**CS 180** Homework 2

**Problem 1**

1. To prove the existence of a king, we start by choosing a node from the independent set S in the tournament graph with the highest outdegree, meaning it reaches out to the most nodes. This node is the king. If we suppose there existed another node that could dominate this king, then by transitivity, it would dominate the nodes that the king can reach in one step and therefore reach more nodes than the king. However this is a contradiction and would disprove our original choice of the node with the highest outdegree. The node we originally chose then is the king.

function findKing(graph)

max = 0

result = null

for node in graph

if node.outDegree > max

max = node.outDegree

result = node

return result

1. This algorithm iterates through each node of the graph so it runs in O(n) time complexity.

**Problem 2**

**Problem 3**

For an array of n values where array[0] and array[n-1] are +infinity, the algorithm to find a local minimum would be as follows:

function findLocalMin(array, start, end) // 0 is the start index, n-1 is the end index

mid = start + end / 2 // use integer division here

if array[mid-1] > array[mid] && array[mid+1] > array[mid]

return array[mid] // found local minimum

if array[mid-1] < array[mid] // left half is smaller, local min will be there

findLocalMin(array, start, mid)

else

findLocalMin(array, mid, end)

By halving the array each time a local minimum isn’t found, this algorithm performs a binary search and costs O(log n) time.

**Problem 4**

Given an unsorted array A of size n, and a heap data structure, this would be an O(n log n) algorithm to sort A:

function heapSort(A, n)

end = n - 1

while end > 0

swap(A[end], A[0])

end = end - 1

buildHeap(A, 0, end)

function buildHeap(A, start, end)

root = start

while root \* 2 + 1 ≤ end

child = root \* 2 + 1

swap = root

if A[swap] < A[child]

swap = child

if child+1 ≤ end && A[swap] < A[child+1]

swap = child + 1

if swap == root

return

else

swap(A[root], A[swap])

root = swap