USJT- Universidade São Judas Tadeu

Algoritmos e Estruturas de Dados – ALGESD

SIN2AN-MCA – Prof. Calvetti

Lista de Exercícios – Parte 1/2

Vetores:

1º) Elaborar um programa, em linguagem Java, capaz inicializar com 0 (zero) um vetor do tipo *int* de 100 elementos, utilizando-se das malhas (laços, *loops*, etc.) *while*, *do-while* e *for*;

2º) Elaborar um programa, em linguagem Java, capaz de, em um vetor do tipo *int* de 100 elementos, carregar seus índices pares com o número 0 (zero) e seus índices ímpares com o valor do próprio índice (ex.: [0, 1, 0, 3, 0, 5, 0, 7, ..., 97, 0, 99]), utilizando-se de malhas;

3º) Elaborar um programa, em linguagem Java, capaz de carregar um vetor do tipo *char* de 26 elementos com os caracteres de A até Z pelo próprio programa, sem que haja digitação, utilizando malhas;

4º) Dado o vetor gerado pelo exercício 3 (['A', 'B', 'C', 'D', ... , 'W', 'X', 'Y', 'Z']), elaborar um programa em linguagem Java capaz de trocar a ordem de seus elementos, de dois em dois, até o final do mesmo (['B', 'A', 'D', 'C', ... , 'X', 'W', 'Z', 'Y']), utilizando malhas;

5º) Dado um vetor do tipo *int* de 16 elementos, a serem digitados aleatoriamente, elaborar um programa, em linguagem Java, capaz de apresentar a quantidade de capicuas de 4 elementos existentes ao longo desse vetor (capicua: número que representa o mesmo valor quando lido da esquerda para a direita e vice-versa). Exemplo: Vetor digitado

índices:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
elementos:	0	1	1	0	3	2	4	4	2	2	4	7	7	7	7	7;

Total de Capicuas: 5

Obs.: Foram encontradas as seguintes capicuas: 0110 (índices 0, 1, 2 e 3), 2442 (índices 5, 6, 7 e 8), 4224 (índices 7, 8, 9 e 10), 7777 (índices 11, 12, 13 e 14) e 7777 (índices 12, 13, 14 e 15).

Matrizes:

6º) Elaborar um programa, em linguagem Java, capaz de limpar (colocar -1 em todas as suas posições) qualquer tipo de matriz, inteira e positiva e que seja declarada globalmente. O programador deverá definir o tamanho da matriz (linhas e colunas), antes de compilar o programa.

7º) Determinar a matriz transposta de uma matriz global qualquer e digitada, através de um programa em linguagem Java. Uma matriz é dita transposta quando a matriz original tiver suas linhas transformadas em colunas e suas colunas transformadas em linhas.

8º) Elaborar um programa, em linguagem Java, capaz de informar quando uma matriz qualquer é simétrica. Essa matriz deverá ser global e todos os seus elementos digitados. Uma matriz é dita simétrica quando ela for igual à sua transposta.

9º) Determinar se uma matriz é identidade (ou unidade), através de um programa em linguagem Java. A matriz deverá ser global e todos os seus elementos digitados. Uma matriz é dita identidade (ou unidade) quando ela for quadrada (no. de linhas iguais ao no. de colunas), tiver todos os elementos de sua diagonal principal (no. da linha igual ao no. da coluna) iguais a 1 (um) e todos os demais elementos iguais a 0 (zero).

10º) Um teatro possui 3000 lugares divididos em 30 fileiras, cada uma com 100 cadeiras. Elaborar um programa em linguagem Java, capaz de gerenciar a venda dos ingressos para este teatro. Cada lugar poderá estar Livre (0), Reservado (1) ou Vendido (2).

