UNISINOS - UNIVERSIDADE DO VALE DO RIO DOS SINOS
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS (C6/6) – Curso: Informática

PROGRAMAÇÃO I – AULA 06

Disciplina: Linguagem de Programação PASCAL **E-mail:** ***osorio@exatas.unisinos.tche.br***
Professor responsável: ***Fernando Santos Osório*** **Web:**
Semestre: 99/1 ***http://www.inf.unisinos.tche.br/~osorio/prog1.html***
Horário: 61 **Xerox : *Pasta 54* (Xerox do C6/6)**

1. Comando REPEAT – Laço para repetição condicional de comandos

O comando REPEAT-UNTIL repete (laço = loop) a execução de um conjunto de comandos até que a expressão condicional do comando se torne verdadeira. Inicialmente o comando que segue o "REPEAT" é executado, seguindo-se do teste da expressão condicional contida no "UNTIL", onde se ela for ***falsa*** *voltamos* para uma nova execução do comando, e em caso contrário, se a expressão for ***verdadeira*** *termina* o laço de repetição. Ao terminar o comando REPEAT-UNTIL, a execução segue normalmente à partir da próxima linha que segue o "UNTIL".

A diferença entre o WHILE-DO e REPEAT-UNTIL é que no caso do While o comando pode não ser executado nenhuma vez, caso a expressão seja falsa na primeira vez que for testada, e já no caso do Repeat, o comando sempre é executado ao menos uma vez. No While o teste é no iniício do laço, e no Repeat o teste é no final do laço. O teste do While/Do faz com que o comando seja executado diversas vezes enquanto a condição for verdadeira, no teste do Repeat/Until o comando é executado enquanto a condição for falsa (terminando quando a expressão se tornar verdadeira).

* SINTAXE:

```
REPEAT  <comando> ;
UNTIL  <condição>  ;
```

ou

```
REPEAT <comando> ;
            ...
        <comando> ;
UNTIL   <condição> ;
```

<condição> = Expressão condicional cujo resultado é um valor lógico (True ou False).
A expressão condicional é descrita de maneira análoga à <condição> de um comando IF.
O laço continua a ser repetido somente se a condição não for satisfeita (resultado é false).

<comandos> = Comando que é executado repetidas vezes enquanto a expressão condicional do Repeat/Until não for verdadeira. O comando pode ser substituído por mais de um comando se usarmos um Begin/End ou mesmo se não fizermos o uso deste.

" Repita o <comando> Até que satisfaça a <condição> "

Observações:

- Todo <comando> pode ser substituído por um BEGIN/END contendo múltiplos comandos no seu interior (da mesma forma como ocorre com os outros comandos do Pascal, como por exemplo no IF/THEN/ELSE). O Repeat/Until, tendo ou não um Begin/End, sempre vai terminar por um ';' após os comandos e após a expressão condicional.

- Certifique-se de que a expressão condicional do Repeat/Until será afetada pelo(s) comando(s) contido(s) dentro do laço (comandos descritos após o "Repeat"). Caso a expressão não seja afetada, você irá criar um laço sem fim (loop infinito), pois jamais a expressão condicional mudará o seu estado.

Exemplos:

```
X : = 0 ;                       { Inicializa o contador       }
REPEAT X := X + 1 ;             { Faz uma contagem até 10 }
UNTIL  X = 10 ;
...

REPEAT ReadLn (Nota);
UNTIL  ( Nota >= 0.0 )  AND  ( Nota  <= 10.0 ) ;
...

REPEAT
   Writeln ('Hello World!');
   Write    ('Novamente (S/N) ? ');
   ReadLn (Resp);
UNTIL  ( Resp = 'N' )  OR ( Resp ='n' ) ;
...

REPEAT                          { Media => Usamos um valor especial para terminar  (-1)  }
    BEGIN                       { -1 serve como um sinalizador (flag) para terminar o laço }
        ReadLn (Media);
        IF (Media <> -1)
        THEN  IF  ( Media >= 6.0 )
                  THEN  WriteLn ('Aprovado')
                  ELSE   WriteLn ('Reprovado');
   END;
UNTIL Media = -1 ;
...
```

CONTRA-Exemplo: *EXEMPLO DE USO INCORRETO DO REPEAT-UNTIL*

```
IDADE := 60;
REPEAT Salario := Salario – Desconto;
UNTIL  Idade < 20 ;                    { Laço Infinito – Idade não é alterada no laço! }
...
```

EXERCÍCIOS – AULA 05

1. Faça um comando Repeat equivalente ao descrito abaixo (que realize o mesmo tipo de procedimento e obtenha o mesmo resultado), mas sem utilizar o operador relacional "And". Você pode usar outros operadores lógicos ou relacionais, exceto o "And".

```
REPEAT ReadLn (Nota);
UNTIL ( Nota >= 0.0 )  AND  ( Nota  <= 10.0 ) ;
```

2. Faça um programa que leia dois números inteiros, o primeiro é o valor inicial de um contador, e o segundo é o valor final do contador (testar se o valor inicial fornecido é inferior ao valor final). Usando o comando *Repeat*, escreva na tela uma contagem que comece no primeiro número lido, escreva os números seguintes colocando sempre apenas um número em cada nova linha da tela, e terminando a contagem quando chegar ao valor final indicado.

