Write your answers on the answer sheets provided.

\[ \Theta(g(n)) = \{ f(n) \mid \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \text{ such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \} \]

1. (4 pts) Use the definition of \( \Theta \) to establish whether or not \( \frac{1}{2}n^2 - 120 \in \Theta(n^2) \). (If so, your answers should include specific selections for \( c_1, c_2, \text{ and } n_0 \)).

2. (2 pts) Suppose that you have shown that \( f(n) \in \Theta(g(n)) \) and that \( h(n) \in O(g(n)) \). What can you conclude about the relative rates of growth of the functions \( f(n) \) and \( h(n) \)?

3. (6 pts) Determine the complexity of the following algorithm and express your answer in terms of big-O. Explain/show your work. You may assume that the variable \( a \) is an array containing at least \( n \) integer values.

```plaintext
maxcount= 1;
count= 1;
for (i=1; i<n; i++) {
    if (a[i]==a[i-1])
        count= count + 1;
    else
        count= 1;
    if (count > maxcount)
        maxcount= count;
}

count= 1;
for (i=1; i<n; i++) {
    if (a[i]==a[i-1])
        count= count + 1;
    else
        count= 1;
    if (count==maxcount)
        print a[i];
}
```

4. (7 pts) Place the functions below in order from slowest growing to fastest growing. For each function, give the name representing the growth rate of that function: re-do this list: \( n^2 \), \( \log n \), \( n! \), \( n \log n^2 \), \( n^n \), \( n \), 5000.
5. (2 pts each) State (using $O$) the (worst-case) complexity of each of the following operations:

(a) insert in an unordered, linked list-based list
(b) search in an ordered, array-based list
(c) minimum in an ordered linked list-based list
(d) search in an ordered linked list-based list
(e) delete in an ordered, array-based list
(f) pop in an array-based stack
(g) quicksort (give both worst-case and average case complexity)

6. (4 pts) What is tail recursion and why is it significant?

7. (4 pts) Name one advantage of using a linked list instead of an array. Name one advantage of using an array instead of a linked list.

8. (5 pts) Using words give a recursive explanation of a binary search. NOTE: Give a level of detail as necessary to explain the concept to a human (not a computer).

9. (6 pts) Suppose you want to build a triangle out of blocks. If the triangle has two rows there will be two blocks on the bottom and one block on top (a total of 3 blocks). If there are three rows the bottom row will have 3 blocks on the bottom row, then 2, then 1 (a total of 6 blocks). Write a function called triangle that accept a position integer $n$ as a parameter representing the number of rows and will recursively calculate the number of blocks needed to construct a triangle with $n$ rows.

10. (6 pts) What is returned by the following recursive method it is invoked like this: myfunc(6)? Show your work.

   ```java
   public static int myfunc(int n) {
     if (n==0) return 10;
     if (n<0) return -1;
     return myfunc(n/2) + myfunc(n-4);
   }
   ```

11. Suppose that the following declarations are used to represent airline reservations for a single plane in the form of a singly-linked list.

   ```java
   class ListNode {
     public int data;
     public ListNode next;
   }
   class MyList {
     public ListNode head;
     public MyList() { head= null; }
     public void insert(int data) { ... }
   }
   ```
public void display() { ... }

(a) (8 pts) Suppose that the MyList method called insert implements an unordered insert. Add a method to MyList called merge that given another MyList object as a parameter will combine the two lists. The MyList object passed as a parameter should not be changed. Keep in mind that either list could be empty.

(b) (8 pts) Add a method to MyList called createList that accepts a positive integer, n, as a parameter. The method should create a new linked list with n elements with the data of each element being the next odd number (starting with 1). So, the first element will have a data value of 1, the next 3, the next 5, etc. The attribute head should point to the beginning of the new list.