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

PROGRAMAÇÃO I – AULA 07

Disciplina: Linguagem de Programação PASCAL
Professor responsável: ***Fernando Santos Osório***
Semestre: 2001/2
Horário: 63

E-mail: ***osorio@exatas.unisinos.br***
Web:
http://inf.unisinos.br/~osorio/prog1.html
Xerox : *Pasta 54* (Xerox do C6/6)

1. Comando FOR/DO – Laço para repetição de comandos associados à um contador
FOR (Para) uma variável indo do valor inicial TO (Até) um valor final DO (Faça) comando

O comando FOR-TO/DOWNTO-DO repete (laço = loop) a execução de um conjunto de comandos tantas vezes quantas forem indicadas pelo contador. Inicialmente a variável contadora do FOR recebe um valor inicial, o comando é executado, sendo depois esta variável incrementada (no For-*TO*-Do) ou decrementada (no For-*DOWNTO*-Do). Este laço (executa comando + atualliza o contador) é repetido até que a variável alcance o valor final, somando ou subtraindo sempre uma unidade de cada vez (a cada nova iteração do laço) do valor desta variável. Ao terminar o comando FOR/DO, a execução segue normalmente à partir da próxima linha que segue o "DO".

O comando For/Do é muito similar a implementação de um laço While/Do ou Repeat/Until, quando este é usado para realizar uma contagem. É *importante* ressaltar que a variável utilizada no controle do laço For/Do (variável contadora) **TEM** que ser do *tipo ordinal* (do tipo inteiro), e além disso, é importante lembrar que o For/Do só pode usar um contador simples *que conta de uma em uma unidade*, somando (for-TO-do) ou subtraindo (for-DOWNTO-do) uma unidade da variável contadora.

* SINTAXE:

```
FOR <variável> := <valor_inicial>  TO <valor_final>
DO   <comando> ;
```

ou

```
FOR <variável> := <valor_inicial>  DOWNTO <valor_final>
DO   <comando> ;
```

<variável> = Variável do tipo ordinal que será usada como contador do comando For/Do.

<valor_inicial> = Valor ou expressão que indica o valor inicial a ser atribuído à variável contadora.

<valor_final> = Valor ou expressão que indica o valor final, ou seja, o valor que quando o contador ultrapassar o valor final o laço de repetição de comandos termina.

<comandos> = Comando que é executado repetidas vezes enquanto não ultrapassarmos o valor final estabelecido para o contador. O comando pode ser substituído por mais de um comando se usarmos um Begin/End, como nos demais comandos do Pascal.

" Para que a <variável>, varie de < início> até <fim>, faça o <comando> "

Observações:

- A variável do For/Do é incrementada ou decrementada *AUTOMATICAMENTE*, sendo que você não deve *DE MODO ALGUM* alterar o valor desta variável dentro do laço do For/Do. *Não atribua, ou altere o valor da variável contadora do For/Do!!*

- O valor inicial e/ou final da variável usada no For/Do pode ser definido por uma variável ou uma expressão numérica cujo resultado deve ser um número do tipo inteiro. Atenção: você não deve DE MODO ALGUM alterar o valor final do For/Do durante a sua execução (usualmente toda a alteração do valor final do For/Do será ignorada).

- A variável do For/Do deve ser do tipo inteiro (ordinal), sendo que o valor inicial deve ser menor ou igual ao valor final quando usarmos um For-**TO**-Do, ou, o valor inicial deve ser maior que o valor final quando usarmos um For-**DOWNTO**-Do.

- O ';' é colocado somente após o final do comando For, ou seja, após o comando associado ao "DO".

- Certifique-se de que a variável contadora vai realmente chegar até o valor final indicado no For/Do, pois caso no contrário teremos um laço infinito.

- 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).

Exemplos:

```
FOR  X : = 1  TO  10                { Inicializa o contador com 1 e conta até 10 }
DO WriteLn (X) ;                    { Exibe o valor do contador X na tela        }
...

FOR Aluno := 1 TO 5
DO   BEGIN
        ReadLn (Nota1,Nota2);
        WriteLn ('Média = ', (Nota1+Nota2)/2);
     END ;
...

WriteLn('Iniciando a contagem...');
FOR Contagem_Regressiva := 10 DOWNTO 1
DO   BEGIN
         WriteLn (Contagem_Regressiva);
         Delay(1000);                        { Espera 1000 milesegundos = 1 seg ... }
     END ;
WriteLn('Pronto!');
...
```

CONTRA-Exemplo: *EXEMPLO DE USO INCORRETO DO FOR-DO*

```
FOR Contador := 1 TO 10
DO Contador := Contador –1;          { Laço Infinito - Contador é alterado no laço! }
...
```

EXERCÍCIOS – AULA 07

1. Faça um programa de contagem regressiva. Cada vez que o usuário apertar uma tecla (readkey) você deve passar para o próximo número. Faça o programa usando o comando For-Do, e fazendo uma contagem de 10 até chegar em 0.

