CS 50010 Assignment 2

Due Monday the 26th in class

Partial credit is only awarded if you show your work

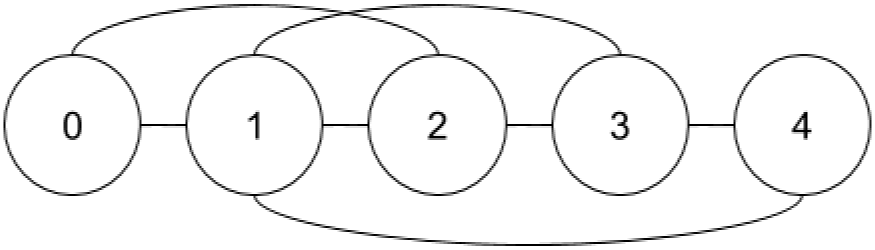
Recursion

For each of the following problems, find the recurrence relation and big theta runtime of the function using the Master Theorem. If the Master Theorem does not apply, state why it can not be used. If the master theorem does not work you do not need to provide a big theta runtime.

```
1. bool foo(int* arr, int p1, int p2, int a)
   {
         int m = (p1 + p2)/2;
         if(p1 == p2)
               Return false;

         if(arr[m] == a) return true;
         else if(arr[m] > a)
               return foo(arr, p1, m, a);
         else
               return foo(arr, m, p2, a);
   }

2. int bar(int* arr, int p1, int p2)
   {
         return 1 + bar(arr, p1+1, p2)+bar(arr, p1, p2-1);
   }

3. void four_way_mergesort(int* arr, int start, int end)
   {
         int x1 = (start) + ((start+end)/4);
         int x2 = (start) + 2*((start+end)/4);
         int x3 = (start) + 3*((start+end)/4);
         four_way_mergesort(arr,start,x1);
         four_way_mergesort(arr,x1,x2);
         four_way_mergesort(arr,x2,x3);
         four_way_mergesort(arr,x3,end);

         four_way_merge(arr,start,x1,x2,x3,end); //assume linear time
}
```

Graphs and Trees

1. Below is an undirected, unweighted graph, G. Show the adj. matrix and adjacency list representations of G.



2. An undirected *n* vertex graph G = (V,E) is called a *tree + c* if G is connected and |E| = *n* + *c*. Assume G is an *n* + 6 graph with weights on the edges represented as an adjacency list. Describe an *O(n)* time algorithm for finding a minimum spanning tree in a weighted *n* + 6 graph. (pseudocode is fine)

Searching in Graphs and Trees

1. You are given a directed acyclic graph (DAG) with weighted edges. The *diameter* of a graph is the length of the longest path from any *u* to *v (*assuming you are taking the shortest path from *u* to *v*). Provide an algorithm to find the diameter of a DAG. (Hint: Try modifying the weights to make this into an easier problem)

2. You are trying to write an algorithm that generates random 2d-grid mazes, the style found in newspapers, children's’ menus, etc… Describe how you can generate these mazes by creating a graph and using some graph search algorithm. Provide pseudocode for your algorithm, and analyze the time and space complexity

Trees and Hashing

1. You are given a list arranged in no particular order. Using a binary search tree and traversals, provide a way to sort this list. Analyze the runtime for your algorithm.
2. A commonly used hash function for strings is to treat the characters as integers and sum them up. However this is a poor idea for situations where collision resistance is desired. Demonstrate this by finding a collision for “The quick brown fox jumps over the lazy dog”. Assume you are using ASCII values for the characters.