Métodos (declarações e chamadas):

11º) Utilizando programação por chamadas de métodos, elaborar um programa, em linguagem Java, capaz de receber os valores de *a*, *b* e *c* de uma equação de 2º grau qualquer ($a\,x^2 + b\,x + c = 0$), calcular e apresentar o valor de Δ (delta) e informar se suas raízes são imaginárias, reais iguais ou reais diferentes, apresentando seus valores para os casos quando foram reais. Dica: Criar um método para calcular a raiz 1 e outro método para calcular a raiz 2;

12º) Criar um método, que se utilize de malhas, capaz de informar se o número inteiro e maior que zero digitado pelo operador é ou não primo. Este método deverá ser utilizado por um programa em linguagem Java, que pedirá ao operador a digitação do número, verificará se o mesmo é ou não primo e apresentará a conclusão em tela;

13º) Da linha de produção de uma metalúrgica serão pegos, aleatoriamente, no decorrer de um dia, pelo controle de qualidade da empresa, 10 pregos sem cabeça para análise. Para cada amostra de prego pego, serão medidos seu comprimento e diâmetro, ambos em milímetros. Durante a medição, esses dados serão inseridos, um a um, em um programa de computador desenvolvido especialmente para isso. Após a digitação dos 10 pares de valores (comprimento e diâmetro), o programa deverá fornecer:

a) Comprimento e Diâmetro Médios das amostras verificadas;

b) O número e o comprimento da amostra mais longa (comprimento maior);

c) O número e o diâmetro da amostra mais fina (diâmetro menor).

Obs.: Valor médio do comprimento das amostras: $C_{médio} = (C_1+C_2+ ... + C_9 + C_{10}) / 10$

Valor médio do diâmetro das amostras: $D_{médio} = (D_1+D_2+ ... + D_9 + D_{10}) / 10$

14º) Utilizando programação por chamadas de métodos, elaborar um programa, em linguagem Java, capaz de carregar, via teclado, os 10 elementos de um vetor do tipo *double*. Após isto, apresentar ao operador um menu contendo 3 opções e aguardar a digitação da opção por ele escolhida: Digitar '+' para apresentar o próximo elemento do vetor; Digitar '-' para apresentar o elemento anterior do vetor; Digitar '. ' para sair. Dependendo da opção digitada, apresentar o respectivo elemento do vetor carregado. Obs.: O 1° elemento a ser apontado é o de índice 0;

15º) Utilizando programação por chamada de métodos, elaborar um programa em linguagem Java, capaz de criar um menu em tela, com as seguintes opções:

A- Inserir número secreto;

B- Jogar;

C- Apresentar resultados;

D- Sair;

- Se a opção A for digitada, o programa deverá fornecer instruções ao operador e receber, via teclado, um número inteiro e positivo, secreto, objetivo de adivinhação do jogo. Após isto, o programa deverá voltar ao menu principal;

- Se a opção B for digitada, o jogo se iniciará e será colocado 0 no contador de palpites. O programa fornecerá instruções ao operador e receberá, via teclado, um número inteiro e positivo, a ser comparado com o número secreto. Se o número digitado for maior que o secreto, o programa informará ao operador a palavra ALTO, incrementará o contador de palpites e aguardará o próximo palpite; Se o número digitado for menor que o secreto, o programa informará ao operador a palavra BAIXO, incrementará o contador de palpites e aguardará o próximo palpite; Se o número digitado for igual ao secreto, o programa informará ao operador a palavra ACERTOU e retornará ao menu principal; Se o número digitado for negativo, o programa deverá retornar ao menu principal.

- Se a opção C for digitada, o programa apresentará a quantidade de palpites que foram necessários para acontecer último acerto. Após isto, o programa deverá retornar ao menu principal.

- Se a opção D for digitada, o programa deve encerrar-se.

<u>Métodos (passagem de parâmetros)</u>:

16º) Elaborar um método, em linguagem Java, com passagem de parâmetros, dentro de um programa teste, capaz de informar se o número digitado é par ou ímpar. A digitação do valor e a apresentação do resultado deverá acontecer externamente a este método;

17º) Rescrever o exercício no. 11 desta lista, utilizando métodos com passagem de parâmetros;

18º) Rescrever o exercício no. 12 desta lista, utilizando métodos com passagem de parâmetros;

19º) Rescrever o exercício no. 13 desta lista, utilizando métodos com passagem de parâmetros;

20º) Rescrever o exercício no. 14 desta lista, utilizando métodos com passagem de parâmetros;

Métodos (recursividade):

21º) Elaborar um programa, em linguagem Java, para calcular o fatorial de um número a ser digitado. Criar um método para cálculo do mesmo utilizando recursividade;

22º) Elaborar um programa, em linguagem Java, para calcular o N-ésimo elemento da série de Fibonacci. O índice desse elemento deverá ser digitado para a realização do cálculo. Criar um método que calcula o resultado da série, utilizando-se da recursividade.

