Questions

1. (12 points) What will be the value of sum after each of the following program segments complete?

```
(i)
sum = 0;
for (i = 0; i < N; i++){
  for (j = 0; j < i; j++){
    sum ++;
  }
}
```

```
(ii)
sum = 0;
for (i = 0; i < N; i++){
  k = pow(2, N);
  for (j = 1; j < k; j *= 2){
    sum ++;
  }
}
```

```
(iii)
sum = 0;
for (i = 0; i < N; i++){
  for (j = 0; j < N * N; j++){
    sum ++;
  }
}
```

2. (12 points)

- Write a function to evaluate

$$f(x) = \sum_{i=0}^{N} \frac{2^i}{i!}$$

  for a specific N.

- Give an analysis of running time of your algorithm.

3. (10 points) How will be the link list structure when the following program completed? Draw the link list.

```
void main(){
  Listptr l;
  l = empty_list();
  insert_front(l, 5);
  insert_back(l, 3);
  insert_back(l, 2);
  remove_front();
  insert_back(l, 8);
  insert_front(l, 3);
  insert_front(l, 2);
  remove_back();
}
```

4. (10 points) Implement delete_after for doubly link lists. Your function will take one parameter n, which points to the previous node of the deleted node. Prototype: **void delete_after(Nodeptr n)**You may assume that the deleted node is in the middle of the list. (No need for updating firstnode or lastnode of the list)

5. (10 points) Write the function ith_element which returns the address of the i'th element in the singly link list. Prototype: **Nodeptr ith_element(Listptr l, int i)**

6. (12 points) Show elements, head and tail links of the queue after each of the following operations. enqueue(3), enqueue(1), dequeue(), enqueue(4), dequeue(), enqueue(6), enqueue(9), dequeue(), enqueue(2), dequeue(), dequeue(), enqueue(5), enqueue(7), dequeue(), dequeue(), dequeue(). The size of the queue is 5. What is the minimum size of the queue so that we can do those operations without getting an error such as **empty queue** or **queue full**?

7. (12 points) Implement queue functions (empty_queue, enqueue, dequeue) using two stacks. (Hint: The data will be stored in the first stack. To dequeue an element from the queue, use the second stack.) Give also big O estimates of enqueue and dequeue operations.

```
struct queue{
            Stackptr s1;
            Stackptr s2;
};
typedef struct queue Queue;
typedef Queue* Queueptr;
Queueptr empty_queue();
void enqueue(Queueptr q, int data);
int dequeue(Queueptr q);
```

8. (12 points)

- Show the result of inserting 3, 1, 4, 6, 9, 2, 5, 7 into an initially empty binary search tree.
- What will be the output if we traverse the tree (i) preorder and (ii) inorder?
- Show the result of deleting the root.

9. (10 points)

- Write the function **int sum_of_tree(Treenodeptr r)** which computes the sum of all elements in the tree with root r.
- Write the function **int satisfy_tree_property(Treenodeptr r)** which returns 1 if the binary tree with root r satisfies the binary search tree order property at every node, 0 otherwise.