Lista de Exercícios – Lógica de Programação

Vetores

1. **Inversão de uma série de números** – Crie um programa que permita a um usuário informar 12 números quaisquer e, em seguida, os exiba na ordem inversa. Após isso, modifique o programa para que o usuário possa continuar informando uma quantidade qualquer de números, até que um valor de sentinela seja inserido (se um valor sentinela não for informado, inverta a sequência se a quantidade de números informados pelo usuário chegar a 12).

2. **Desvios em relação à média** – Crie um programa que permita ao usuário informar 12 números e calcule a média deles. Ao final, exiba cada um dos números e a sua diferença em relação à média.
   Posteriormente, modifique o programa para que o usuário possa inserir uma quantidade qualquer de números, até que um valor de sentinela seja inserido ou o usuário informar 12 números.

3. **Maior e menor valor** – Construa um programa onde o usuário possa informar 12 números e, na sequência, exiba todos os números informados, o maior e o menor.

4. **Livraria infantil** – Numa livraria especializada em crianças, é necessário armazenar o número de livros que cada autor escreveu (de 1 a 50) e a idade do seu público alvo (de 0 a 16). Por exemplo, um determinado autor pode ter escrito 4 livros destinado a crianças de 12 anos, outro pode ter escrito 9 livros para crianças de 6 anos e assim por diante.
   Baseado nesse relato, crie um programa que aceite continuamente o nome de um autor, o número de livros escritos e a idade do público alvo até que um valor sentinela seja inserido. Em seguida, exiba uma lista de quantos participantes escreveram cada número de livros (de 1 a 50). Por exemplo, nenhum autor escreveu apenas 1 livro, 4 autores escreveram 2 livros, nenhum autor escreveu 3 livros, 7 autores escreveram 4 livros, etc.
   Feito isso, modifique o programa para ele imprima uma lista de quantos participantes escreveram de 1 a 5 livros, de 6 a 12 livros e de 13 a 50 livros.
   Finalmente, altere mais uma vez o programa para que seja exibida uma listagem com o número de livros escritos para cada um dos seguintes grupos etários: até 3 anos, de 3 a 7 anos, de 8 a 10 anos, de 11 a 13 anos e de 14 a 16 anos.

5. **Folha de pagamento** – Crie um programa que calcule a folha de pagamento de uma empresa, conforme as instruções a seguir:

a) O usuário pode inserir continuamente os nomes dos empregados até que um valor de sentinela apropriado seja informado;
b) Após informar o nome de cada empregado, o usuário deverá informar o valor do salário da hora trabalhada desse empregado e quantas horas ele trabalhou;
c) O programa deve calcular o salário bruto de cada empregado, a percentagem de imposto retido na fonte (com base na tabela abaixo), o valor do imposto retido na fonte e o salário líquido (pagamento bruto menos imposto retido na fonte);
d) Depois que o usuário inserir os dados do último empregado, o programa deve exibir o salário bruto, salário líquido, percentual de imposto e valor do imposto para cada funcionário;
e) Por último, o programa deve exibir a soma de todas as horas trabalhadas, o total da folha de pagamento bruta, o total de imposto e a folha de pagamento líquida total.

Percentuais de imposto	
Salário bruto	Percentual
Até R$ 2.999,99	10%
Entre R$ 3.000,00 e R$ 5.499,99	13%
Entre R$ 5.500,00 e R$ 7.999,99	16%
Acima de R$ 8.000,00	20%

6. **Total de vendas de uma loja** – Escreva um algoritmo que pergunte ao usuário qual foi o total vendido (em reais) a cada dia da semana (identificado por um número de 1 a 7), armazenando os dados num vetor. Ao final, utilize esse vetor para determinar o total vendido na semana e escreva esse valor na tela.
Feito isso, modifique o algoritmo para que, ao perguntar o total vendido em cada dia da semana, ele informe de qual dia se trata, por extenso (“segunda feira”, “terça feira”, “quarta feira”, etc).

7. **Gerador de Números de Loteria –** Crie um algoritmo que gere o número de um bilhete de loteria de acordo com as seguintes instruções:
   a) O número deverá ter 7 dígitos;
   b) Cada dígito deve ser armazenado numa posição de um vetor criado por você (ou seja, esse vetor terá que ter 7 posições);
   c) Para gerar o número, você deve preencher cada posição do vetor com um número aleatório entre 0 e 9;
   d) Ao final, exiba o número gerado.