Série de Fibonacci: O próximo elemento tem o valor igual à soma dos dois elementos anteriores da série: 1, 1, 2, 3, 5, 8, 13, 21, ..., ∞;

23º) Utilizando recursividade, criar um programa, em linguagem Java, que calcule e apresente a soma de todos os elementos de um vetor, inteiro, de tamanho 10, o qual deverá ser preenchido, anteriormente ao cálculo, pelo operador;

24º) Criar um método, em linguagem Java, que se utilize da recursividade, dentro de um programa capaz de receber, via teclado, um número inteiro qualquer e informar se o mesmo é ou não primo;

25º) Elaborar um programa, em linguagem Java, que utilize apenas métodos recursivos, capaz de receber, via teclado, 10 elementos tipo *float*, formando um vetor, e um outro elemento tipo *float*, o qual deverá ser comparado à cada elemento do vetor. O programa deverá informar o valor do primeiro índice do vetor, a partir do 0, que contém um elemento igual ao valor digitado. Ex.:

Índice:	0	1	2	3	4	5	6	7	8	9
Elementos Digitados:	1	2	3	4	5	6	7	8	9	10

Número Digitado: 4 → Resposta: O número 4 está localizado no índice 3 do vetor.

26º) Elaborar um programa, em linguagem Java, que utilize apenas métodos recursivos, capaz de receber, via teclado, dois números inteiros e positivos, calculando o Máximo Divisor Comum entre eles.

27º) Elaborar um programa, em linguagem Java, que utilize apenas métodos recursivos, capaz de receber, via teclado, dois números inteiros e positivos, calculando o primeiro elevado ao segundo.

28º) Elaborar um programa, em linguagem Java, que utilize apenas métodos recursivos, capaz de receber, via teclado, dois números inteiros e positivos, calculando a multiplicação entre esses dois números, porém, não se utilizando a multiplicação para realizar o cálculo e sim a operação de soma sucessiva.

Exemplo: 4 * 3 é igual a 3 + 3 + 3 + 3.

29º) Elaborar um programa, em linguagem Java, que utilize apenas métodos recursivos, capaz de receber, via teclado, dois números inteiros e positivos, calculando a subtração do segundo no primeiro, porém, não se utilizando da subtração para realizar o cálculo e sim a operação de comparação sucessivamente.

Exemplo: 5 – 3 => 3+1 = 4; 3+2=5. Portanto 5 – 3 = 2.

30º) Elaborar um programa, em linguagem Java, que utilize apenas métodos recursivos, capaz de receber, via teclado, um número inteiro, positivo, que represente o termo da série abaixo. Calcular o valor da série, ou seja, a soma de todos os valores calculados, do 1º termo até o termo digitado.

$$S = (1/2^0) + (1/2^1) + (1/2^2) + (1/2^3) + (1/2^4) + ... + (1/2^{N-1}) + (1/2^N)$$

Exemplo: Digitado o termo: 4 => S = 1 + 1/2 + 1/4 + 1/8 => S = 1.875

31º) Reescrever o exercício anterior, porém colocando mensagens dentro do método recursivo, de valores de entrada e valores de retorno, afim de acompanhar-se a evolução da execução do programa.

Busca Linear:

32º) Elaborar um programa, em linguagem Java, capaz de carregar, em um vetor do tipo *int*, 15 números digitados pelo operador, formando uma espécie de base de dados. Após isto, o programa deverá solicitar a digitação de outro número, denominado número de busca, o qual será localizado no vetor anterior. O programa deverá apresentar, em tela, o resultado de uma busca linear, informando o índice do vetor no qual se encontra o número de busca (utilizar o tamanho máximo do vetor como terminador da operação, caso o número de busca não seja localizado). A busca linear deverá ser um método que se utilize de passagem de parâmetros e o vetor não deverá ser variável global do programa.