3. Ler o nome de um aluno e suas duas notas A e B, e após calcular a média ponderada entre estas notas (A tem peso 1 e B tem peso 2). Repetir este procedimento para uma turma composta por cinco alunos, usando o comando *Repeat*. Exemplo de tela de saída:

```
Entre com o nome do aluno: João da Silva
Entre com o grau A: 5.0
Entre com o grau B: 6.0
O aluno João da Silva tem uma média:5.66
```

4. Baseado no programa anterior, faça um novo programa de maneira que possamos trabalhar com turmas compostas por um número variável de alunos. Após calcular e imprimir a média de um aluno, exibir uma mensagem perguntando ao usuário se existem mais alunos (resposta: sim / não). Se tiver mais alunos, continuar o procedimento de leitura das notas e o cálculo da média até que o usuário responda 'não'. Usar o comando *Repeat* e gerar uma saída conforme o exemplo de tela de saída abaixo:

```
Entre com o nome do aluno: João da Silva
Entre com o grau A: 5.0
Entre com o grau B: 6.0
O aluno João da Silva tem uma média:5.66
Continuar (sim/não) ? sim
```

5. Baseado no programa anterior, faça um novo programa de maneira a validar as notas fornecidas pelo usuário (notas devem ser valores positivos entre 0.0 e 10.0). Indicar ao usuário se a nota fornecida é inválida e pedir para fornecer uma nova nota, repetindo este processo até que o usuário informe uma nota correta. Usar um laço *Repeat* na leitura das nota, e gerar uma saída conforme o exemplo de tela de saída abaixo:

```
Entre com o nome do aluno: João da Silva
Entre com o grau A: 15.3
ERRO: Nota inválida! Digite novamente a nota.
Entre com o grau A: 5.0
Entre com o grau B: 6.0
O aluno João da Silva tem uma média:5.66
Continuar (sim/não) ? não
```

6. Fazer um programa que calcule e imprima o fatorial de um número fornecido pelo usuário, usando o comando *Repeat*. Repetir a execução do programa tantas vezes quantas o usuário quiser. Lembre-se que o resultado do cálculo de um fatorial pode ser um número "grande" (Exemplo: Fatorial de 8 = 40320). Exemplo de tela de saída:

```
Entre com um número: 5
O fatorial de 5 é 120
Outro número (sim/não) ? não
```

7. Escrever um programa que calcule todos os números inteiros divisíveis por um certo valor indicado pelo usuário, e compreendidos em um intervalo também especificado pelo usuário. O usuário deve entrar com um primeiro valor correspondente ao divisor e após ele vai fornecer o valor inicial do intervalo, seguido do valor final deste intervalo. Usar o comando *Repeat*. Exemplo de tela de saída:

```
Entre com o valor do divisor: 3
Início do intervalo: 17
Final do intervalo: 29
Números divisíveis por 3 no intervalo de 17 à 29:
18  21  24  27
```

8. Apresentar na tela a tabuada de multiplicação dos números de 1 até 10. O programa deve exibir o resultado das multiplicações de 1x1, 1x2, ... até 1x10, pedir para o usuário teclar algo, e recomeçar com 2x1, 2x2, ... até 2x10, pedir novamente para teclar algo e seguir assim sucessivamente até chegar em 10x10.

9. Fazer um programa que leia o número total de bits usados para representar um valor, e exiba em seguida o maior valor inteiro e sem sinal que podemos representar com este número de bits. Lembre-se que um número binário com 8 bits pode representar valores entre 0 e 2^N-1 (2 na potência 8, menos 1 = 2*2*2*2*2*2*2*2 - 1 = 255).

10. Faça um programa que teste os reflexos do usuário de um computador. O programa deve inicialmente aguardar um certo tempo antes de começar uma contagem à partir de 1, sendo este tempo determinado de maneira aleatória. Ao terminar o tempo de espera inicial do programa, ele deve pedir para o usuário apertar uma tecla e começar imediatamente uma contagem à partir de 1, exibindo na tela os números a medida que for contando. Se o usuário pressionar uma tecla o programa deve terminar a contagem e exibir até onde ele conseguiu contar antes do usuário ter pressionado a tecla. Enquanto o usuário não apertar uma tecla o programa segue contando.

    Dicas: delay – Função do Pascal que permite esperar um certo tempo.
    random – Função do Pascal que permite escolher um número aleatório
    keypressed – Função do Pascal que permite saber se uma tecla foi pressionada ou não, sem no entanto ficar paralisado aguardando que uma tecla seja pressionada.