2. Altere o programa anterior, trocando o *readkey* pelo comando *delay* e faça uma verdadeira contagem regressiva de 1 em 1 segundo.

3. Faça um programa que leia dois números, o primeiro é o valor inicial de um contador, e o segundo é o valor final do contador (verifique se o valor inicial fornecido é inferior ao valor final). Usando o comando *For/Do*, escreva na tela uma contagem que comece no primeiro número lido, escreva os números seguintes colocando apenas um número em cada nova linha da tela, até chegar ao valor final indicado.

```
Entre com o Valor Inicial: 11
Entre com o Valor Final: 13
Contagem:
11
12
13
Final da contagem...
```

4. Faça um programa que escreva na tela os números pares entre 0 e 50, usando um comando For-Do. Não utilize nenhum IF/THEN neste programa, apenas o comando For-Do.

5. Re-escreva o comando For-Do que é dado abaixo, usando primeiramente um comando do tipo While-Do, e depois re-escreva novamente o mesmo comando usando um comando Repeat-Until. Os dois programas que forem usar o While e o Repeat devem se comportar de maneira idêntica ao programa que usava o For-Do.

```
Program Contagem;
Var
   Cont : integer;
Begin
    For Cont:= 10 to 20
    Do  WriteLn ('Contador = ', (Cont /10-1.0):3:2);
    ReadLn;
End.
```

6. Ler o nome de um aluno e suas duas notas A e B, e após calcular a média ponderada entre estas notas (A tem 30% do peso do grau final e B tem 70% do peso). Repetir este procedimento para uma turma composta por cinco alunos, usando o comando *For/Do*. 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:5.7
```

7. Fazer um programa que calcule e imprima o fatorial de um número fornecido pelo usuário, usando o comando *For – **TO** - Do*. Perguntar ao usuário se ele deseja calcular o fatorial de outro número e repetir a execução do programa tantas vezes quantas o usuário indicar. Exemplo de tela de saída:

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

8. Alterar o programa anterior do fatorial, usando o comando do tipo "For–**DOWNTO**–Do" no lugar dos comandos "For-**TO**-Do". Exemplo de tela de saída:

```
Entre com um número: 4
O fatorial de 4 é 24
Outro número (sim/não) ? sim
Entre com um número: ...
```

9. Ler um número e gerar todos os números primos entre 1 e o número fornecido, escrevendo na tela o resultado. Usar o comando *For/Do*. Procure fazer uma nova versão otimizada deste programa, de modo que o cálculo seja realizado de forma mais rápida (evite fazer testes/cálculos desnecessários) – usar qualquer tipo de comando.

```
Entre com o valor final: 10
1 é primo
2 é primo
3 é primo
5 é primo
7 é primo
Fim!
```

10. Escrever um programa que escreva 'Hello' na tela, deslocando-o coluna à coluna, e linha à linha, por toda a tela do microcomputador (considerar uma tela de 24 linhas com 80 colunas). Para deslocar a palavra na tela, você deve escrever, após deve apagar, e escrever novamente com um certo deslocamento em relação a sua última posição. Use o comando *For* juntamente com o comando gotoxy. Exemplo de tela de saída:

```
Hello   =>    Hello   =>      Hello
```

Atenção para não passar os limites da tela, considerando o tamanho da palavra. Testar o efeito do deslocamento usando um comando *clrscr* após cada deslocamento/escrita na tela, e após alterar o programa, fazendo uso do comando *delay* seguido de um *write* com espaços em branco para apagar somente o que havia sido escrito anteriormente...

11. Fazer um programa que leia uma string e converta todos os caracteres desta string para maiúsculo. Considerar também os caracteres especiais (caracteres acentuados) nesta conversão. Depois de convertida a string, exibir o resultado na tela. Dica: você pode concatenar caracteres no final de uma string usando o comando '+'... e para quebrar a string e obter o n-ésimo caracter de uma string você pode usar os colchetes texto[elemento].

```
Exemplos:  txt_string := txt_string + 'X' ;   { Adiciona 'X' ao final da  txt_string    }
           caracter := txt_string[posicao] ;  { Extrai o caracter  localizado na posicao
                                                indicada dentro string }
```

12. Fazer um programa de "criptografia" (codificação de dados visando a privacidade de acesso as informações), onde dada uma string este programa codifique os dados através de um processo de substituição de letras. Você pode definir o seu próprio método de criptografia, desde que depois seja possível reverter este processo, ou seja, um código criptografado deve poder ser convertido novamente ao valor inicial). Fazer um outro programa complementar a este que deve ser capaz de descriptografar a string, ou seja, deve pegar uma string codificada e retornar ao texto original.

CRIPTOGRAFAR Entre como texto a ser criptografado: *Pascal* Texto criptografado: **Qbtdbm**

DESCRIPTOGRAFAR Entre como texto a ser descriptografado: Qbtdbm Texto descriptografado: **Pascal**