Universidade Estadual de Maringá
Departamento de Informática
Projeto e Análise de Algoritmos / Prof. Daniel Kikuti

Aluno(a): ______________________________

Primeiro trabalho (Parte 3)

1. O i-ésimo menor elemento de um conjunto de n elementos é chamado de *i-ésima estatística de ordem*. Por exemplo, o mínimo de um conjunto de elementos é a primeira estatística de ordem ($i = 1$), e o máximo é a n-ésima estatística de ordem ($i = n$). Dado um conjunto A de n números (distintos) e um número i, com $1 \leq i \leq n$, definimos o *problema de seleção* como sendo o problema de encontrar o elemento $x \in A$ que é maior que exatamente $i - 1$ outros elementos de A. Este problema pode ser resolvido no tempo $O(n \lg n)$, pois podemos ordenar os números usando o Mergesort (por exemplo) e então indexar o i-ésimo elemento no vetor de saída. Outra forma de fazer isto é usando o algoritmo descrito a seguir, sendo `RANDOMIZED-PARTITION` o mesmo algoritmo usado no Quicksort aleatório. Argumente (informalmente, mas de maneira precisa) por que o algoritmo funciona. Faça a análise de complexidade do algoritmo para o melhor caso e para o pior caso (use a notação assintótica).

```
RANDOMIZED-SELECT(A,p,r,i)
if p == r
   return A[p]
q = RANDOMIZED-PARTITION(A,p,r)
k = q - p + 1
if i == k      //O valor pivô é a resposta
   return A[q]
else if i < k
   return RANDOMIZED-SELECT(A,p,q-1,i)
else return RANDOMIZED-SELECT(A,q+1,r,i-k)
```

2. Considere o algoritmo `heapsort` descrito a seguir.
   (a) Argumente que o algoritmo está correto usando o seguinte invariante de laço: "No começo de cada iteração do laço **for** das linhas 2–5, o subvetor $A[1..i]$ contém os i menores elementos de $A[1..n]$, e o subvetor $A[i+1..n]$ contém os $n - i$ maiores elementos de $A[1..n]$ em ordem.
   (b) É verdade que para qualquer entrada o algoritmo `heapsort` tem comportamento $O(n \lg n)$? Justifique.

```
heapsort(A)
1 build-max-heap(A)
2 for i = A.comprimento downto 2
3   troca(A[1], A[i])
4   A.tamanho-do-heap = A.tamanho-do-heap - 1
5   max-heapify(A, 1)
```