Universidade Tecnológica Federal do Paraná (UTFPR)
Departamento Acadêmico de Informática (DAINF)

Estruturas de dados II

Professora: Juliana de Santi (jsanti@utfpr.edu.br)

Lista de exercícios

1) Relembrando EDI: Qual a complexidade de tempo (pior caso) dos seguintes algoritmos clássicos: busca-binária, busca-linear, selection-sort, insertion-sort, bubble-sort, merge-sort, quick-sort, inserir um elemento em uma pilha, algoritmo clássico de multiplicação de matrizes.

2) Coloque as funções a seguir em ordem crescente assintoticamente:

1. $7n + \sqrt{n}$
2. $n^2 - 2n + 5$
3. $\lg n^2$
4. $3n\lg n + 5n$
5. $an^3 + bn^2 - cn + 2$
6. e^n
7. $3^n + \lg n + n^3$
8. $5n^2\lg n + n^2 - n + 1$
9. 3500000000
10. 2^{n^2}
11. $4^{\lg n}$
12. $2^{\lg n}$

3) Os algoritmos W, X, Y e Z possuem tempo de execução no pior caso de $20nlog_{10}n$, $5n^2$, $0.005n^3$ e $500n$, respectivamente. Responda as seguintes questões:

- Qual a notação assintótica destes quatro algoritmos?
- Utilizando a resposta da questão anterior, qual o ordem destes quatro algoritmos (do melhor para o pior).
- Utilizando o custo exato de cada algoritmo (não na forma assintótica), qual o ordem destes quatro algoritmos, do melhor para o pior, para 30 elementos?
- Utilizando o custo exato de cada algoritmo (não na forma assintótica), qual o ordem destes quatro algoritmos, do melhor para o pior, para 100.000 elementos?

4) O que fazem os algoritmos abaixo. Calcule a complexidade de cada um deles em função de n.

```
Algoritmo1
1. s = 0;
2. for (i=1;i<=n;i++)
3.    s = s+i;
```

```
Algoritmo2
1. s = 0;
2. for(i=1;i<=n;i++)
3.    for(j=1;j<=i;j++)
4.       s = s + 1;
```

```
Algoritmo3
1. s = n*(n+1)/2;
```

5) Considere o seguinte fragmento de código:

```
int d = MAX;
for (i = 0; i < n; i++) {
   for (j = 0; j < n; j++) {
      if ((i != j) && abs(V[i] - V[j]) < d) {
         d = abs(V[i] - V[j]);
      }
   }
}
return d;
```

Descreva o que ele faz e qual a complexidade utilizando a notação assintótica. Descreva se existe algum modo de melhorar a complexidade.

6) Considere o seguinte código abaixo. Descreva o que ele faz e qual a complexidade utilizando a notação assintótica.

```
#include <stdio.h>
#include <string.h>

/* Funcao que troca o conteudo indicado pelos ponteiros x e y.*/
void troca (char* x, char* y)
{
    char temp;
    temp = *x;
    *x = *y;
    *y = temp;
}

void func (char* a, int l, int r)
{
    int i;
    if (l == r)
        printf("%s\n", a);
    else {
```

```
        for (i = l; i <= r; i++) {
            troca((a + l), (a + i));
            func(a, l + 1, r);
            troca((a + l), (a + i));
        }
    }
}

int main()
{
    char str[] = "ABC";
    int n = strlen(str);
    func(str, 0, n - 1);
    return 0;
}
```

7) Sejam as funções:

- $f(n) = n + \lg n$
- $g(n) = 5$
- $h(n) = n \lg n$
- $l(n) = n^2 + h(n)$.

Marque V ou F.

( ) $f(n) \notin O(l(n))$
( ) $g(n) \in \Omega(f(n))$
( ) $h(n) \notin O(n^2)$
( ) $l(n) \in \Omega(h(n))$
( ) $g(n) \in O(1)$
( ) $l(n) \notin \Theta(n^2)$