Nome: Flávio Eduardo Oliveira e Silva
Matrícula: 2017018013

Exercício 02 - Algoritmos Recursivos

1 - Considere a seguinte função g, definida no conjunto dos números naturais, da seguinte forma:

$g(0) = 0$, $g(1) = 1$ e,

$g(n) = 5g(n-1) – 6g(n-2)$, para $n \geq 2$.

a)Escreva um algoritmo que computa g:

```
   g(n)
    se(n=0 ou n=1)
          então retorna n
          senão retorna 5*g(n-1) - 6*g(n-2)
```

b)Escreva e resolva uma recorrência para o tempo T(n) consumido pela função g(n), determine seu comportamento assintótico

0	$\mathbf{2^0.K}$
1	$\mathbf{2^1.K}$
2	$\mathbf{2^2.K}$
3	$\mathbf{2^3.K}$
...	
n-1	$\mathbf{2^{n-1}.K}$
n	$\mathbf{2^n.K}$

Tomando K como uma constante:

$\mathbf{T(n) = 2^0.K + 2^1.K + ... 2^n.K}$

$\mathbf{T(n) = (2^0.K + 2^n.K)* n/2}$

$\mathbf{T(n) = (K + 2^n.K)* n/2}$

Comportamento Assintótico: $\mathbf{O(2^{n)}}$

2 -

Max(A,e,d)

```
1   se e=d                              n+(n-1)
2         então devolve A[e]            n
3         senão x<- ⌊(e+d)/2⌋           (n-1)
4               a<- Max(A, e, x)        (n-1)
5               b<- Max(A, x+1, d)      (n-1)
6               se a > b                (n-1)
7                     então devolve a   0
8                     senão devolve b   1
```

Seja C(n) o número de vezes que a comparação da linha 6 é executada em uma chamada de Max(A, e, d), onde $n=d – e + 1$.

Tomando como o último elemento do vetor sendo o maior, a linha 7 não será executada nenhuma vez.
Para n=1, a execução é encerrada na linha 2, portanto, para n>1, a linha 6 será executada **(n-1)** vezes. Portanto:

C(1) = 0, para n=1;
C(n) = n-1, para n>1

3 -

Pior caso iterativo

```
Inserção(A,n)
1    para j<-2 até n faça n                 n
2         x<- A[j]                          n-1
3         i<- j-1                           n-1
4         enquanto i>0 e A[i]>x faça        (n+2).(n-1)/2
5              A[i+1] <- A[i]               n.(n-1)/2
6              i<- i-1                      n.(n-1)/2
7    A[i+1]<- x                             n-1
```

F(n) = n+3(n-1) + 2[(n).(n-1)/2] + (n+2).(n-1)/2
...
F(n) = $3n^2/2$ + 7.n/2 - 4

Comportamento Assintótico: $O(n^2)$

Melhor caso iterativo

```
Inserção(A,n)
1    para j<-2 até n faça n                 n
2         x<- A[j]                          n-1
3         i<- j-1                           n-1
4         enquanto i>0 e A[i]>x faça        n-1
5              A[i+1] <- A[i]               0
6              i<- i-1                      0
7    A[i+1]<- x                             n-1
```

F(n) = n + 4(n-1)
F(n) = n + 4n - 4
F(n) = 5n - 4

Comportamento Assintótico: O(n)

Pior caso recursivo

```
Inserção(A,e,d)
1    se e<d                                 1
2         então Inserção(A,e,d-1            F(n-1)
3              x<- A[d]                     1
4              i<- d-1                      1
5              enquanto i>=e E A[i] > x faça     n
6                   A[i+1] <- A[i]          n-1
7                   i<- i-1                 n-1
8    A[i+1]<- x                             1
```

$F(1) = 1$, para $n=1$
$F(n) = F(n-1) + 3n + 2$, para $n > 2$

$F(n) = F(n-1) + 3n + 2$
$F(n) = F(n-2) + [3(n-1) + 2] + [3n + 2]$
$F(n) = F(n-3) + [3(n-2) + 2] + [3(n-1) + 2] + [3n + 2]$
...
$F(n) = 3n2 /2 + 7n/2 – 4$

Comportamento Assintótico: $O(n^2)$

Melhor caso recursivo

```
Inserção(A,e,d)
se e<d                                      1
      então Inserção(A,e,d-1                F(n-1)
            x<- A[d]                        1
            i<- d-1                         1
            enquanto i>=e E A[i] > x faça      n
                  A[i+1] <- A[i]            0
                  i<- i-1                   0
A[i+1]<- x                                  1
```

$F(1) = 1$, para $n=1$
$F(n) = F(n-1) + n + 4$, para $n > 2$

$F(n) = F(n-1) + n + 4$
$F(n) = F(n-2) + [(n-1) + 4] + [n + 4]$
$F(n) = F(n-3) + [(n-2) + 4] + [(n-1) + 4] + [n + 4]$
...
$F(n) = n 2 /2 + 9n/2 – 4$

Comportamento Assintótico: $O(n^2)$

Com base nos cáculos, tem-se que tanto no pior caso iterativo quanto no recursivo, o algoritmo possui o mesmo comportamento, de $O(n^2)$.

Porém no melhor caso, o algoritmo iterativo leva mais vantagem, pois seu comportamento é de $O(n)$, enquanto que o recursivo é de $O(n^2)$.