Recursion

Q1. Consider the following code.

```
unsigned long func(int a, unsigned int n) {
  if (n == 0 )
      return 1;
  else if( n == 1)
      return a;
  else if (n%2 == 0)
      return func(a, n/2)*func(a, n/2);
  else
      return a*func(a, n/2)*func(a, n/2);
}
```

A. What is the above code computing?

a^n

B. What is the time complexity of this code.

O(n)

C. The above code has inefficiencies. What can you do to improve its time complexity?

```
func(a, n/2) is being computed twice.
Instead of func(a, n/2)*func(a, n/2), do the following:
res = func(a, n/2);
return res * res;

Time complexity : O(log(n))
```

Q2. Consider the following code.

```
long add(int m) {
  int i;
  long sum=0;
  for (i=0; i<m; i++) {
      sum += i; }
  return sum;
}
long RecCompute(int n) {
  int p;
  if (n==1) return n;
```

```
  else {
    RecCompute(n/2);
    p = n/4;
    return add(p); }
}
```

What is the time complexity of the above code.

O(n)

Graphs

Q3.

1. The time and space complexity of an algorithm does not depend on the choice of the data structure for the problem.	TRUE / FALSE
2. The tree constructed by Breadth-First-Search algorithm is unaffected by the order in which the vertices are explored.	TRUE / FALSE

Q4.

Given an undirected graph **`G(V,E)`** with n vertices and m edges. How long does it take to **delete** an edge (u,w) from G if an adjacency **list** is used to represent the graph?

O(degree(u) + degree(w))

Give an example of a sparse graph where number of edges are linear in the number of vertices.

A tree.

Given a directed graph **`G(V,E)`** with n vertices and m edges. Suppose graph G is represented as an **adjacency list**. I want to find the number of **in-coming** edges for each of the vertices. What is the time complexity for finding that?

Time Complexity in Big-Oh notation ____O(m+n)________
Time to traverse the adjacency list.

Q5. Suppose you are given a **connected, undirected, weighted graph** G(V,E) whose **edges all have the same positive weight**. How will you find the shortest path between two vertices 'u' and 'v' in G?

Since all edges have the same weight, BFS will return the shortest path.

Q6. Your friend ran the Prim's algorithm on an undirected, weighted graph G and constructed a minimum spanning tree. Your friend claims that the minimum spanning tree is a bipartite graph. Do you agree?

YES or NO? ____YES________
Brief explanation of your answer.

All trees are bipartite, because they have no cycles.

Q7. Your friend claims that he/she can draw an n-node **tree** of height greater than `O(log(n))`. Do you agree with his/her claim?

If AGREE, then draw an example of such a tree. If DISAGREE then prove it.
AGREE / DISAGREE ____AGREE__________