

Nome: _____

N. Mec.: _____

- 4.0 **1:** O algoritmo *merge sort* divide o *array* a ser ordenado ao meio, ordena (recursivamente) cada uma das duas partes, e depois junta-as. A sua complexidade computacional é $\Theta(n \log n)$. Um aluno está convencido que se em vez de se dividir o *array* em duas partes se se dividir em cinco partes (todas mais ou menos do mesmo tamanho), então a complexidade computacional desta variante do *merge sort* será ainda mais baixa. Responda às seguintes perguntas:

- 1.0 a) Que estratégia algorítmica usa o *merge sort*? *Recursiva*
- 2.0 b) O aluno tem razão? Justifique. *Não*
- 1.0 c) Nesta variante do *merge sort*, a fase de *merge* é mais fácil ou mais complicada que a do algoritmo original?

A complexidade depende do n° de comparações e de trocas

O *master theorem* afirma que se $T(n) = aT(n/b) + f(n)$ então

- se $f(n) = O(n^{\log_b a - \epsilon})$ para um $\epsilon > 0$, então $T(n) = \Theta(n^{\log_b a})$,
- se $f(n) = \Theta(n^{\log_b a})$, então $T(n) = O(n^{\log_b a} \log n)$,
- se $f(n) = \Omega(n^{\log_b a + \epsilon})$ para um $\epsilon > 0$ e se $af(\frac{n}{b}) \leq cf(n)$ para $c < 1$ e n suficientemente grande, então $T(n) = \Theta(f(n))$.

- 4.0 **2:** Num tabuleiro de xadrez, pretende-se ir do canto inferior esquerdo **(0,0)** para o canto superior direito **(7,7)** fazendo movimentos apenas para a direita e para cima. O seguinte código apresenta uma maneira de calcular o número de maneiras de fazer isso.

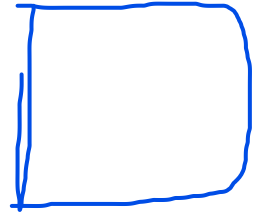
```
1  long eval(int x,int y)
   {
2    if(x < 0 || x > 7 || y < 0 || y > 7)
3      return 0L;
4    else if(x == 7 && y == 7)
5      return 1L;
6    else
7      return eval(x + 1,y) + eval(x,y + 1);
   }
8  long count_paths(void)
   {
9    return eval(0,0);
10   }
```

Responda às seguintes perguntas:

- 1.0 a) Que estratégia algorítmica é usada por este código? *Recursividade*
- 3.0 b) Explique por palavras qual é o objetivo de cada uma das partes numeradas do código.

- 4.0 **3:** Tal como no problema anterior, num tabuleiro de xadrez, pretende-se ir do canto inferior esquerdo **(0,0)** para o canto superior direito **(7,7)** fazendo movimentos apenas para a direita e para cima. O seguinte código apresenta uma maneira de calcular o número de maneiras de fazer isso.

```
1  long count_data[8][8];
2  long eval(int x,int y)
   {
3      if(x < 0 || x > 7 || y < 0 || y > 7)
3          return 0L;
4      if(count_data[x][y] < 0L)
4          count_data[x][y] = eval(x - 1,y) + eval(x,y - 1);
5      return count_data[x][y];
   }
6  long count_paths(void)
   {
7      for(int x = 0;x < 8;x++)
7          for(int y = 0;y < 8;y++)
7              count_data[x][y] = -1L;
8      count_data[0][0] = 1L;
9      return eval(7,7);
   }
```



Responda às seguintes perguntas:

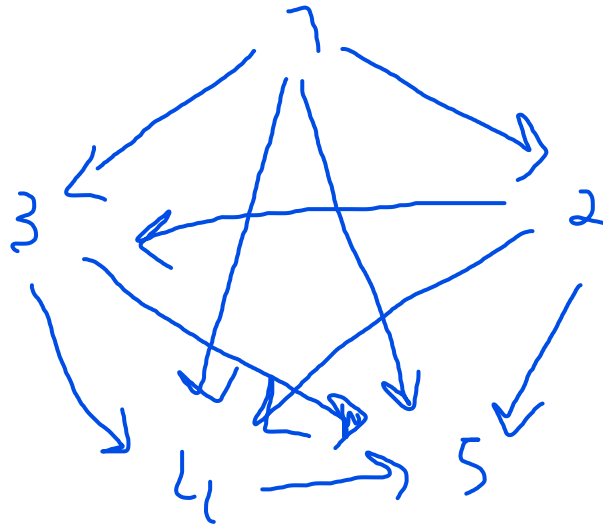
- 1.0 a) Que estratégia algorítmica é usada por este código? *Dynamic Programming*
- 3.0 b) Explique por palavras qual é o objetivo de cada uma das partes numerados do código.

*Com
Recursividade*

4.0 **4:** Um grafo com 5 vértices, numerados de **1** a **5**, tem uma aresta entre o vértice número i e o número j se e só se $i < j$. Responda às seguintes perguntas:

- 1.0 a) Desenhe o grafo.
- 1.0 b) Represente o grafo usando uma matriz de adjacência.
- 1.0 c) Represente o grafo usando listas de adjacência.
- 1.0 d) É possível representar este grafo usando um único inteiro de **32 bits**? Se sim, como? Se não, por que não?

$$\frac{5 \times 4}{2} = 10$$



4.0 **5:** Você foi capturado pelos Borg e levado para um cubo Borg para ser assimilado.¹ Você conseguiu fugir mas está perdido dentro do cubo Borg, que pode ser considerado um labirinto tridimensional. Responda às seguintes perguntas:

- 2.0 a) Para encontrar uma saída do cubo, usaria depth search ou breadth search?
- 2.0 b) Que material levaria consigo para o ajudar implementar a estratégia algorítmica que escolheu na alínea anterior?

Bloco de notas, migalhas (sinalizadores)

¹Os Borg são uma raça alienígena do universo *Star Trek*. As suas maiores naves são os cubos Borg, que têm o volume de **27** quilómetros cúbicos. A resistência é fútil!