

Lógica de Programação
Atividade Avaliativa 1

RESPOSTAS

Avaliação individual. A interpretação das questões faz parte da avaliação. Não é permitida conversa bem como troca de materiais entre os alunos;

1) A velocidade média de um veículo é dado pela expressão $V_m = DS / Dt$, onde:
DS: variação de espaço (ponto de chegada – ponto de partida) em quilômetros
Dt: intervalo de tempo (tempo final – tempo inicial) em horas

a) Quais são as variáveis com os respectivos tipos de dados necessários para que possamos calcular a velocidade média de um veículo para percorrer um determinado percurso?

float pc, pp, tf, ti, vm;

b) Escreva uma expressão computacionalmente válida para resolver questão anterior.

Vm = (pc – pp) / (tf – ti);

2) O índice de massa corporal (IMC) é uma fórmula utilizada para verificar se um adulto está acima do peso, obeso ou abaixo do peso ideal considerado saudável. A fórmula utilizada para calcular o IMC é dada pela expressão: $IMC = peso / (altura * altura)$.

a) Quais são as variáveis com os respectivos tipos de dados necessários para que possamos calcular o IMC de uma pessoa?

float imc, peso, altura;

b) Escreva uma expressão computacionalmente válida para resolver a questão anterior.

*imc = peso / (altura * altura);*

3) O salário mínimo é reajustado anualmente. Para isso, o governo determina o percentual de reajuste.

a) Quais são as variáveis com os respectivos tipos de dados necessários para que possamos calcular o valor do salário reajustado.

float salario, reajuste, salario_reajustado;

b) Escreva a expressão para calcular o valor do salário reajustado.

*salario_reajustado = salario + (salario * (reajuste/100));*

4) Assinale com C os nomes de variáveis corretos e com I os incorretos. Explique o que está errado nos incorretos.

(I) n#1

(I) \$din

(C) n1

(C) dep

(C) tempo

(I) n 1

(I) U F

(C) nome2

(C) n_1

(C) K2K

(I) 2nome

(I) val#r

5) Explique por que ao declaramos uma variável faz-se necessária a especificação do tipo de dados que terá.

R: Uma variável é uma posição de memória utilizada para armazenar um valor. A linguagem C é uma linguagem fortemente tipada, ou seja, onde a declaração do tipo é obrigatória. Todas as variáveis que serão utilizadas durante o programa precisam ser declaradas antes de seu primeiro uso no algoritmo. É através do tipo que o compilador sabe quanta memória deve ser reservada para uma determinada variável, pois cada tipo ocupa uma certa quantidade de memória.

6) Quando devemos utilizar os operadores relacionais?

R: Os operadores relacionais estabelecem uma relação entre dois elementos, retornando verdadeiro ou falso. São utilizados em expressões condicionais para a comparação do valor de duas expressões.

7) Ao escrever uma expressão aritmética, por exemplo, devemos considerar a precedência dos operadores. Explique o que é a precedência e como podemos impor a precedência de alguma operação.

R: Precedência é a prioridade com que os operadores são executados pelo compilador. Caso os operadores tenham o mesmo nível de precedência eles são analisados da esquerda para a direita. A precedência dos operadores pode ser mudada utilizando parênteses.

8) Considere a expressão a seguir e indique a ordem em que serão realizadas as operações:

$$X = Y + 2 < j \parallel d > 5 \&\& k < > 3$$

R: 1º. Operações aritméticas

2º. Operações relacionais

3º. Operações lógicas

9) Qual a função dos operadores lógicos?

R: Os operadores lógicos são utilizados para conectar expressões lógicas sendo geralmente utilizados em expressões condicionais.

10) Preencha as tabelas verdades

A	B	A && B
V	V	V
V	F	F
F	V	F
F	F	F

A	B	A B
V	V	V
V	F	V
F	V	V
F	F	F

A	!A
V	F
F	V

11) Dadas as expressões a seguir, identifique o resultado verdadeiro ou falso que cada uma delas retornaria, em função dos valores dados.

A= 2; B= 7;

Exemplo:

(A == 2) && (B == 5)
V && F => F

a) (A == 3) && (B == 7)
F && V => F

b) (A < 3) || (B != 7)
V || F => V

c) (A <= 2) && (B == 7)
V && V => V

d) !(A == 2) && (B == 7)
!V && V
F && V => F

e) (A >= 1) && (A <= 10) || (A >= 5)
V && V || F => V

f) (A < 5) && (B > 2) || (B != 7)
V && V || F => V

12) Verifique se as afirmações a seguir são verdadeiras ou falsas:

(F) Em uma expressão que utiliza operadores aritméticos, lógicos e relacionais, são realizadas primeiro as comparações.

