UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
DISCIPLINA: Estrutura de Dados I
PROFESSOR: Bruno Monteiro
UNIDADE 1 – PRÁTICAS PROPOSTAS **Versão: 27/11/2023**

Para cada um dos problemas abaixo, desenvolva um programa na linguagem Java:

1. Inicialize variáveis adequadas para: sua idade, sua altura, primeira letra do seu nome, seu nome completo. Depois exiba os dados na tela.
2. Atribua com valores vindos do teclado variáveis adequadas para: sua idade, sua altura, primeira letra do seu nome, seu nome completo. Depois exiba os dados na tela.
3. Receba do teclado dois números inteiros, calcule e exiba os resultados das seguintes operações: adição, subtração, multiplicação, quociente da divisão e o resto da divisão.
4. Calcular e exibir a média aritmética de três números inteiros.
5. Calcular e exibir a média aritmética de três números reais.
6. Construa um programa que gerencia a conta de uma pizzaria.
   Preço do refrigerante: R$ 1,50. Preço da fatia de pizza: R$ 3,00. Taxa do garçom: 10%.
   Receba do usuário: a quantidade de refrigerantes, a quantidade de fatias e a quantidade de pessoas na mesa.
   Calcule e exiba: o total sem a taxa, o total com a taxa, o rateio por pessoa com a taxa.
7. Calcular uma divisão entre dois números reais. Produza um alerta em caso de divisão por zero.
8. Calcular o IMC (índice de massa corporal) de uma pessoa. Produza um alerta em caso de divisão por zero.
9. Construa um programa que calcule a área de um círculo, tendo como entrada o valor do raio, que deve ser positivo. Use o valor da constante PI vindo da biblioteca matemática.
10. Construa um programa que leia um número inteiro e identifique se ele é par ou ímpar.
11. Construa um programa que simule uma transferência bancária, entre duas contas. Primeiro, inicialize cada conta com R$ 100. Depois, permita que o usuário defina quanto deve transferir, da conta1 para a conta2, porém a transferência só deve ser realizada caso haja saldo suficiente.
12. Distinguir, com base na média parcial do aluno, se ele está aprovado, reprovado ou na final. Aplique as regras da UFERSA.
13. Construa um programa que calcule para o aluno sua média parcial e informe sua situação parcial (Aprovado, Recuperação ou Reprovado). Caso ele esteja em Recuperação, calcule quanto ele precisa tirar na 4ª prova para ser aprovado (média final maior ou igual que 5,0). Observação: utilize os pesos e regras da UFERSA.
14. Construa um programa que calcule uma equação do 2º grau.
    Crie uma função com retorno para calcular o delta.
15. Construa um programa que leia um número inteiro digitado pelo usuário. Caso o número pertença ao intervalo de 1 a 5, exiba o número por extenso. Caso o número não pertença a este intervalo, exiba a mensagem "valor invalido".
16. Construa um programa que leia do usuário um número inicial e um número final. Em seguida, exiba na tela uma sequência com os números desse intervalo informado pelo usuário. Exemplo: caso o usuário entre com os números 4 e 10, o resultado do programa seria: 4 5 6 7 8 9 10
    a) Construa este programa utilizando a estrutura **while**.
    b) Construa este programa utilizando a estrutura **do-while**.
    c) Construa este programa utilizando a estrutura **for**.
17. Construa um programa que leia do usuário um número inicial e um número final. Em seguida, exiba na tela uma sequência apenas com os números ímpares dentro deste intervalo informado pelo usuário. Exemplo: caso o usuário entre com os números 4 e 10, o resultado seria: 5 7 9
    a) Construa este programa utilizando a estrutura **while**.
    b) Construa este programa utilizando a estrutura **do-while**.
    c) Construa este programa utilizando a estrutura **for**.
18. Verificar se a senha, informada durante a execução, é correta. Quando a senha estiver correta, exiba “senha correta” e o programa é encerrado. Quando a senha estiver errada, exiba “senha incorreta”, e permita a entrada novamente da senha, repetindo esse processo até que a senha informada seja correta.
    a) Resolva esse problema utilizando a estrutura **while**.
    b) Resolva esse problema utilizando a estrutura **do-while**.
19. Construa um programa para exibir a tabuada de qualquer número “n” (1 a 9), sendo “n” um número fornecido pelo usuário. Utilize estrutura de repetição.
20. Construa um programa que calcule o rendimento mensal de um investimento em poupança. Variáveis: investimento inicial, investimento mensal, quantidade de meses, saldo acumulado, taxa de juros mensal, rendimento mensal.
21. Construa um programa que identifique se um número é primo.
22. Construa um programa que calcule o somatório dos números inteiros de um intervalo, definido por um número inicial e um número final. Exemplo: caso as entradas fossem 4 e 9, o resultado seria: 39

