

BLOCO 3 – Estruturas de Dados Indexadas (3 semanas) – PL7

ASSUNTO – Estruturas de Dados indexadas

Objetivos Específicos:

- Mediante a apresentação dum problema, os alunos deverão ser capazes de o analisar, conceber e descrever o algoritmo estruturado em módulos e utilizar *arrays* monodimensionais.
- Desenvolver métodos de manipulação de *arrays monodimensionais*.

Conteúdo da aula

Exercício 1 (*)

Considerando o seguinte programa:

```
import java.util.Scanner;

public class Enigma {
    public static void main(String[] args) {
        int i, s=0, c=0;
        int[] v = new int[10];
        Scanner ler = new Scanner(System.in);

        for(i = 0; i < v.length; i++)
            v[i]=ler.nextInt();

        for(i = 0; i < v.length; i++)
            if (v[i] % 2 == 0){
                s = s + v[i];
                c++;
            }
        if(c!=0)
            System.out.println(((double)s)/c);
        else
            System.out.println("Operação impossível de realizar");
    }
}
```

BLOCO 3 – Estruturas de Dados Indexadas (3 semanas) – PL7

- a) Descreva a sua funcionalidade;
- b) Acrescente ao programa um método para receber um vetor de inteiros e retornar o menor número armazenado nesse vetor;
- c) Altere novamente o programa de forma a mostrar os índices dos menores elementos do vetor v, usando o método da alínea anterior.

Exercício 2 (**)

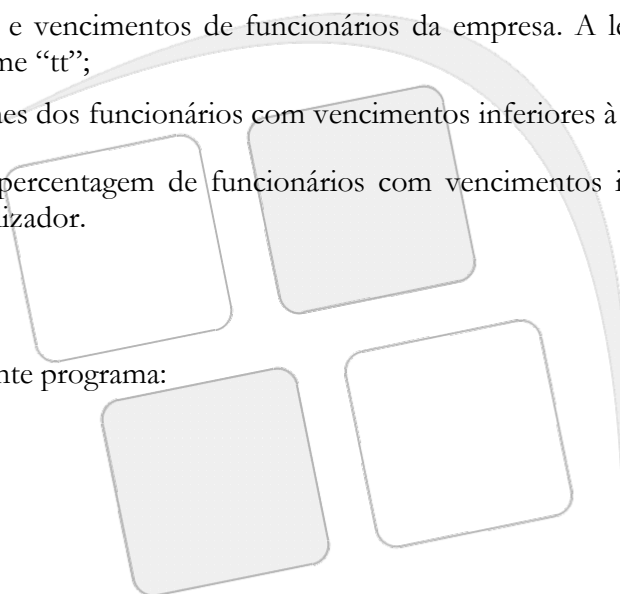
Pretende-se uma aplicação modular para determinar algumas estatísticas sobre vencimentos de funcionários duma empresa. O número de funcionários varia ao longo do tempo mas não é superior a 50.

O programa deve ter as seguintes funcionalidades:

- a) Leitura de nomes e vencimentos de funcionários da empresa. A leitura deve terminar com a introdução do nome “tt”;
- b) Listagem dos nomes dos funcionários com vencimentos inferiores à média;
- c) Apresentação da percentagem de funcionários com vencimentos inferiores a um dado valor fornecido pelo utilizador.

Exercício 3 (*)

Considerando o seguinte programa:



BLOCO 3 – Estruturas de Dados Indexadas (3 semanas) – PL7

```
import java.util.Scanner;
public class Enigma {
    public static void main(String[] args) {
        int n=0;
        String nomes[] = new String[100];
        Scanner ler = new Scanner(System.in);
        String m=" 1-Ler  Nomes\n2-Enigma  Nome\n3-Terminar\n\nEscolha  uma
opção:";
        char op;
        do {
            System.out.println(m);
            op = ler.next().charAt(0);
            switch (op) {
                case '1':
                    n = lerNomes(nomes);
                    break;
                case '2':
                    System.out.println("Nome:");
                    String nome = ler.nextLine();
                    n = enigma(nomes,nome,n);
                    break;
                case '3':
                    break;
                default:
                    System.out.println("Opção inválida!!");
            }
        } while (op != '3');
    }

    private static int lerNomes(String[] vec) {
        // Lê uma sequência de nomes terminada com a palavra FIM.
        // Armazena os nomes em vec e retorna o número desses nomes.
    }

    private static int listar(String[] vec, int n) {
        // Apresenta os primeiros n elementos de vec
    }

    private static int enigma(String[] nomes, String nome, int n) {
        int i=0;
        while (i<n && !nomes[i].equalsIgnoreCase(nome)) {
            i++;
        }
        if(i==n)
            return n;
        else{
            for (int j = i; j < n-1; j++)
                nomes[j]=nomes[j+1];
            return --n;
        }
    }
}
```

BLOCO 3 – Estruturas de Dados Indexadas (3 semanas) – PL7

- Descreva a sua funcionalidade;
- Complete os métodos `lerNomes` e `listar`;
- Corrija todos os aspetos que considerar relevantes.

