```
 ___________________________
| Diogo Araujo Miranda     |
| Matricula: 705657        |
|__________________________|
```

Exercicio (1)

```
// Verificar o menor e o maior elemento do array
int maior = 0;
int menor = 0;
if(array[0] > array[1])
{
      maior = array[0];
      menor = array[1];
} else {
      maior = array[1];
      menor = array[0];
}

for(int i = 2; i < n; i++)
{
      if(array[i] > maior)
            maior = array[i];
      else if(array[i] < menor)
            menor = array[i];
}
```

Análise de Complexidade do melhor caso (Comparação entre elementos do array)

Melhor caso - Se o array tiver em ordem crescente:
1 + n-2 = n-1 = ordem de teta(n)

Pior caso :
1 + 2(n-2) = 1 + 2n-4 = 2n-3 = ordem de teta(n)

Caso medio :
n-1 + 2n-3 / 2 = 3n-4 / 2 = ordem de teta(n)

Exercicio (2)

Programa A.1: Função para obter o maior elemento de um Array

```
int Max(Vetor A)
{
      int i, Temp;

      Temp = A[0];
      for(i = 1; i < n; i++) {
            if(Temp < A[i]) Temp = A[i];
      }
} // Max
```

Programa A.2 : Função para obter o maior e o menor elemento de um Array

```
void MaxMin1(Vetor A, int *Max, int *Min)
{
      int i;
      *Max = A[0];
      *Min = A[0];
      for(i = 1; i < n; i++) {
            if (A[i] > *Max) *Max = A[i];
            if (A[i] < *Min) *Min = A[i];
```

```
      }
}
```

Programa A.3 : Implementação melhorada para obter o maior e o menor elemento de um Array

```
void MaxMin2(Vetor A, int *Max, int *Min)
{
      int i;

      *Max = A[0];
      *Min = A[0];
      for(i = 1; i < n; i++) {
            if (A[i] > *Max)
                  *Max = A[i];
            else if (A[i] < *Min)
                  *Min = A[i];
      }
}

void MaxMin3(Vetor A, int *Max, int *Min)
{
      int i, FimDoAnel;

      if( (n & 1) > 0) {
            A[n] = A[n-1];
            FimDoAnel = n;
      } else
            FimDoAnel = n-1;
      if( A[0[] > A[1] ) {
            *Max = A[0];
            *Min = A[1];
      } else {
            *Max = A[1];
            *Min = A[0];
      }
      i = 3;
      while (i <= FimDoAnel) {
            if(A[i-1] > A[i]) {
            if( A[i-1] > *Max)
                  *Max = A[i-1];
            if( A[i < *Min)
                  *Min = A[i];
            } else {
              if (A[i-1] < *Min)
                        *Min = A[i-1];
               if (A[i] > *Max)
                        *Max = A[i];
            }
            i += 2;
      }
}

void MaxMin4(int Linf, int Lsup, int *Max, int *Min)
{
      int Max1, Max2, Min1, Min2, Meio;

      if(Lsup - Linf <= 1) {
            if (A[Linf-1] < A[Lsup - 1]) {
                  *Max = A[Lsup - 1];
                  *Min = A[Linf - 1];
            } else {
                  *Max = A[Linf - 1];
                  *Min = A[Lsup - 1];
```

```
            }
            return;
      }
      Meio = (Linf + Lsup) / 2;
      MaxMin4(Linf, Meio, &Max1, &Min1);
      maxMin4(Meio + 1; Lsup, &Max2, &Min2);
      if(Max1 > Max2)
            *Max = Max1;
      else
         *Max = Max2;
      if(Min1 < Min2)
            *Min = Min1;
      else
            *Min = Min2;
} // MaxMin4
```

Exercício 3)

	$O(1)$	$O(\lg n)$	$O(n\lg(n))$	$O(n^2)$	$O(n^3)$	$O(n^5)$	$O(n^{20})$
$F(n) = \lg(n)$	F	V	V	V	V	V	V
$F(n) = n*\lg(n)$	F	F	V	V	V	V	V
$F(n) = 5n+1$	F	F	F	V	V	V	V
$F(n) = 7n^5-3n^2$	F	F	F	F	F	V	V
$F(n) = 99n^3-1000n^2$	F	F	F	F	V	V	V
$F(n) = n^5 - 9999n^4$	F	F	F	F	F	V	V

Exercicio 4)

