Q1: código no Git Hub;

Questão 2) O método Selection Sort, abaixo, sofreu um pequeno ajuste: passou a trabalhar com a seleção do maior valor, colocando-o na última posição; do segundo maior valor colocando-o na penúltima posição e, assim, sucessivamente. Complete-o!

public void selectionSort () {

for (int i = valor[].length; i >= 0; i --) {

int maior = this.posMaior(i);

if (maior != i) this.troca (i, maior);

}

}

/\* Métodos auxiliares: \*/

private int posMaior (int fim) {

int maior = fim;

for (int i = fim-1; i >= 0; i--)

if (valor[i] > valor[maior]) maior = i;

return maior;

}

private void troca (int a, int b) {

float aux;

aux = valor[a];

valor[a] = valor[b];

valor[b] = aux;

}

Questão 3) Disponível no Git Hub

(Questão 4) Dados quatro algoritmos (A, B, C e D) e suas respectivas fórmulas gerais, analise a complexidade de cada um deles e classifique-os em ordem crescente de complexidade. Responda no seu PIÁ.

C<D<B<A

Questão 5) Disponível no Git Hub

Questão 6) Disponível no Git Hub; Pior caso: 3n + 9;

Questão 7) Disponível no Git Hub

Questão 8) Disponível no Git Hub

Questão 9) Disponível no Git Hub

Questão 10)

Bubble Sort:

1. [3, 7, 1, 4, 9, 2]
2. [3, 1, 7, 4, 9, 2]
3. [3, 1, 4, 7, 9, 2]
4. [3, 1, 4, 7, 2, 9]
5. [1, 3, 4, 7, 2, 9]
6. [1, 3, 4, 2, 7, 9]
7. [1, 3, 2, 4, 7, 9]
8. [1, 2, 3, 4, 7, 9]

Selection Sort:

1. [3, 7, 1, 4, 9, 2]
2. [3, 7, 1, 4, 2, 9]
3. [3, 2, 1, 4, 7, 9]
4. [1, 2, 3, 4, 7, 9]

Insertion Sort:

1. [3, 7, 1, 4, 9, 2]
2. [3, 1, 7, 4, 9, 2]
3. [1, 3, 7, 4, 9, 2]
4. [1, 3, 4, 7, 9, 2]
5. [1, 3, 4, 7, 2, 9]
6. [1, 3, 4, 2, 7, 9]
7. [1, 3, 2, 4, 7, 9]
8. [1, 2, 3, 4, 7, 9]

Questão 11) A estabilidade de um método de ordenação é não alterar a ordem de valores iguais durante a ordenação.

Ordenando com Selection Sort:

1. [7, 1, 1, 3, 2]
2. [2, 1, 1, 3, 7]
3. [1, 1, 2, 3, 7]

Neste exemplo a estabilidade não é mantida pois na 3º modificação o “1” que estava na terceira posição passa a estar na primeira posição, enquanto o “1” que estava na segunda posição permanece.

Ordenando com Bubble Sort:

1. [7, 1, 1, 3, 2]
2. [1, 7, 1, 3, 2]
3. [1, 1, 7, 3, 2]
4. [1, 1, 3, 7, 2]
5. [1, 1, 3, 2, 7]
6. [1, 1, 2, 3, 7

Neste caso a ordem foi mantida, ou seja, é estável.

Questão 12) Código no Git Hub

Eu não escolheria o método recursivo por questões de desempenho, em comparação com o método linear:

Número de instruções para o método recursivo:

melhor caso: 2;

pior caso: 5 (n - m) + 1 se (n - m) = 2 -> 11 se (n – m) = 3 -> 16

Número de instruções para o método linear:

melhor caso: 6;

pior caso: 3 (n - m) + 5 se (n – m) = 2 -> 11 se (n – m) = 3 -> 14

então se (n – m) > 2 o método recursivo torna-se inferior comparado ao linear.