33º) Elaborar um programa, em linguagem Java, capaz de carregar uma *string*, com 15 caracteres, digitada, em um vetor do tipo *char*, com 15 posições, formando uma espécie de base de dados. Após isto, o programa deverá solicitar a digitação de um caractere, denominado caractere de busca, o qual será localizado no vetor anterior. O programa deverá apresentar, em tela, o resultado de uma busca linear, informando o índice do vetor no qual se encontra o caractere de busca (utilizar o tamanho máximo do vetor como terminador da operação, caso o caractere de busca não seja localizado). A busca linear deverá ser um método que se utilize de passagem de parâmetros e o vetor não deverá ser uma variável global do programa.

34º) Elaborar um programa, em linguagem Java, capaz de carregar uma *string*, digitada em um vetor do tipo *char*, com no máximo 15 posições, formando uma espécie de base de dados. Após isto, o programa deverá solicitar a digitação de um caractere, denominado caractere de busca, o qual será localizado no vetor anterior. O programa deverá apresentar, em tela, o resultado de uma busca linear, informando o índice do vetor no qual se encontra o caractere de busca (utilizar o tamanho máximo do vetor ou o tamanho máximo da *string* digitada como terminadores da operação, caso o caractere de busca não seja localizado). A busca linear deverá ser um método que se utilize de passagem de parâmetros e o vetor não deverá ser uma variável global do programa.

35º) Elaborar um programa, em linguagem Java, capaz de controlar os dados de uma prova de atletismo, onde, via matriz, possamos armazenar, através de digitação, a quantidade de corredores inscritos na prova, o número de inscrição de cada um, sua posição de chegada e seu respectivo tempo de prova, conforme os mesmos forem passando pela linha de chegada. Os atletas inscritos que não ultrapassarem a linha de chegada deverão ter cadastrados 0 (zero) para a posição de chegada e tempo de prova. Ao término da digitação, o programa deverá realizar a busca da posição de chegada e do tempo de prova, para um determinado corredor, dada a digitação de seu número de inscrição. Utilizar matriz para armazenamento das informações, porém não sendo globais. O número de inscrição deve ser variável do tipo int. O tempo de prova deve ser em segundos e armazenado em variável do tipo int.

36º) Elaborar um programa, em linguagem Java, que receba, via digitação, a quantidade de alunos matriculados em uma universidade e apresente um menu contendo 3 possibilidades de escolha para o operador, realizando as respectivas funções:

A) Cadastrar RA de todos os alunos matriculados. (Utilizar um vetor do tipo *int* para o RA, ocupando os mesmos índices do respectivo vetor de NOTA);

B) Cadastrar NOTA de prova para todos os alunos matriculados. (Utilizar um vetor do tipo *double* para a NOTA, ocupando os mesmos índices do respectivo vetor de RA);

C) Realizar a busca linear e apresentar a NOTA para um determinado RA, a ser digitado;

D) Sair do programa.

37º) Em um serviço de *check-in* de uma companhia aérea, após a apresentação de seu R.G, o passageiro deve receber o primeiro número do assento vago disponível na aeronave, afim de realizar sua viagem. Para isto, a companhia aérea conta com um programa, realizado em linguagem Java. Esse programa informa ao operador, quando solicitado, o número do primeiro assento que estiver vago. Após isto, o operador digita o número do R.G. do passageiro, para realizar a reserva do assento. O programa deverá informar quando o avião estiver lotado e em que assento se encontra determinado R.G. digitado. Deverão existir opções para cancelar a reserva de um determinado assento e para cancelar todas as reservas de assento de uma só vez. Elaborar este programa.

38º) Elaborar um programa, em linguagem Java, que utilize apenas métodos iterativos, capaz de receber, via teclado, 10 elementos tipo *double* de um vetor e um outro elemento tipo *double* com o qual será feita uma busca linear no vetor digitado, ao longo de todos os seus índices, no sentido do menor índice para o maior índice. O programa deverá informar o valor do maior índice do vetor que contém o elemento igual ao valor digitado. Ex.:

Índice: 0 1 2 3 4 5 6 7 8 9
Elementos Digitados: 7 3 4 5 8 4 7 4 9 8
Número Digitado: 4 → Resposta: O número 4 está localizado no índice 7 do vetor.

