Algoritmos e Estruturas de Dados – 2018.1
Prova A1 – Prof. Alexandre Rademaker

1. * Lembre-se da definição de que $f = O(g)$ se existe uma constante $c > 0$ tal que $f(n) \leq cg(n)$ para todo n. O que significa que "f não cresce mais rápido que g". Dada a lista de funções abaixo, ordene as funções em ordem crescente de "taxa crescimento". Isto é, se g estiver logo depois de f na sua ordenação, então $f = O(g)$.

$$f_1(n) = n^{2.5} \quad f_2(n) = \sqrt{2n} \quad f_3(n) = n + 10 \quad f_4(n) = 10^n \quad f_5(n) = 10^{10^n} \quad f_6(n) = n^2 \log_2 n$$

2. * Suponha que você tem 6 algorítmos com tempos de execução dados abaixo. Assuma que cada uma das funções abaixo retorna exatamente o número de instruções executadas para uma entrada de tamanho n. Suponha que você tem um computador que pode executar 10^{10} operações por segundo e você precisa computar a resposta em até uma hora. Para cada algorítmo, qual é a maior entrada, n, que você conseguiria obter o resultados em 1 hora?

$$A_1 = n^2 \qquad A_2 = n^3 \qquad A_3 = 100n^2 \qquad A_4 = n \log_2 n \qquad A_5 = 2^n \qquad A_6 = 2^{2^n}$$

   Lembre-se que você pode computar $f(n) = X/log_2 n$ com interações sucessivas até o valor convergir. Isto é, $f(\ldots f(f(chute))\ldots)$ irá ser um valor bem próximo da solução. O chute inicial pode ser qualquer valor maior que 1 e menor que X.

3. * Aprendemos duas formas de resolver relações de recorrência. Uma usando o "Master Theorem" e outra usando o método de substituições sucessivas (exercícios).

   Teorema 1 (Master Theorem) *If $T(n) = aT(\lceil n/b \rceil) + O(n^d)$ for some constants $a > 0$, $b > 1$, and $d \geq 0$ then*

$$T(n) = \begin{cases} O(n^d) & \text{if } d > \log_b a \\ O(n^d \log n) & \text{if } d = \log_b a \\ O(n^{log_b a}) & \text{if } d < \log_b a \end{cases}$$

   Resolva usando o teorema a relação $T(n) = 3T(n/2) + O(n)$.

4. Suponha que você tem k listas ordenadas, cada uma com n elementos. Você deseja combinar todos eles em um único array com kn elementos. Vide Figura 1

```
Input: k = 3, n =  4
arr[][] = [ {1, 3, 5, 7},
            {2, 4, 6, 8},
            {0, 9, 10, 11} ] ;
Output: 0 1 2 3 4 5 6 7 8 9 10 11
```

   Figura 1: Exemplo entrada

   (a) * Uma possível estratégia: Usando a função *merge-1* (arquivo *respostas.lisp*) junte os primeiros 2 arrays, depois junte o resultado com o terceiro, depois com o quarto e assim por diante. Implemente este algorítmo na função *questao-4-a* no arquivo *respostas.lisp*. Sua função deve receber listas de listas, usando o exemplo acima:

```
(questao-4-a '((1 3 5 7) (2 4 6 8) (0 9 10 11)))
```

   (b) * Qual a complexidade de tempo desta função do item anterior em termos de k e n? Responda na função *questao-4-b*.

   (c) Dê soluçãos mais eficientes implementando as funções *linear-k-merge*, *heap-merge* e *dc-merge* no arquivo *respostas.lisp* conforme descritas.

5. * Considere um grafo direcionado onde as únicas arestas negativas são aquelas que saem do nó s. Todas as outras arestas são positivas. Pode o algoritmo de Dijkstra [1], começando em s, falhar neste grafo? Ele precisa ser adaptado como? Implemente o algoritmo de Dijkstra adaptado (se necessário).

[1] `http://bit.ly/2H0sjsq`

6. Desenvolva um algoritmo eficiente que receba com entrada um grafo direcionado $G = (V, E)$ e dois vértices $u, v \in V$ e devolva como resposta o número de diferentes caminhos de u para v. (Dica: você pode partir do DFS [2] e adapta-lo.)

7. Em Barbacena, no interior de Minas, existe uma longa e calma rua com casas posicionadas de forma bastante espaçada. Existem apenas casas nesta rua. Podemos pensar na rua com um segmento de reta indo do norte para o sul. Embora seja uma cidade calma de interior, os moradores desta rua são compulsivos usuários de seus celulares que reclamam constantemente da falta de sinal. Sua missão, como novo engenheiro da XIM Celulares é definir onde deverão ser posicionadas antenas de transmissão ao longo desta rua de tal forma que, cada casa esteja a uma distância não superior à 4 Km de alguma antena.

   (a) Implemente na função *questao-7-a* um algoritmo eficiente que calcule o posicionamento das antenas usando o menor número possível de antenas. Considerando a rua como um segmento reto, a entrada do algoritmo será uma lista de números reais representando as posições de cada casa neste segmento.

   (b) * Qual a complexidade do seu algoritmo? Responda na função *questao-7-b*.

8. Implemene o algoritmo guloso na função *questao-8* para a satisfatibilidade de uma formula Horn (Seção 5.3) em tempo linear no comprimento da fórmula (o número de ocorrências de literais) (Dica: use um gráfico direcionado, com um nó por variável, para representar as implicações). Teste seu algorítmo resolve para a entrada da Fórmula 1.

$$(w \wedge y \wedge z) \rightarrow x, (x \wedge z) \rightarrow w, x \rightarrow y, \rightarrow x, (x \wedge y) \rightarrow w, (\bar{w} \vee \bar{x} \vee \bar{y}), (\bar{z}) \qquad (1)$$

[2] https://en.wikipedia.org/wiki/Depth-first_search