Lista 5 – estrutura de controle: repetição (sala de aula revisão)

Assunto:

Variáveis, instruções de entrada e saída de dados e expressões matemáticas, estruturas de decisão e de repetição.

Forma de resolução:

a) Se necessário:

a.1) representar o algoritmo em descrição narrativa, utilizando o seguinte procedimento para análise do problema e definição da solução:

a.1.1) Dados de entrada

a.1.2) Processamento (instruções a serem realizadas)

a.1.3) Dados de saída

a.2) representar o algoritmo em fluxograma.

b) Representar o algoritmo de solução do problema na linguagem C, com as entradas informadas pelo usuário.

Importante:

a) Chaves
   a. Elas delimitam um bloco de instruções (comandos) que estão subordinados a uma cláusula if, else ou else if ou a uma estrutura de repetição do, while e do while.
   b. As instruções delimitadas por um conjunto de chaves deve obrigatoriamente estar identada. Um conjunto de espaços (tab) de avanço.
   c. Cada chave aberta deve ter a sua correspondente fechada. Para facilitar, sempre, abra uma chave, feche-a e retorne para escrever o código (as instruções) dentro dessas chaves.
   d. Para haver um teste lógico é obrigatória a existência de if e vice-versa. O teste lógico é colocado dentro de parênteses.
b) Erros:
   a. Verificar se as chaves estão corretas
   b. Verificar se há teste lógico sem if.
   c. Loop infinito. A variável de controle não está sendo incrementada ou decrementada, o teste lógico que verifica a condição de continuidade ou parada não está sendo realizado corretamente.

Exercícios:

1) Ler um número que representa a quantidade de valores pares, divisíveis por 3 e não divisíveis por 5 que devem ser mostrados. Apresentar esses valores n por linha. n é informado pelo usuário e deve ser positivo. Os valores são apresentados separados por tabulação.

Por exemplo:

O usuário digita 7 (significa que ele quer visualizar os sete primeiros valores que atendem condições) e em seguida 5 (que significa a quantidade de valores por linha que devem ser mostrados).

Será mostrado:

6 12 18 24 36 //cinco valores por linha

42 48

2) Elaborar um programa que efetue a leitura de valores positivos inteiros até que um valor negativo seja informado. Ao final devem ser apresentados o maior e o menor valor informados pelo usuário. O valor negativo, a condição de saída, não deve ser considerado nas comparações para localizar o maior e o menor.

3) Elaborar um programa para implementar a operação de potência.

4) Escreva um programa que determine o valor de S, com n informado pelo usuário, da série S = 1/1 – 2/4 + 3/9 – n / n^2. Para n = 10:

S = 1/1 - 2/4 + 3/9 - 4/16 + 5/25 - 6/36 + ... - 10/100.

5) Escreva um programa que determine a soma dos n primeiros termos (informado pelo usuário) que inicia com 500 e se necessário vai para números negativos.

S = 2/500 - 5/450 + 2/400 - 5/350...

6) Fazer um programa para determinar o valor S da série:

S = (x +1 / y – 10) + (x + 2 / y – 9) + (x+3 / y – 8) + ... + (x+10 / y -1).

Onde x e y são valores informados pelo usuário.

7) Fazer um programa que calcule o valor de S com n informado pelo usuário, determinado pela série: S = (n-1 *n) / 1 + (n-2 * n-1) / 2 + (n-3 * n-2) / 3 + ... (1 * 2) / n-1. Para n = 38:

S = (37 * 38) / 1 + (36 * 37) / 2 + (35 * 36) / 3 + ... + (1 * 2) / 37

8) Qual o objetivo do algoritmo representado no trecho de código a seguir. Explique a forma de ação de cada estrutura de repetição.

```
for (Cont=1; Cont<=5; Cont++)
{
    do
    {
        printf("Informe a Idade:");
        scanf("%d",&Idade);
    } while (Idade < 0);

    do
    {
        printf("Informe o tipo, (E)estudante/(P)Professor:");
        fflush(stdin);
        scanf("%c",&Tipo);
    }while(Tipo !='E' && Tipo!='e' && Tipo!='P' && Tipo!='p');

    if (Tipo=='p' || Tipo =='P')
    {
        do
        {
             printf ("Informe o salario:");
             scanf("%f",&Salario);
        }while (Salario<=0);
    }
}
```

9) Qual o objetivo do algoritmo representado no trecho de código a seguir.

```
do
{
   printf("Informe um valor positivo: ");
   scanf("%f",&Num);
   if (Num < 0)
   {
      Tentativas++;
   }
} while(Num < 0);
```

10) Qual o objetivo do algoritmo representado no trecho de código a seguir.

```
do
{
    printf("Informe um número entre 1 e 50: ");
    scanf("%d",&Num);
    if((Num < 1) || (Num > 50))
```

```
    {
        printf("valor informado fora da faixa.\n");
    }
}while((Num < 1) || (Num > 50));


while((Num * 2) < 250)
{
    Num = Num * 2;
    printf("%d\n",Num);
}
```

11) Elaborar um programa que solicita a idade e se é profissional autônomo ou funcionário, calcula e mostra:

a) o total de pessoas profissionais autônomos com menos de 18 anos;

b) a média das idades das pessoas funcionários com mais de 18 anos.

O programa deve validar as entradas: permitir somente valores positivos para a idade e A/a/F/f (autônomo ou funcionário) para o tipo de profissional.

O programa termina quando for informado um valor negativo para a idade, que não deve ser considerado. Portanto, ao ser informado um valor negativo para a idade não deve ser solicitado o tipo de profissional.

12) Elaborar um programa que valide o número de uma conta corrente com três dígitos e retorne o dígito verificador e informe se o número da conta está correto. O número da conta é informado da seguinte maneira:

1234. Sendo 123 o número da conta e 4 o dígito verificador.

1112. Sendo 111 o número da conta e 2 o dígito verificador.

Ler os quatro dígitos como um único número. Separe os dígitos de forma a que os três primeiros sejam o número da conta e o último o dígito verificador. O usuário deve informar um número com 4 dígitos. Valide a entrada, ou seja, repetir a entrada do número até que seja informado um número com quatro dígitos.

Para obter o dígito verificador:

a) somar o número da conta com o seu inverso = 123 + 321 = 444

b) multiplicar cada dígito pela sua ordem posicional e somar os resultados = 4 *1 + 4 * 2 + 4 * 3 = 24

c) O último dígito desse resultado é o verificador: 4

13) Implemente um algoritmo que lê números informados pelo usuário. O algoritmo deve parar quando informado um número negativo. Para cada número lido, exceto o número negativo que representa a saída, o algoritmo deve imprimir a quantidade de dígitos do número informado.

Desafio: elaborar o algoritmo para qualquer valor de entrada (inteiro válido).

Exemplo:

876654 = 6 dígitos

154 = 3 dígitos

8 = 1 dígito

0 = 1 dígito (cuidado com esse caso especial)

14) (Desafio) As instruções de programa a seguir podem ser alteradas de forma que a instrução *"if (Num != -99)"* não seja necessária? Se sim, como; e se não, por quê?

```
while (Num != -99)
{
    printf("Informe um número: ");
    scanf("%d",&Num);

    if (Num != -99)
    {
        if (Num < Menor)
        {
            Menor = Num;
        }
        else if (Num > Maior)
        {
            Maior = Num;
        }
    }
}
```