<u>Busca Linear Recursiva</u>:

39º) Elaborar um programa, em linguagem Java, que utilize apenas métodos recursivos, capaz de receber, via teclado, 10 elementos tipo *int* de um vetor e um outro elemento tipo *int* com o qual será feita uma busca linear no vetor digitado, ao longo dos seus índices ímpares somente, no sentido do menor índice para o maior índice. O programa deverá informar o valor do menor índice do vetor que contém o elemento igual ao valor digitado. Ex.:

Índice: 0 1 2 3 4 5 6 7 8 9
Elementos Digitados: 7 3 4 5 8 4 7 8 9 4
Número Digitado: 4 → Resposta: O número 4 está localizado no índice 5 do vetor.

40º) Elaborar um programa, em linguagem Java, que utilize apenas métodos recursivos, capaz de receber, via teclado, 10 elementos tipo *char* de um vetor e um outro elemento tipo *char* com o qual será feita uma busca linear no vetor digitado, ao longo de todos os seus índices, no sentido do maior índice para o menor índice. O programa deverá informar o valor do maior índice do vetor que contém o elemento igual ao valor digitado. Ex.:

Índice: 0 1 2 3 4 5 6 7 8 9
Elementos Digitados: c d a b k a f g a w
Caractere Digitado: a → Resposta: O caractere a está localizado no índice 8 do vetor.

41º) Elaborar um programa, em linguagem Java, que utilize apenas métodos recursivos, capaz de receber, via teclado, 10 elementos tipo *double* de um vetor e um outro elemento tipo *double* com o qual será feita uma busca linear no vetor digitado, ao longo de todos os seus índices, no sentido do menor índice para o maior índice. O programa deverá informar o valor do maior índice do vetor que contém o elemento igual ao valor digitado. Ex.:

Índice: 0 1 2 3 4 5 6 7 8 9
Elementos Digitados: 7 3 4 5 8 4 7 4 9 8
Número Digitado: 4 → Resposta: O número 4 está localizado no índice 7 do vetor.

Busca Binária:

42º) Elaborar um programa, em linguagem Java, capaz de receber 10 elementos diferentes do tipo *char*, em ordem crescente, armazená-los em um vetor do mesmo tipo e tamanho, no sentido do menor para o maior índice. O programa deverá pedir ao operador para que digite mais um caractere, o qual será localizado no determinado vetor. O programa apresentará em tela o resultado de uma busca binária por todos os índices, do menor para o maior, informando ao operador o número do índice do vetor no qual se encontra o caractere digitado, ou, se for o caso, que o caractere não se encontra no vetor (utilizar o tamanho máximo do vetor como terminador da busca no vetor). Utilizar um método iterativo de busca binária construído com passagem de parâmetros.

43º) Tomando o exercício anterior como base, desenvolver dois métodos de busca especiais, um linear e outro binário (iterativo), para o mesmo vetor e o mesmo elemento chave, informando ao final, ao invés do índice onde foi encontrado o elemento chave, quantas tentativas foram necessárias para se localizar ou não o elemento chave pelos dois métodos. Com o programa sendo executado diversas vezes, para diversos valores armazenados ou não no vetor, redigir um comparativo entre os métodos de busca linear e busca binária, tentando concluir quando devemos utilização de um método é vantajosa em relação ao outro.

44º) Realizando através de busca binária iterativa, resolver o exercício no. 32 desta lista.

45º) Realizando através de busca binária iterativa, resolver o exercício no. 33 desta lista.

46º) Realizando através de busca binária iterativa, resolver o exercício no. 34 desta lista.

47º) Realizando através de busca binária iterativa, resolver o exercício no. 35 desta lista.

48º) Realizando através de busca binária iterativa, resolver o exercício no. 36 desta lista.

Busca Binária Recursiva:

49º) Elaborar um programa, em linguagem Java, capaz de receber, via teclado, 10 elementos tipo *int* de um vetor e um outro elemento tipo *int* com o qual será feita uma busca binária recursiva no vetor digitado, ao longo de todos os seus índices e no sentido do menor para o maior índice. O programa deverá informar o valor do menor índice do vetor que contém o elemento igual ao valor digitado. Ex.:

Índice:	0	1	2	3	4	5	6	7	8	9
Elementos Digitados:	7	3	4	5	8	4	7	8	9	4

Número Digitado: 4 → Resposta: O número 4 está localizado no índice 2 do vetor.