   Dica: para gerar um dígito aleatório, utilize a função `random(9)`. Para que essa função gere números verdadeiramente aleatórios, utilize a procedure `randomize` logo no início do programa (antes do laço que você utilizará para preencher o vetor com o número gerado).

8. **Estatísticas pluviométricas** – o *índice pluviométrico* é uma medida em milímetros que resulta da quantidade total da precipitação de água num

determinado local. Sabendo disso, escreva um algoritmo que permita ao usuário informar o índice pluviométrico de cada mês do ano. Após ler esses dados, o algoritmo deve exibir qual foi a precipitação total no ano (ou seja, a soma do índice pluviométrico de cada mês), a média de precipitação e quais foram os meses com os valores mais alto e mais baixo de precipitação (ou seja, quais foram os meses em que choveu mais e menos no ano).

9. **Estatística descritiva –** Crie um algoritmo que leia continuamente vários números informados pelo usuário e os armazene num vetor, até que o usuário digite um valor de sentinela (como -999) ou tenha informado 20 números (o que acontecer antes). Depois disso, exiba as seguintes estatísticas:
   a) O menor número do vetor;
   b) O maior número;
   c) A quantidade total de números que o usuário informou;
   d) A média dos números fornecidos.

10. **Validação de número de conta -** Crie um algoritmo que peça ao usuário para fornecer um número de conta e determine se esse número é válido. Para ser válido, esse número deve ser igual a um dos listados abaixo:

```
5658845 4520125 7895122 8777541 8451277 1302850
8080152 4562555 5552012 5050552 7825877 1250255
1005231 6545231 3852085 7576651 7881200 4581002
```

    Se o número fornecido pelo usuário estiver nessa lista, o algoritmo deverá informar que o número é válido; caso contrário, deverá informar que é inválido.

11. **Quantidade de dias em cada mês –** Construa um algoritmo que mostre o número de dias em cada mês, imprimindo na tela um relatório como o mostrado abaixo:

```
Janeiro tem 31 dias.
Fevereiro tem 28 dias.
Março tem 31 dias.
Abril tem 30 dias.
Maio tem 31 dias.
Junho tem 30 dias.
Julho tem 31 dias.
Agosto tem 31 dias.
Setembro tem 30 dias.
Outubro tem 31 dias.
Novembro tem 30 dias.
Dezembro tem 31 dias.
```

    Para resolver esse problema, utilize dois vetores paralelos, um para o nome dos meses e outro para a quantidade de dias.

12. **Lista telefônica –** Utilizando dois vetores paralelos, crie um algoritmo que armazene o nome de 7 pessoas num deles e os números de telefone dessas pessoas no outro (essas informações devem ser fornecidas pelo usuário). Ao

final, o algoritmo deve perguntar ao usuário informe o nome de uma pessoa e exibir o número dela. Caso esse nome não exista, deve-se exibir uma mensagem de erro. Faça um laço para que o usuário possa indicar se deseja continuar pesquisando por mais números de telefone ou se deseja encerrar o algoritmo.

13. **Folha de pagamento** – Imagine que você esteja desenvolvendo um programa para processar a folha de pagamento de uma empresa que possui 10 empregados. Para isso, você deve utilizar vetores paralelos com 10 posições para armazenar as seguintes informações:

    Vetor 1 - Código do Empregado: códigos numéricos para identificar cada um dos empregados. Esses códigos são informados abaixo:

    ```
    56588 45201 78951 87775 84512
    13028 75804 92057 69403 29625
    ```

    Vetor 2 - Horas Trabalhadas: o número de horas trabalhadas por cada um dos 10 empregados num determinado mês;

    Vetor 3 - Valor da Hora Trabalhada: o valor da hora trabalhada de cada empregado;

    Vetor 4 - Valor dos Salários: o salário bruto de cada empregado.

    Note que os vetores devem ser paralelos, o que significa que o programa pode relacionar os dados dos diferentes vetores através dos índices. Por exemplo, o valor armazenado na posição 5 do vetor 2 é o número de horas trabalhadas pelo empregado cujo código está armazenado na posição 5 do vetor 1. O valor da hora trabalhada desse mesmo empregado deve ser armazenado na posição 5 do vetor 3.

