UNIVERSIDADE FEDERAL DE SERGIPE
CENTRO DE EXATAS E TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO
PROFA. LEILA MACIEL DE ALMEIDA E SILVA

LISTA OBRIGATÓRIA PARCIAL DE PROJETO E ANÁLISE DE ALGORITMO

SEMANAS 3 e 4

1. Considere duas soluções *s* e *t* para resolver um dado problema. A complexidade das duas soluções é dada a seguir. Determine qual das duas soluções é mais eficiente para grandes valores de *n*. Na sua solução você deve utilizar o método visto nas notas de aula e o Teorema Mestre para calcular as recorrências das soluções *s* e *t*, respectivamente.
   solução *s*: $T(n) = 2T(n/2) + n$, $T(1) = 0$
   solução *t*: $T(n) = 2T(n/4) + n^{1/2}$, *para* $n > 2$ *e para* $n \leq 2$, $T(n)=c=O(1)$.

Para os problemas a seguir faça:

a) Descreva a ideia de sua solução;
b) Elabore o algoritmo em pseudo-linguagem;
c) Calcule a complexidade do algoritmo para o pior caso;
d) Implemente o algoritmo e teste-o.

2. Seja $X = [x_1, x_2, ..., x_{n-1}, x_n]$ um vetor de inteiros, ordenados em ordem crescente. Uma **rotação** em *X* consiste no deslocamento dos elementos de *X* de tal forma que o vetor rotacionado *X´* fica na forma $X' = [x_n, x_1, x_2, ..., x_{n-1}]$. Por exemplo, para *X=[2,4,7,9,15]* uma rotação em *X* resultaria no vetor *[15,2,4,7,9]* e três rotações em *X* resultaria no vetor *[7,9,15,2,4]*.

   Seja *A* um vetor de inteiros, ordenados em ordem crescente e sem elementos repetidos. Seja *R* um vetor que sofreu *r* rotações a partir de *A*, $0 \leq r \leq n-1$. Elabore um algoritmo para dado somente o vetor *R*, determinar quantas rotações *R* sofreu. A complexidade de seu algoritmo deve ser $O(\log n)$. Ex: Para *R = [7,9,15,2,4]* a resposta seria *3*. Para *R = [2,4,7,9,15]* a resposta seria *0*.

3. Considere uma função monotonicamente crescente *f(x)*, cujo domínio é o conjunto de inteiros não negativos. Elabore um algoritmo para encontrar o menor valor de *x* para o qual $f(x) \geq 0$. Por exemplo, para a função $f(x) = 2x - 35$, definida para $x \geq 0$, a resposta seria $x = 16$, pois de $x = 0$ até $x = 15$, $f(x) < 0$. A sua solução deve ter complexidade $O(\log x)$. Na implementação da sua solução você deve adotar uma função monotonicamente crescente fixa, de sua escolha. (Dica: leia a Seção 6.2 do Udi Manber para se inspirar).

4. Dado um vetor de inteiros distintos, em ordem arbitrária, elabore duas soluções *in loco* conceitualmente distintas para rearranjar o vetor de tal forma que todos os elementos que ocupem posições pares sejam maiores que os seus vizinhos no vetor. Uma solução é *in loco*, ou no local, quando não gasta espaço adicional, a menos de poucas variáveis simples. Qual solução é mais eficiente? Por exemplo, para o vetor *X=[9,5, 8, 2, 6]*, seu algoritmo poderia retornar *X=[2, 9, 5, 8, 6]* ou *X=[5, 9, 2, 8, 6]*.

5. Dado um vetor de inteiros, elabore duas soluções de complexidade distintas para encontrar o maior produto de dois elementos do vetor. Por exemplo, para o vetor *X= [5, -10, 4, 6, -2, -3]*, o maior produto é 30, resultante da multiplicação do par *(-10, -3)* ou

do par *(5, 6)*. Nenhuma de suas soluções pode ser quadrática. Compare as soluções obtidas em relação às complexidade de tempo e espaço.

6. Considere um vetor de inteiros, com alguns elementos nulos. Elabore soluções *in loco* de tempo *O(n)* para organizar o vetor de forma que os zeros sejam deslocados para o final do vetor.
   (a) A sua solução deve ser estável, ou seja, a ordem relativa de entrada dos números é preservada no vetor de saída.
   (b) A sua solução não é estável, ou seja, organiza os não zeros e depois os zeros sem se preocupar com a ordem de entrada original.

   Para o vetor *X = [6, 0, 8, 2, 3, 0, 4, 0, 1]*, no item (a) sua solução retornaria *X = [6, 8, 2, 3, 4, 1, 0, 0, 0]* e em (b) poderia retornar, por exemplo, *X=[6, 1, 8, 2, 3, 4, 0, 0, 0]*.