Atividade 04 – Análise e Complexidade de Algoritmos

Entrega: AVA (on-line)
Formato: documento Word / PDF
Data Limite: 18/05/2025
INDIVIDUAL

1-) Dados os algoritmos a seguir, mostre sua função completa e sua ordem assintótica resultante:

a-)

```
void func(int n) {
  int i;
  for (i = 0; i < n; ++i)
    printf("%d", i);
}
```

b-)

```
void func(){
   int index = 5;
   int item = list[index];
   if (item > 10) then
      item += 50;
   else
      item += 60;
   for (int i = 1; i < 100; i++)
      for (j = 1; j < 200; j++)
         item += i + j;
}
```

c-)

```
void func(int ar[])
{
   for (int i = (ar.length - 1); i >= 0; i--)
   {
      for (int j = 1; j ≤ i; j++)
      {
         if (ar[j-1] > ar[j])
         {
              int temp = ar[j-1];
              ar[j-1] = ar[j];
              ar[j] = temp;
   } } } }
```

d-)

```
int func(int n){
   min = 9999;
   for (int a=0; a < n; a++)
      for (int b=0; b < n; b++)
         for (int c=0; c < n; c++)
            if ((5a+4b+c == n) && (a+b+c < min))
               min = a+b+c;
   return min;
}
```

e-)

```
void func(int n) {
  int i;
  for (i = 1; i < n; i=i*2)
    printf("%d", i);
}
```

f-)

```
void func(int[] ar)
{
   for (int i=1; i ‹ ar.length; i++)
   {
      int index = ar[i]; int j = i;
      while (j > 0 && ar[j-1] > index)
      {
          ar[j] = ar[j-1];
          j--;
      }
      ar[j] = index;
} }
```

g-)

```
void func(int[] array, int targetValue) {
  for (int guess=0; guess < array.length; guess++) {
    if (array[guess] == targetValue) {
       return guess;  // found it!
    }
  }
  return -1;  // didn't find it
};
```

h-)

```
def func(matrix):
    n = len(matrix)
    for i in range(n):
        for j in range(n):
            for k in range(n):
                matrix[i][j] += matrix[j][k]
```

i-)

```
void func(int n) {
  int i;
  for (i = n; i>=1 ; i=i/2)
    printf("%d", i);
}
```

j-)

```
def func(A, B):
    dicionario_A = {x: True for x in A}  # Conjunto A em um dicionário
    dicionario_B = {x: True for x in B}  # Conjunto B em um dicionário

    for key in dicionario_A:  # Acessa cada elemento de A
        print(f"Elemento de A: {key}")

    for key in dicionario_B:  # Acessa cada elemento de B
        print(f"Elemento de B: {key}")
```

k-)

```
int func(int n){
   min = 0;
   for (int a=0; a < n; a++)
      for (int b=0; b < n; b++)
         min = calculate(a, b, n, min);
   return min;
}

int calculate(int x, int y, int z, int valor){
   for (int c=0; c < z; c++)
      if ((5x+4y+z == n) && (x+y+z < valor))
         valor = x + y + z;
      else
         valor = x – y – z;
   return valor;
}
```

l-)

```
def func(arr):
    n = len(arr)
    for i in range(n):
        for j in range(n):
            arr[i] += arr[j]
```

m-)

```
def func(arr):
    soma = 0
    for x in arr:
        soma += x
    return soma
```

n-)

```
def func(n):
    while n > 1:
        n = n // 2
```

o-)

```
def func(arr, k=0):
    if k == len(arr):
        print(arr)
    else:
        for i in range(k, len(arr)):
            arr[k], arr[i] = arr[i], arr[k]
            func(arr, k + 1)
            arr[k], arr[i] = arr[i], arr[k]
```

p-)

```
def func(arr):
    if len(arr) <= 1:
        return arr
    mid = len(arr) // 2
    esquerda = func(arr[:mid])
    direita = func(arr[mid:])
    return merge(esquerda, direita)

def merge(esq, dir):
    resultado = []
    i = j = 0
    while i < len(esq) and j < len(dir):
        if esq[i] < dir[j]:
            resultado.append(esq[i])
            i += 1
        else:
            resultado.append(dir[j])
            j += 1
    resultado.extend(esq[i:])
    resultado.extend(dir[j:])
    return resultado
```

q-)

```
def func(n):
    if n == 0:
        return 1
    return func(n - 1) + func(n - 1)
```

r-)

```
def func(lista1, lista2):
    for x in lista1:
        print(x)
    for y in lista2:
        print(y)
```

s-)

```
def func(n):
    if n == 0:
        return 1
    return func12(n - 1) + func12(n - 1) + func12(n - 1)
```

t-)

```
void func(int[] ar){
   for (int i = 0; i ‹ ar.length-1; i++)
   {
      int min = i;
      for (int j = i+1; j ‹ ar.length; j++)
            if (ar[j] ‹ ar[min]) min = j;
      int temp = ar[i];
      ar[i] = ar[min];
      ar[min] = temp;
} }
```

2-) É verdade que $n^3-999999n^2-1000000 = O(n^2)$? Justifique.

3-) Podemos classificar um determinado problema com complexidade O(log n) como P, NP ou NP-Completo. Justifique sua resposta.

Grupo Anchieta
Prof. Clayton Valdo