(V) Ao avaliar uma expressão que faz uso do operador lógico AND o resultado será verdadeiro apenas quando todas as expressões envolvidas resultarem verdadeiro.

(V) Nos algoritmos, as variáveis são utilizadas para representar valores.

(F) Como regra, os nomes de variáveis devem iniciar com um número.

13) Sobre o comando **switch-case**, marque V ou F:

(F) O teste da variável só é permitido com variáveis do tipo **int** ou **float**

(F) O comando que segue os dois pontos do **case** não pode ser nem **if** nem **if-else**

(F) Nos rótulos dos casos podem ser utilizadas expressões

(V) Não é obrigatório o uso do comando **default**

(V) É facultativo o uso do comando **break**

14) Considere o trecho de programa abaixo:

```
#include <stdio.h>

int main(){
    int numero;
    printf("Digite um numero entre 0 e 9: ");
    scanf("%d", &numero);
    switch(numero)
    {
        case 0: printf("0");
        case 1: printf("1");
        case 2: printf("2");
        case 3: printf("3");
        case 4: printf("4");
        case 5: printf("5");
        case 6: printf("6");
        case 7: printf("7");
        case 8: printf("8");
        case 9: printf("9");
    }
    return 0;
}
```

Defina a saída na tela quando os valores de entrada digitados pelo usuário forem:

- a) 5 : 56789
- b) 9 : 9
- c) 0 : 0123456789

15) Escreva um programa para ler um número e verificar se ele é um número **quadrado perfeito**. Um número é quadrado perfeito quando tem um número inteiro como resultado da sua raiz quadrada, e esse resultado pode ser escrito como o seu quadrado.

Ex: $\sqrt{4} = 2$ e $2^2 = 4$ \therefore 4 é quadrado perfeito , $\sqrt{25} = 5$ e $5^2 = 25$ \therefore 5 é quadrado perfeito

$\sqrt{8} = 2,83$ \therefore 8 NÃO é quadrado perfeito , $\sqrt{15} = 3,87$ \therefore 15 NÃO é quadrado perfeito

```
#include <stdio.h>
#include <math.h>

int main()
{
    int num, raiz;

    printf("Digite um numero: ");
    scanf("%d", &num);
    raiz = sqrt(num);
    if (raiz*raiz == num)
        printf("É um número quadrado perfeito");
    else
        printf("Não é um número quadrado perfeito");

    return 0;
}
```

16) Escreva um programa em C que lê a idade de uma pessoa e informa a sua classe eleitoral. Utilize a estrutura **if-else-if**:

- Não-eleitor (abaixo de 16 anos);
- Eleitor facultativo (entre 16 e 18 anos e maior de 65 anos);
- Eleitor obrigatório (entre 18 e 65 anos).

```
#include <stdio.h>

int main()
{
    int idade;

    printf("Digite a sua idade: ");
    scanf("%d", &idade);
    if ((idade < 16) && (idade >= 0))
        printf("Não eleitor");
    else if (((idade >= 16) && (idade < 18)) || (idade > 65))
        printf("Eleitor facultativo");
    else if ((idade >= 18) && (idade <= 65))
        printf("Eleitor obrigatório");
    else
        printf("Idade inválida");

    return 0;
}
```

17) Faça um programa para ler um número de 1 a 12 referente ao mês do ano e informar o trimestre correspondente, sendo Jan/Fev/Mar o 1º. Trimestre, Abr/Maio/Jun o 2º. Trimestre, e assim sucessivamente. Se for um número incorreto, mostrar uma mensagem de erro. Utilize o **switch-case** com a opção de deixar **case's** em branco.

```
#include <stdio.h>

int main()
{
    int mes;

    printf("Digite o mês: ");
    scanf("%d", &mes);
    switch(mes)
    {
        case 1:
        case 2:
        case 3:
            printf("1o. Trimestre");
            break;
        case 4:
        case 5:
        case 6:
            printf("2o. Trimestre");
            break;
        case 7:
        case 8:
        case 9:
            printf("3o. Trimestre");
            break;
        case 10:
        case 11:
        case 12:
            printf("4o. Trimestre");
            break;
        default:
            printf("Mês inválido!");
    }
    return 0;
}
```