	$\Omega(1)$	$\Omega(\lg n)$	$\Omega(n\lg(n))$	$\Omega(n^2)$	$\Omega(n^3)$	$\Omega(n^5)$	$\Omega(n^{20})$
$F(n) = \lg(n)$	V	V	F	F	F	F	F
$F(n) = n*\lg(n)$	V	V	V	F	F	F	F
$F(n) = 5n+1$	V	V	V	F	F	F	F
$F(n) = 7n^5-3n^2$	V	V	V	V	V	V	F
$F(n) = 99n^3-1000n^2$	V	V	V	V	V	F	F
$F(n) = n^5 - 9999n^4$	V	V	V	V	V	V	F

Exercicio 5)

	$\Theta(1)$	$\Theta(\lg n)$	$\Theta(n\lg(n))$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^5)$	$\Theta(n^{20})$

F(n) = lg(n)	F	V	F	F	F	F	F
F(n) = n*lg(n)	F	F	V	F	F	F	F
F(n) = 5n+1	F	F	F	F	F	F	F
F(n) = $7n^5-3n^2$	F	F	F	F	F	V	F
F(n) = $99n^3-1000n^2$	F	F	F	F	V	F	F
F(n) = $n^5 - 9999n^4$	F	F	F	F	F	V	F

Exercicio 6)

$f(n) = 3n^2- 5n - 9$
$g(n) = n*lg(n)$
$l(n) = n.lg^2(n)$
$h(n) = 99n^8$

a) $f(n) + g(n) - h(n) = \theta(n^8)$

B) $O(f(n) + O(g(n)) - O(h(n)) = O(n^8)$

C) $3n^2- 5n - 9 \times n*lg(n) = 3n^2*n = 3n^3 = O(n^3)$

D) $n*lg(n) \times n.lg^2(n) \times 99n^8 = n*n*n^8 = O(n^{10})$

E) $3n^2- 5n - 9 \times n*lg(n) \times n.lg^2(n) = n^2 \times n \times n = O(n^4)$

F) $O(n^2)$

Exercicio 7)

- Mostre um valor c e outro m tal que, para n ≥ m, |3n 2 + 5n +1| ≤ c x |n 2 |, provando que 3n 2 + 5n +1 é O(n 2)

m = 6 e c = 4, pois

n	$3n^2+5n+1$	$4*n^2$
1	9	4
2	23	16
3	43	36
4	69	64
5	101	100
6	139	144
7	183	196
8	233	256
9	289	324
10	351	400

- Mostre um valor c e outro m tal que, para n ≥ m, |3n 2 + 5n +1| ≤ c x |n 3 |, provando que $3n^2 + 5n +1$ é $O(n^3)$

m = 5 e c = 1, pois

n	$3n^2+5n+1$	$4*n^2$
1	9	1
2	23	8
3	43	27
4	69	64
5	101	125

6	139	216
7	183	343
8	233	512
9	289	729
10	351	1000
11	419	1331
12	493	1728
13	573	2197
14	659	2744
15	751	3375

- Prove que $3n^2 + 5n + 1$ não é O(n)

Qualquer constante multiplicada a n, teremos algum valor de n que irá ultrapassar o limite superior de O(n)

Exemplo

c = 50

n	3^2+5n+1	50n
1	9	50
2	23	100
3	43	150
4	69	200
5	101	250
6	139	300
7	183	350
8	233	400
9	289	450
10	351	500
11	419	550
12	493	600
13	573	650
14	659	700
15	751	750

Em n = 15, o valor de f(n) já ultrapassa o valor de O(n)

Exercicio 8)

- Mostre um valor c e outro m tal que, para $n \geq m$, $|g(n)| \geq c \times |f(n)|$, provando que $3n^2 + 5n + 1$ é $\Omega(n^2)$

c = 2 e m = 1, pois

n	$3n^2+5n+1$	$2n^2$	
1	9	2	<-------
2	23	8	
3	43	18	
4	69	32	
5	101	50	
6	139	72	

- Mostre um valor c e outro m tal que, para $n \geq m$, $|g(n)| \geq c \times |f(n)|$, provando que 3n 2 + 5n +1 é $\Omega(n)$

m = 15 e c = 50, pois

n	$3n^2+5n+1$	50n
1	9	50
2	23	100
3	43	150
4	69	200

5	101	250
6	139	300
7	183	350
8	233	400
9	289	450
10	351	500
11	419	550
12	493	600
13	573	650
14	659	700
15	751	750 <------
16	849	800