50º) Realizando através de busca binária recursiva, resolver o exercício no. 40 desta lista. Os dados devem estar ordenados decrescentemente, ou seja, o dado com o valor mais baixo deverá estar posicionado no maior índice do vetor e o dado com o valor mais alto deverá estar posicionado no menor índice do vetor.

51º) Resolver o exercício anterior desta lista, com os dados ordenados crescentemente, ou seja, o dado com o valor mais baixo deverá estar posicionado no menor índice do vetor e o dado com o valor mais alto deverá estar posicionado no maior índice do vetor.

52º) Realizando através de busca binária recursiva, resolver o exercício no. 42 desta lista.

53º) Realizando através de busca binária recursiva, resolver o exercício no. 43 desta lista.

54º) Realizando através de busca binária recursiva, resolver o exercício no. 44 desta lista.

55º) Realizando através de busca binária recursiva, resolver o exercício no. 45 desta lista.

56º) Realizando através de busca binária recursiva, resolver o exercício no. 46 desta lista.

57º) Realizando através de busca binária recursiva, resolver o exercício no. 47 desta lista.

58º) Realizando através de busca binária recursiva, resolver o exercício no. 48 desta lista.

Ordenação por Trocas:

59º) Elaborar um programa, em linguagem Java, capaz de receber, em ordem aleatória, o conteúdo de cada elemento de um vetor do tipo *char*, de tamanho 5, via teclado, executar uma ordenação crescente no mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação por trocas.

60º) Elaborar um programa, em linguagem Java, capaz de receber, via teclado e em ordem aleatória, o conteúdo de cada um dos elementos de um vetor do tipo *int*, de tamanho variável, também digitado, executar a ordenação crescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação por trocas.

61º) Elaborar um programa, em linguagem Java, capaz de receber, em ordem aleatória, o conteúdo de cada elemento de um vetor do tipo *double*, de tamanho 20, via teclado, executar uma ordenação decrescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação por trocas.

62º) Elaborar um programa, em linguagem Java, capaz de informar quantas operações de troca realizaremos no exercício no. 59 para ordenar o vetor e, através dele, concluir quantas são realizadas para o melhor e para o pior caso?

63º) Elaborar um programa, em linguagem Java, capaz de realizar a busca de um determinado caractere, digitado, em um vetor qualquer de tamanho 10, também com seus elementos do tipo *char* digitados, informando qual o índice do vetor que esse caractere se localiza. Utilizar ordenação do tipo troca e busca binária recursiva.

Ordenação por Seleção:

64º) Elaborar um programa, em linguagem Java, capaz de receber, em ordem aleatória, o conteúdo de cada elemento de um vetor do tipo *char*, de tamanho 5, via teclado, executar uma ordenação crescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação por seleção.

65º) Elaborar um programa, em linguagem Java, capaz de receber, via teclado e em ordem aleatória, o conteúdo de cada um dos elementos de um vetor do tipo *int*, de tamanho variável, também digitado, executar a ordenação crescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação por seleção.

66º) Elaborar um programa, em linguagem Java, capaz de receber, em ordem aleatória, o conteúdo de cada elemento de um vetor do tipo *double*, de tamanho 20, via teclado, executar uma ordenação decrescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação por seleção.

67º) Elaborar um programa, em linguagem Java, capaz de informar quantas operações de seleção realizaremos no exercício no. 64 para ordenar o vetor e, através dele, concluir quantas são realizadas para o melhor e para o pior caso?

68º) Elaborar um programa, em linguagem Java, capaz de realizar a busca de um determinado caractere, digitado, em um vetor qualquer de tamanho 10, também com seus elementos do tipo *char* digitados, informando qual o índice do vetor que esse caractere se localiza. Utilizar ordenação do tipo seleção e busca binária recursiva.

