
};
```

describe/draw the entities and memory usage utilized by the arguments and the return value of the member functions (variables, pointers, references, …).

For example, a function defined as an additional inline member of the class Node:

```
node* read_next() {return next;}
```

- **Return**: Value of type Node* pointer
- **Argument**: none

As another example, a function (unrelated to this class):

```
int f(int* p, int& a) {return (2+p*); }
```

- **Return**: Value of type integer
- **Arguments**:

Integer pointer	Reference of an integer

Answers:

7.a [3 pts] Node(char e, Node* N){element=e;next=N;};

- **Return**: N/A
- **Arguments**:

Character Variable	Node Pointer

7.b [3 pts] void set_element(char e) {element=e;}

- **Return**: N/A
- **Arguments**: Character e

7.c [3 pts] char read_element() {return element;}

- **Return**: Value of type char
- **Arguments**: None

7.d [3 pts] void set_next(Node* N) {next=N;}

- **Return**: N/A
- **Arguments**: Node Pointer

7.e [4 pts] char& ref_element() {return element;}

- **Return**: address of character
- **Arguments**: N/A

7.f [4 pts] Node*& ref_next() {return next;}

- **Return**: Address of pointer node
- **Arguments**: N/A