23. Construa um programa que calcule o número fatorial de um número. Use uma estrutura de repetição. Fatorial: $n! = n (n - 1)!$
    Exemplo: $5! = 5 \times 4 \times 3 \times 2 \times 1 = 120$ ou $5! = 1 \times 2 \times 3 \times 4 \times 5 = 120$

24. Construa um programa que simule uma calculadora. Disponibilize um menu de opções e simule a opção desejada, exibindo novamente o menu, até que o usuário escolha sair. Menu de opções:
    1 - potenciação — Dica: utilize a função da biblioteca matemática.
    2 - raiz quadrada — Dica: utilize a função da biblioteca matemática.
    3 - fatorial — Dica: crie e utilize uma função com a solução da questão anterior.
    0 - sair

25. Construa um programa para ler e exibir um vetor de inteiros. Em tempo de execução, o usuário pode definir o tamanho do vetor.

26. Construa um programa para ler e exibir uma matriz de inteiros. Em tempo de execução, o usuário pode definir o tamanho da matriz.

27. Construa e use uma função que produz um novo vetor de inteiros com a ordem inversa do vetor original passado por parâmetro.

28. Construa um programa que identifique o maior e o menor número de um vetor de inteiros.

29. Construa um programa que mova o número da última posição de um vetor para a primeira posição. Faça isso gradativamente: use uma estrutura de repetição, e em cada iteração do loop mova esse número apenas uma posição, ou seja, troque esse número da posição n por n-1.

30. Construa um programa que inverta a frase digitada pelo usuário. Por exemplo, se string1 for “bom dia”, então string2 será “aid mob”.

31. Construa um programa que registre objetos da classe Pessoa, com os seguintes atributos: cpf, nome, idade, sexo, peso, altura, imc. Permita que o usuário defina a quantidade de pessoas em tempo de execução. Ao cadastrar uma pessoa, calcule o IMC (Índice de Massa Corporal). Ao final, exiba a lista de pessoas (com seus respectivos atributos).

32. Evolua a questão anterior. Permita que o programa salve o resultado em um arquivo binário. Permita que o programa abra o arquivo binário e exiba o conteúdo na tela.

33. Construa um programa que conte a quantidade de letras de uma palavra.
    Exemplo: “casa” Resultado: c=1 a =2 s =1

34. Construa um programa que conte a quantidade de vezes que as palavras de uma frase aparecem.
    Exemplo: “estude muito sempre sempre estude” Resultado: estude=2 muito=1 sempre=2

35. Evolua a questão anterior. Permita que o programa salve o resultado em um arquivo de texto CSV. Permita que o programa abra o arquivo de texto CSV e exiba o conteúdo na tela.

36. Construa e use uma função para calcular o número fatorial de um número utilizando recursividade.

37. Calcule o somatório dos números inteiros do intervalo entre dois números. No mesmo programa, implemente esse cálculo em duas funções:
    a) Resolva o problema com estrutura de repetição.
    b) Resolva o problema com recursividade.
    c) Analise os pontos positivos e negativos de cada versão.

38. Utilizando o recurso Generic de Java, faça um método para exibir um objeto (usando seu método toString), e outro método para exibir os elementos de um vetor.

39. Utilizando o recurso Generic de Java, construa um programa que tenha funcionalidades de CRUD (create, read, update e delete) que se adeque para diferentes classes (ex: String, Pessoa, Produto, etc).

40. Construa um programa CODIFICADOR que receba um arquivo de texto de entrada e codifique ele usando um padrão de troca de letras. Após esse processamento, gere um arquivo codificado.
    Construa outro programa, que funcionará como DECODIFICADOR, que seja capaz de ler o arquivo codificado e produzir um arquivo de texto decodificado, que deve ser o mesmo texto original.
    Padrão de troca de letras para codificar um texto: $Z \Leftrightarrow P$ $E \Leftrightarrow O$ $N \Leftrightarrow L$ $I \Leftrightarrow A$ $T \Leftrightarrow R$

41. Utilizando a Collection de Java, faça exemplos de uso das classes abaixo e explicite suas diferenças:

    a) List: ArrayList, Vector, LinkedList
    b) Set: HashSet, LinkedHashS, TreeSet
    c) Queue: PriorityQueue, LinkedList
    d) Deque: LinkedList
    e) Map: HashMap, LinkedHashMap, TreeMap