<u>Ordenação por Inserção</u>:

69º) Elaborar um programa, em linguagem Java, capaz de receber, em ordem aleatória, o conteúdo de cada elemento de um vetor do tipo *char*, de tamanho 5, via teclado, executar uma ordenação crescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação por inserção.

70º) Elaborar um programa, em linguagem Java, capaz de receber, via teclado e em ordem aleatória, o conteúdo de cada um dos elementos de um vetor do tipo *int*, de tamanho variável, também digitado, executar a ordenação crescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação por inserção.

71º) Elaborar um programa, em linguagem Java, capaz de receber, em ordem aleatória, o conteúdo de cada elemento de um vetor do tipo *double*, de tamanho 20, via teclado, executar uma ordenação decrescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação por inserção.

72º) Elaborar um programa, em linguagem Java, capaz de informar quantas operações de inserção realizaremos no exercício no. 69 para ordenar o vetor e, através dele, concluir quantas são realizadas para o melhor e para o pior caso?

73º) Elaborar um programa, em linguagem Java, capaz de realizar a busca de um determinado caractere, digitado, em um vetor qualquer de tamanho 10, também com seus elementos do tipo *char* digitados, informando qual o índice do vetor que esse caractere se localiza. Utilizar ordenação do tipo inserção e busca binária recursiva.

Quick-Sort:

74º) Elaborar um programa, em linguagem Java, capaz de receber, em ordem aleatória, o conteúdo de cada elemento de um vetor do tipo *char*, de tamanho 5, via teclado, executar uma ordenação crescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação *Quick-Sort*.

75º) Elaborar um programa, em linguagem Java, capaz de receber, via teclado e em ordem aleatória, o conteúdo de cada um dos elementos de um vetor do tipo *int*, de tamanho variável, também digitado, executar a ordenação crescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação *Quick-Sort*.

76º) Elaborar um programa, em linguagem Java, capaz de receber, em ordem aleatória, o conteúdo de cada elemento de um vetor do tipo *double*, de tamanho 20, via teclado, executar uma ordenação decrescente do mesmo e apresentar o vetor ordenado em tela. Utilizar uma função de ordenação *Quick-Sort*.

77º) Elaborar um programa, em linguagem Java, capaz de informar quantas operações de *Quick-Sort* realizaremos no exercício no. 74 para ordenar o vetor e, através dele, concluir quantas são realizadas para o melhor e para o pior caso?

78º) Elaborar um programa, em linguagem Java, capaz de realizar a busca de um determinado caractere, digitado, em um vetor qualquer de tamanho 10, também com seus elementos do tipo *char* digitados, informando qual o índice do vetor que esse caractere se localiza. Utilizar ordenação do tipo *Quick-Sort* e busca binária recursiva.

Merge-Sort:

79º) Elaborar um programa, em linguagem Java, capaz de receber, em ordem aleatória, o conteúdo de cada elemento de um vetor do tipo *char*, de tamanho 5, via teclado, executar uma ordenação crescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação *Merge-Sort*.

80º) Elaborar um programa, em linguagem Java, capaz de receber, via teclado e em ordem aleatória, o conteúdo de cada um dos elementos de um vetor do tipo *int*, de tamanho variável, também digitado, executar a ordenação crescente do mesmo e apresentar o vetor ordenado em tela. Utilizar um método de ordenação *Merge-Sort*.

81º) Elaborar um programa, em linguagem Java, capaz de receber, em ordem aleatória, o conteúdo de cada elemento de um vetor do tipo *double*, de tamanho 20, via teclado, executar uma ordenação decrescente do mesmo e apresentar o vetor ordenado em tela. Utilizar uma função de ordenação *Merge-Sort*.

82º) Elaborar um programa, em linguagem Java, capaz de informar quantas operações de *Merge-Sort* realizaremos no exercício no. 74 para ordenar o vetor e, através dele, concluir quantas são realizadas para o melhor e para o pior caso?

83º) Elaborar um programa, em linguagem Java, capaz de realizar a busca de um determinado caractere, digitado, em um vetor qualquer de tamanho 10, também com seus elementos do tipo *char* digitados, informando qual o índice do vetor que esse caractere se localiza. Utilizar ordenação do tipo *Merge-Sort* e busca binária recursiva.