Exercícios: Análise de Complexidade de Algoritmos

1. O que significa dizer que uma função $g(n)$ é $O(f(n))$?
2. O que significa dizer que uma função $g(n)$ é $\Theta(f(n))$?
3. O que significa dizer que uma função $g(n)$ é $\Omega(f(n))$?
4. Suponha um algoritmo A e um algoritmo B com funções de complexidade de tempo $a(n) = n^2 - n + 549$ e $b(n) = 49n + 49$, respectivamente. Determine quais são os valores de n pertencentes ao conjunto dos números naturais para os quais A leva menos tempo para executar do que B.
5. Expresse a função $10n^3 - 5n^2 - 10n + 3$ em termos da notação Θ.
6. É verdade que $2n^3 + 5 = \Theta(n^3)$? Explique.
7. Dois algoritmos A e B possuem complexidade n^5 e 2^n respectivamente. Você utilizaria o algoritmo B ao invés do A, em qual caso? Explique.
8. Qual a ordem de complexidade no pior caso de:
   (a) $2n + 10$
   (b) $(1/2)n(n + 1)$
   (c) $n + \sqrt{n}$
   (d) $n/1000$
   (e) $(1/2)n^2$
   (f) $(1/2)n^2 - 3n$
9. Quais as grandezas físicas que influenciam a eficiência de tempo de um algoritmo na prática?
10. Para o cálculo da complexidade de algoritmos não recursivos, existe um conjunto de regras bastante simples de serem seguidas. Cite e descreva estas regras.
11. Explique que tipos de problemas ou algoritmos costumam ter complexidade da ordem de $n \log n$ e como os identificamos.
12. Quais problemas que possuem geralmente complexidade da ordem de $\log n$?
13. Quais problemas que costumam ser exponenciais?
14. Calcule a complexidade, no pior caso, do fragmento de código abaixo:

```
1       int i, j, k;
2       for (i = 0; i < N; i ++)  {
3         for (j = 0; j < N; j ++)  {
4           R[i][j] = 0;
5           for (k = 0; k < N; k++)
6             R[i][j] += A[i][k] * B[k][j];
7         }
8       }
```

15. Calcule a complexidade, no pior caso, do fragmento de código abaixo:

```
1        int i, j, k, s;
2        for (i = 0; i < N-1; i++)
3          for (j = i + 1; j < N; j++)
4            for (k = 1; k < j; k++)
5              s = 1;
```

16. Calcule a complexidade, no pior caso, do fragmento de código abaixo:

```
1        int i, j, s;
2        s = 0;
3        for (i = 1; i < N-1; i++)
4          for (j = 1; j < 2*N; j++)
5            s = s + 1;
```

17. Obtenha a equação matemática referente á análise do pior e melhor caso do fragmento de código abaixo:

```
1      for (i = 0; i < N; i++)
2        printf("%d", i);
```

18. Obtenha a equação matemática referente á análise do pior e melhor caso do fragmento de código abaixo:

```
1      for (i = 0; i < N; i = i+2)
2        printf("%d", i);
```

19. Obtenha a equação matemática referente á análise do pior e melhor caso do fragmento de código abaixo:

```
1      for (i = 0; i < N; i = i+2)      {
2        printf("%d", i);
3        i--;
4      }
```

20. Obtenha a equação matemática referente á análise do pior e melhor caso do fragmento de código abaixo:

```
1      for (i = 0; i < N; i = i+2)      {
2        for (j = N-i; j >=0; j--){
3          if(V[i] < V[j]) {
4            printf("%d", i);
5          }
6        }
7      }
```

21. Obtenha a equação matemática referente á análise do pior e melhor caso do fragmento de código abaixo:

```
1      for (i = 1; i <= N; i=2*i)
2        printf("%d", i);
```