Lista 8 – estrutura de controle: repetição (revisão)

Assunto:

Variáveis, instruções de entrada e saída de dados e estruturas de decisão e de repetição, enfatizando validação de entradas e execução repetida do programa.

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:

Observações: Para todos os exercícios desta lista:

a) fazer os programas de forma que o usuário possa optar por repetir a execução dos mesmos. Ao final de uma execução questionar se o usuário quer continuar utilizando o programa. Se informado afirmativamente, repetir a execução. Isso pode ser feito utilizando uma estrutura do while que conterá a parte do programa que será repetida. As variáveis que não dependem de dados de entrada ou de execução podem ser declaradas fora dessa estrutura de repetição do programa. Atenção para a inicialização de variáveis: deve ser feita dentro dessa estrutura de repetição.

b) Validar as entradas.

c) Validar para que não sejam realizadas divisões por zero.

1) Ler dois números que representam os limites de um intervalo. Mostrar os pares e divisíveis por três desse intervalo, em ordem decrescente e em colunas ('n' números por linha separados por tabulação). 'n' é informado pelo usuário e deve ser validado para obter uma entrada positiva. Calcular e mostrar a média dos ímpares e não divisíveis por 5 desse intervalo. Validar para que não seja realizada uma divisão por zero.

2) Ler a idade, o tipo (E estudante e P professor) de cinco pessoas. Se o tipo for professor solicitar o salário. Fazer a média dos salários informados. Se o tipo for estudante solicitar se o mesmo recebe mesada. Contar quantos recebem e quantos não recebem mesada. Garantir que o usuário informe uma idade válida, ou seja, positiva, que o tipo seja E ou P e que recebe mesada seja S ou N. Validar essas entradas. Isso no sentido de ficar solicitando essas entradas até que sejam válidas. Validar para que não seja realizada uma divisão por zero.

3) Alice e Beto são amigos e sempre que se encontram relembram os tempos de infância tirando par-ou-ímpar para decidir quem escolhe o filme a ser assistido, ou qual o restaurante em que vão almoçar, etc. Escreva um programa para determinar que ganhou a série de par-ou-ímpar.

Entrada
A primeira entrada deve ser um valor que identifica a quantidade de jogos realizados, identificada pela variável 'n'. A seguir deve-se ler os 'n' resultados dos jogos. Se $R_i = 0$ significa que Alice ganhou o i-ésimo jogo e se $R_i = 1$ Beto ganhou o i-ésimo jogo.

Saída
O programa deve produzir uma linha na saída, no formato 'Alice ganhou X e Beto ganhou Y jogos'.
Veja a sugestão a seguir.

```
Informe o nro de jogos que joram realizados:
6

1o. resultado de 6: 0
2o. resultado de 6: 1
3o. resultado de 6: 1
4o. resultado de 6: 0
5o. resultado de 6: 1
6o. resultado de 6: 1


Alice ganhou 2 e Beto ganhou 4 jogos.
```

4) Fazer um programa que calcule o fatorial dos números inteiros de 1 até 12. Fazer a saída do programa, da parte do fatorial, como mostrado na figura a seguir. Colocar o símbolo de multiplicação somente entre dois números.

```
Informe a quantidade de n·meros para mostrar o fatorial: 12
1! => 1 = 1
2! => 2 * 1 = 2
3! => 3 * 2 * 1 = 6
4! => 4 * 3 * 2 * 1 = 24
5! => 5 * 4 * 3 * 2 * 1 = 120
6! => 6 * 5 * 4 * 3 * 2 * 1 = 720
7! => 7 * 6 * 5 * 4 * 3 * 2 * 1 = 5040
8! => 8 * 7 * 6 * 5 * 4 * 3 * 2 * 1 = 40320
9! => 9 * 8 * 7 * 6 * 5 * 4 * 3 * 2 * 1 = 362880
10! => 10 * 9 * 8 * 7 * 6 * 5 * 4 * 3 * 2 * 1 = 3628800
11! => 11 * 10 * 9 * 8 * 7 * 6 * 5 * 4 * 3 * 2 * 1 = 39916800
12! => 12 * 11 * 10 * 9 * 8 * 7 * 6 * 5 * 4 * 3 * 2 * 1 = 479001600
```

5) Encontrar o maior e o menor de um conjunto de valores informados pelo usuário. O valor 0 representa a saída e não deve ser considerado no restante do processamento.

a) Dos números informados que possuem até 3 dígitos verificar se o mesmo possui como unidade, dezena ou centena um determinado digito (entre 0 e 9) informado pelo usuário. Essa verificação deve ser realizada inclusive para o primeiro número informado e para os positivos ou negativos. Ao ler o dígito que o usuário informar para que seja verificado se o mesmo é unidade, dezena ou centena dos números informados, validar para que esse valor esteja entre 0 e 9.
b) Contar quantos valores pares são informados. Essa contagem inclui todos os números, inclusive o primeiro, exceto o zero que é a condição de saída.

6) Fazer um programa para:
Mostrar os divisores, calcular a quantidade deles e mostrar essa quantidade para os números compreendidos entre o valor 'x' informado pelo usuário e 'x+10', inclusive. Validar a entrada, o usuário deverá fornecer um número positivo entre 2 e 100.
Ao final mostrar a maior quantidade de divisores.
A figura ao lado exemplifica a execução, utilizá-la como modelo para o programa implementado.

```
Informe um nro inteiro:
10

10 -  1,  2,  5, 10, 4 divisores.
11 -  1, 11, 2 divisores.
12 -  1,  2,  3,  4,  6, 12, 6 divisores.
13 -  1, 13, 2 divisores.
14 -  1,  2,  7, 14, 4 divisores.
15 -  1,  3,  5, 15, 4 divisores.
16 -  1,  2,  4,  8, 16, 5 divisores.
17 -  1, 17, 2 divisores.
18 -  1,  2,  3,  6,  9, 18, 6 divisores.
19 -  1, 19, 2 divisores.
20 -  1,  2,  4,  5, 10, 20, 6 divisores.

Maior quantidade de divisores eh igual 6.
```

7) Fazer um programa que leia cinco números inteiros no intervalo entre 10 até 20, inclusive. Validar a entrada. Para cada número exibir a sequência dos pares de 2 até o número gerado e a soma desses pares.

```
11 ===> 2 4 6 8 10 <=== Soma = 30.
14 ===> 2 4 6 8 10 12 14 <=== Soma = 56.
13 ===> 2 4 6 8 10 12 <=== Soma = 42.
20 ===> 2 4 6 8 10 12 14 16 18 20 <=== Soma = 110.
13 ===> 2 4 6 8 10 12 <=== Soma = 42.
```