Exercício 4 (***)

Elabore um programa modular que tenha as seguintes funcionalidades:

- Leitura de N números inteiros para um vetor, sendo N definido pelo utilizador;
- Inversão da ordem dos elementos do vetor;

Exemplo:

--	--	--

 →

--	--	--

- Apresentação do vetor invertido;
- Rotação para a direita dos elementos do vetor invertido;

Exemplo:

--	--	--

 →

--	--	--

- Apresentação do vetor rodado.

Exercício 5 (***)

Pretende-se um programa modular, em código Java, que leia uma sequência de nomes completos de pessoas, terminada com o texto “FIM”. Considere que a sequência nunca poderá ter mais de 100 nomes. No final a aplicação deve apresentar ao utilizador o seguinte:

- Uma listagem de todos os nomes lidos;
- Uma listagem apenas dos nomes com apelidos (último nome) começados por S;
- A percentagem de nomes cujos apelidos começam por S.

Além do método `main` implemente os seguintes métodos:

- `lerNomes`** que lê uma sequência de nomes completos de pessoas, terminada com o texto “FIM”, e retorna o número de nomes lidos. O vetor onde são guardados os nomes é passado por parâmetro;
- `apelido`** que recebe por parâmetro um nome completo e retorna o respetivo apelido;
- `mostrarListagem`** que mostra no ecrã uma listagem do conteúdo de um vetor de strings. Este vetor e a quantidade de elementos listados são passados por parâmetros.
- `preencherVetorApelidosS`** que recebe dois vetores de nomes (`v1` e `v2`) e preenche o vetor `v2` com os nomes do vetor `v1` que têm apelidos começados por S. No final, este método deve retornar a quantidade de nomes com apelidos começados por S.

BLOCO 3 – Estruturas de Dados Indexadas (3 semanas) – PL7

Exercício 6 (***)

Pretende-se uma aplicação modular que permita fazer a gestão de visitantes num hospital. Os visitantes são identificados pelo seu nome e o número máximo permitido é 100. O programa deve ser orientado por um menu que ofereça as seguintes funcionalidades:

- Inserir um visitante;
- Listar todos os visitantes;
- Atualizar um nome dado;
- Eliminar um visitante dado;
- Listar os nomes começados por uma dada letra;
- Listar nomes repetidos.

Exercícios Complementares

Exercício C1 (**)

Elabore uma aplicação modular que leia dois conjuntos de números inteiros para dois vetores e mostre o conjunto intersecção e união. Por definição, um conjunto não contém elementos repetidos.

Exercício C2 (***)

Crie uma classe contendo um método que, dado um vetor de inteiros, imprime o segmento de soma máxima.

Exemplo: No vetor [-1, 5, -4, 7, 2, -3] o segmento de soma máxima é [5, -4, 7, 2].

Exercício C3 (**)

Pretende-se dividir o conteúdo dum vetor em duas partes para que a soma de cada uma das partes seja o mais próximo possível. Com este objetivo, elabore um programa que efetue o preenchimento dum vetor com números inteiros positivos. De seguida, determine qual o ponto ideal de separação e coloque em dois vetores distintos os elementos que compõem cada uma das partes separadas. Finalmente, visualize os elementos de cada uma das partes e o respetivo somatório.

Exemplo: [5,100,0,1,5,2,3,10,20]

[5,100] : 105

BLOCO 3 – Estruturas de Dados Indexadas (3 semanas) – PL7

[0,1,5,2,3,10,20] : 41

Exercício C4 (***)

Construir um programa que permita jogar com o computador o jogo do “Master Mind”, o número de vezes que quiser.

O jogo consiste no seguinte: o computador gera N dígitos e o utilizador tenta identificar os dígitos gerados e respetivas posições. O computador em cada tentativa indica quantos dígitos estão corretos e nas posições corretas e quantos se encontram deslocados, isto é, os dígitos da tentativa que existem no número gerado mas a posição não é a correta.

Exemplo:

Numero Gerado : 9 5 4 3 2

Tentativa1 : 8 3 4 5 5 resposta do computador: Certo =1
Deslocados=2

Tentativa2 : 8 4 5 5 5 resposta do computador: Certo =0
Deslocados=2