- Prove que $3n^2 + 5n + 1$ não é $\Omega(n^3)$

Pois para qualquer valor de c positivo, o valor de n >= m estara abaixo do limite inferior definido

n	$3n^2+5n+1$	$1*n^3$
1	9	1
2	23	8
3	43	27
4	69	64
5	101	125 <----
6	139	216
7	183	343
8	233	512
9	289	729
10	351	1000
11	419	1331
12	493	1728
13	573	2197
14	659	2744

Exercicio 9)

- Mostre um valor para c_1, c_2 e m tal que, para $n \geq m$,
$c_1 \times |f(n)| \leq |g(n)| \leq c_2 \times |f(n)|$, provando que $3n^2 + 5n + 1$ é $\theta(n^2)$

c1 = 2 e c2 = 4 e m = 8, pois

n	$3n^2+5n+1$	$4n^2$	n	$3n^25+n+1$	$2n^2$
1	9	4	1	9	2
2	23	16	2	23	8
3	43	36	3	43	18
4	69	64	4	69	32
5	101	100	5	101	50
6	139	144	6	139	72
7	183	196	7	183	98
8	233	256	8	233	128
9	289	324	9	289	162
10	351	400	10	351	200
11	419	484	11	419	242
12	493	576	12	493	288
13	573	676	13	573	338

- Prove que $3n^2 + 5n + 1$ não é $\theta(n)$

Pois para qualquer valor de c1 e c2, para n >= m, g(n) estará acima do limite

Para m = 15, c1 = 2 e c2 = 50

n	$3n^2+5n+1$	$2n$	n	$3n^2+5n+1$	$50n$
13	573	26	13	573	650

14	659	28	14	659	700	
15	751	30	15	751	750	<-------
16	849	32	16	849	800	
17	953	34	17	953	850	
18	1063	36	18	1063	900	
19	1179	38	19	1179	950	
20	1301	40	20	1301	1000	
21	1429	42	21	1429	1050	

- Prove que 3n 2 + 5n +1 não é O(n 3)

Pois para qualquer valor de c1 e c2, para n >= m, g(n) estará abaixo do limite estabelecido

Para c1 = 2, c2 = 4 e m = 10

n	3^2+5n+1	$2n^3$	n	3^2+5n+1	$4n^3$
10	351	2000	10	351	4000
11	419	2662	11	419	5324
12	493	3456	12	493	6912
13	573	4394	13	573	8788
14	659	5488	14	659	10976
15	751	6750	15	751	13500
16	849	8192	16	849	16384
17	953	9826	17	953	19652
18	1063	11664	18	1063	23328
19	1179	13718	19	1179	27436
20	1301	16000	20	1301	32000

Exercicio 10)

A)
Melhor caso (número de chamadas) :

1 + (n-2) = n-1 = O(n), Ω(n), θ(n)

Pior caso (número de chamadas) :

1 + (n-2) = n-1 = O(n), Ω(n), θ(n) = Melhor caso

B)
Melhor caso (número de chamadas) :

telefone () == false (1 chamada)

1 chamada

dentro do for:

sensor(i-2) for true em todas as chamadas

(n-2) * (1 chamada)

Total : 1 + n-2 = n-1 = O(n), Ω(n), θ(n)

Pior caso (número de chamadas) :

telefone() == true e luz() == false ou true

(2 chamadas)

dentro do for:

sensor(i-2) == false e camera(i-2) == true ou false

(2 chamadas) * n-2

Total = 2 + 2(n-2) = 2 + 2n-4 = 2n-2

Exercicio 11)

```
// Código para verificar o maior elemento de um array

int menor = array[0];
int maior = array[0];
for(int i = 1; i < n; i++) {
	if(array[i] > maior)
		maior = array[i];
	else if(array[i] < menor)
		menor = array[i];
}
```

Operações mais importantes
- Movimentação de elementos do array
- Comparação entre elementos do array

Função de complexidade para Movimentação de elementos do array :

Melhor caso (se o número já for o maior e o menor elemento do array):
$1 + 1 = 2$

Pior caso :
$1 + 1 + 1*(n-1) = 2 + (n-1) = n+1 = O(n), \Omega(n), \Theta(n)$

Função de complexidade para Comparação entre elementos do array:

Melhor caso (se todos os números forem maiores que o maior) :
$(n-1)*1 = n-1 = O(n), \Omega(n), \Theta(n)$

Pior caso :
$(n-1)*2 = 2n-2 = O(n), \Omega(n), \Theta(n)$