    O programa deve exibir o código de cada funcionário e solicitar que o usuário informe as horas e o valor da hora trabalhada desse empregado. Após isso, o programa deve calcular os salários brutos desse empregado (número de horas trabalhadas multiplicado pelo valor da hora), armazenando-os no vetor 4.

    Ao final, o programa deverá exibir o código de cada empregado e os salários brutos deles.

    Modificação: considere que o valor da hora trabalhada depende da função de cada empregado. Nessa empresa existem as seguintes funções (o número à esquerda é o código da função):

    ```
    001-Programador     002-Analista   003-DBA
    004-Arquiteto       005-Webmaster  006-Designer
    ```

    Considerando isso, modifique o programa que você fez para que, antes de iniciar o processamento dos salários, o usuário informe qual é o valor da hora de cada função. Depois disso, ao invés de informar o valor da hora trabalhada dos empregados, o usuário informará somente o código da função deles, e o próprio programa deve determinar qual é o valor dessa hora e o salário bruto.

    Ao final, o programa deve imprimir uma listagem com o código de cada empregado, sua função, o número de horas trabalhadas e o seu salário bruto.

14. **Exame de Habilitação para CNH** – Para tirar a Carteira Nacional de Habilitação (CNH), é necessário realizar um exame teórico, onde o candidato responde a 20

questões objetivas de uma prova, cada uma delas possuindo 4 alternativas (“A”, “B”, “C”, “D” e “E”) como resposta. Imagine que você esteja escrevendo um programa para automatizar essa prova e, para isso, precisa pedir que o candidato informe qual é a resposta que ele deu para cada uma das questões. As respostas corretas são dadas abaixo:

```
01.B  06.A  11.B  16.C
02.D  07.B  12.C  17.C
03.A  08.A  13.D  18.B
04.A  09.C  14.A  19.D
05.C  10.D  15.D  20.A
```

O seu programa deve armazenar as respostas corretas num vetor e as respostas do candidato em outro vetor. Quando o usuário acabar a prova, o programa deve corrigi-la, dizendo se o usuário foi aprovado ou não (é necessário acertar no mínimo 15 questões para ser aprovado) e exibir uma listagem das questões que foram respondidas incorretamente.

15. **Jogo da Velha** – Crie um programa que permita a duas pessoas jogar o jogo da velha. Para isso, utilize uma matriz (ou seja, um vetor bidimensional) com três linhas e três colunas, como no tabuleiro do jogo da velha. Cada elemento da matriz deveria ser inicializado com um asterisco (“*”) e o programa deveria executar um laço para implementar o seguinte comportamento:
    a) O tabuleiro deve ser mostrado na tela (ou seja, a matriz deve ser impressa);
    b) O primeiro jogador deve selecionar uma posição no tabuleiro para marcar sua posição com um “X”. Para isso, o programa deve perguntar a esse jogador qual é a linha e a coluna da posição em que ele pretende jogar;
    c) Da mesma forma que o primeiro jogador, o programa deve perguntar ao segundo jogador qual é a linha e a coluna da posição em que ele pretende marcar com um “O”;
    d) O programa deve verificar se algum dos jogadores ganhou ou se houve um empate e, caso uma dessas situações tenha ocorrido, o programa deveria informar isso e encerrar. Naturalmente, em caso de uma vitória, o programa informa quem ganhou;
    e) O primeiro jogador ganha a partida se conseguir marcar três “X” numa única linha ou coluna, ou então na diagonal do tabuleiro. O segundo jogador ganha se fizer o mesmo com três “O”. Um empate ocorre quando todas as posições do tabuleiro estão preenchidas sem que ninguém tenha vencido.

16. **Quadrado mágico** – Na matemática, um “quadrado mágico” é uma estrutura matricial que contém um número n de linhas e colunas. Cada uma das posições da estrutura contém um número inteiro único (ou seja, que não se repete nas demais posições), que varia entre 1 e n^2. Além disso, a soma de cada linha, cada coluna e cada diagonal somam o mesmo número.

Baseado nessas informações, crie um programa que permita ao usuário inicializar uma matriz com números inteiros quais quer e, ao final, informe se a matriz forma um quadrado mágico.

Exemplo: abaixo é mostrado um quadrado mágico 3x3 (ou seja, n = 3) que armazena os números inteiros de 1 a 9. Repare que ao somarmos os elementos de uma linha, coluna ou diagonal, o resultado é sempre o mesmo (15), o que comprova que se trata efetivamente de um quadrado mágico.