Exercícios – Lista 4

1) Considere uma struct que armazena os seguintes dados de uma conta bancária: número da conta, o primeiro nome do cliente e saldo. Faça um algoritmo que preencha os dados da conta bancária de 20 clientes e depois imprima quantos deles têm saldo bancário maior do que R$ 1.000,00.

2) Considerando a struct do exercício anterior, faça um algoritmo que preencha os dados da conta bancária de 20 clientes e depois imprima quantos deles têm saldo bancário de no máximo 3 salários mínimos, onde o salário mínimo é igual a R$ 880,00. Utilize um **procedimento** para preencher os dados dos clientes e uma **função** para retornar (para a função *main*) a quantidade de clientes.

3) Considere uma struct que armazena as coordenadas x e y de um ponto no R^2. Faça um algoritmo que leia as coordenadas de dois pontos e imprima a distância entre eles. Utilize um **procedimento** para ler as coordenadas dos pontos e uma **função** para retornar (para a função *main*) a distância entre eles.

4) Considere uma struct que armazena os seguintes dados de um aluno: matrícula, primeiro nome, nota da P1, nota da P2 e nota da P3. Faça um algoritmo que preencha os dados de 30 alunos, sendo que a nota da P3 será calculada pelo algoritmo como a média das outras duas notas. O algoritmo também tem que imprimir a quantidade de alunos Aprovados e Reprovados, considerando a média para aprovação maior ou igual a 6,0. Utilize dois **procedimentos**: um para preencher os dados dos alunos e outro para imprimir as informações solicitadas.

5) Considerando a struct do exercício anterior, faça um algoritmo que preencha os dados de 30 alunos, sendo que a nota da P3 será calculada pelo algoritmo como a média das outras duas notas, e depois imprima as três notas de um aluno cujo nome será informado pelo teclado na função *main*. Caso a turma tenha mais de um aluno com o mesmo nome, imprima as três notas de todos eles. Utilize dois **procedimentos**: um para preencher os dados dos alunos e outro para imprimir as informações solicitadas.

6) Considere um vetor com 15 números inteiros lidos pelo teclado. Faça um algoritmo para imprimir este vetor em ordem não-decrescente utilizado o algoritmo **BubbleSort**. Utilize três **procedimentos**: um para preencher o vetor, outro para ordenar o vetor e um terceiro para imprimir o vetor antes e depois da ordenação.

7) Considere um vetor com 20 números inteiros gerados aleatoriamente de 1 até 30. Faça um algoritmo para imprimir este vetor em ordem não-crescente utilizado o algoritmo **BubbleSort**. Utilize três **procedimentos**: um para preencher o vetor, outro para ordenar o vetor e um terceiro para imprimir o vetor antes e depois da ordenação.

8) Considere um vetor com 15 números inteiros lidos pelo teclado. Faça um algoritmo para imprimir este vetor em ordem não-decrescente utilizado o algoritmo **SelectionSort**. Utilize três **procedimentos**: um para preencher o vetor, outro para ordenar o vetor e um terceiro para imprimir o vetor antes e depois da ordenação.

9) Considere um vetor com 20 números inteiros gerados aleatoriamente de 1 até 30. Faça um algoritmo para imprimir este vetor em ordem não-crescente utilizado o algoritmo **SelectionSort**. Utilize três **procedimentos**: um para preencher o vetor, outro para ordenar o vetor e um terceiro para imprimir o vetor antes e depois da ordenação.

10) Considere um vetor com 15 números inteiros lidos pelo teclado. Faça um algoritmo para imprimir este vetor em ordem não-decrescente utilizado o algoritmo **InsertionSort**. Utilize três **procedimentos**: um para preencher o vetor, outro para ordenar o vetor e um terceiro para imprimir o vetor antes e depois da ordenação.

11) Considere um vetor com 20 números inteiros gerados aleatoriamente de 1 até 30. Faça um algoritmo para imprimir este vetor em ordem não-crescente utilizado o algoritmo **InsertionSort**. Utilize três **procedimentos**: um para preencher o vetor, outro para ordenar o vetor e um terceiro para imprimir o vetor antes e depois da ordenação.

12) Considere um vetor com 15 números inteiros lidos pelo teclado. Faça um algoritmo para imprimir este vetor em ordem não-decrescente utilizado o algoritmo **QuickSort**. Utilize três **procedimentos**: um para preencher o vetor, outro para ordenar o vetor e um terceiro para imprimir o vetor antes e depois da ordenação.

13) Considere um vetor com 20 números inteiros gerados aleatoriamente de 1 até 30. Faça um algoritmo para imprimir este vetor em ordem não-crescente utilizado o algoritmo **QuickSort**. Utilize três **procedimentos**: um para preencher o vetor, outro para ordenar o vetor e um terceiro para imprimir o vetor antes e depois da ordenação.

14) Considere o seguinte vetor:

7	5	10	2	8	9	1	4	3	6

Descreva **passo a passo** a ordenação desse vetor utilizando cada um dos algoritmos de ordenação estudados: BubbleSort, SelectionSort, InsertionSort e QuickSort.

15) Considere uma struct que armazena o primeiro nome, a idade, o salário e o cargo de 10 funcionários de uma empresa. Faça um algoritmo que imprima os funcionários em ordem não-decrescente pela **idade** utilizando o algoritmo **SelectionSort**. Utilize três **procedimentos**: um para preencher os dados dos funcionários, outro para realizar a ordenação e um terceiro para imprimir os dados depois da ordenação.

16) Considere um número inteiro n ($n \geq 0$) lido pelo teclado. Faça um algoritmo **recursivo** para calcular o fatorial de n.

17) Considere dois números inteiros a e b ($b \geq 0$) lidos pelo teclado. Faça um algoritmo **recursivo** para calcular o valor de a^b.

18) Considere um vetor com 20 números inteiros gerados aleatoriamente de 1 até 50. Faça um algoritmo **recursivo** para calcular a **soma** dos elementos deste vetor.

19) Na Série de Fibonacci os dois primeiros elementos são conhecidos (1 e 1), sendo que os demais são gerados a partir da soma dos dois anteriores. Por exemplo, os dez primeiros elementos da Série são: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55.

Faça um algoritmo **recursivo** para imprimir o elemento da posição n ($n \geq 1$) dessa Série. Exemplos: se o usuário digitar $n = 1$ (ou seja, a **primeira** posição), deve ser impresso o número **1**. Se o usuário digitar $n = 6$ (ou seja, a **sexta** posição), deve ser impresso o número **8**.

20) Considere um vetor com 20 números inteiros gerados aleatoriamente de 1 até 40. Faça um algoritmo **recursivo** para imprimir o **maior** valor deste vetor.