UTN Universidad Tecnológica Nacional
Facultad Regional La Plata

Ingeniería en Sistemas de Información

Algoritmos y Estructuras de Datos

Proyecto

Pascal // Fascículo 3

Contenido:

Jefe de Cátedra:
- Lic. Carlos A. López

Jefe de Trabajos Prácticos:
- Ing. Gustavo Cerveri

Tutores de Proyecto:
- Marcelo Cardós - marcelo@ayed.com.ar
- Eduardo M. Puricelli - edus@ayed.com.ar
- Sergio R. Vaquero - sergio@ayed.com.ar

Bibliografía:
Programación en Turbo/Borland Pascal 7
3ra edición - Osborne / McGraw-Hill
Luis Joyanes Aguilar

http://www.frlp.utn.edu.ar/materias/algoritmos/

http://www.ayed.com.ar/

1. Estructuras de decisión: simples (SI..ENTONCES..SINO) y múltiples (CASO)

Bueno, aquí estamos de nuevo con un nuevo fascículo. Trataremos de ver las estructuras de decisión, que en el caso de Turbo Pascal son bastante intuitivas, ya que se trata casi de traducir el pseudocódigo...

Ante todo, vamos a definir lo que es un BLOQUE DE CÓDIGO en Pascal, como un conjunto de sentencias que se encuentran relacionadas por algún motivo, las cuales se presentan entre un "**begin**" y un "**end;**". En el caso de que el bloque contenga una sola sentencia, generalmente no se la "encierra" entre "begin" y "end;", pero sí se hace cuando hay más de una instrucción. Hasta ahora lo hemos usado pero no habíamos hecho esta salvedad. Será importante para explicar lo siguiente:

1.1. IF..THEN, IF..THEN..ELSE.

Veamos el siguiente ejemplo, que resume el uso del SI en Pascal. Podemos ver lo explicado recién acerca del bloque de código: el bloque del **IF** (si) no posee BEGIN-END ya que su contenido es una instrucción simple, pero el bloque del **ELSE** (sino) sí lo necesita.

```
PROGRAM condicional;

VAR
  respuesta: string;

BEGIN
  writeln('Es usted alumno de la UTN?');
  read(respuesta);

  IF (respuesta='no') THEN
    writeln('Muchas gracias')
  ELSE
  begin
    writeln(' Es usted alumno de la UTN');
    writeln('Se requerirán sus notas');
  end;
END.
```

```
PROGRAMA condicional

VARIABLES
  respuesta: caracter

HACER
  imprimir:'Es usted alumno de la UTN?'
  leer(respuesta)

  si (respuesta='no') entonces
    imprimir:'Muchas gracias'
  sino
    imprimir:'Es usted alumno de la UTN'
    imprimir:'Se requerirán sus notas'
  fin si
FIN HACER
FIN PROGRAMA
```

Tal como en pseudocódigo, el bloque del "else" podría omitirse según el caso.

Un detalle muy importante a tener en cuenta, es la ausencia de punto y coma antes del "**ELSE**". En caso de haber un punto y coma en ese lugar, el compilador interpreta que el "**IF**" ha terminado, y en este caso eso no sería cierto.

1.2. Anidamiento.

Al igual que en pseudocódigo, puedo tener más de un IF anidado, o sea un IF dentro de otro. Para ello, un consejo importante es el de prestar gran atención a los lugares donde se utiliza el punto y coma.

Olvidarse de poner puntos y comas es el error más común al usar Turbo Pascal. Recuerden que cuando compilan el programa y les da: **"ERROR 85: ";" expected."**, Pascal les deja posicionado el cursor en la línea siguiente de donde **les falta un ";".**

1.3. CASE (variable) OF…ELSE.

Entre las diferencias más importantes con respecto al pseudocódigo, es que podemos especificar rangos o listas de valores para los cuales se debe proceder tal como se muestra en el ejemplo.

Una de las características importantes, es la aparición de un “end;” al final del “case” sin la existencia de un “begin”.

```
PROGRAM caso;

VAR
  nota: integer;

BEGIN
  writeln(‘Ingrese su nota de AyED’);
  readln(nota);

  CASE nota OF
    0:
    begin
      writeln(‘Usted ingresó 0);
      write(‘Estuvo ausente’);
    end;

    1,2,3: write(‘Estudie más’);

    4..9: write(‘Muy bien!’);

    10: write(‘Excelente!’);
  ELSE
    write(‘Nota equivocada’);
  END;
END.
```

```
PROGRAMA caso

VARIABLES
  nota: entero

HACER
  imprimir:‘Ingrese su nota de AyED’
  leer(nota)
  caso
    nota = 0:
      imprimir:‘Usted ingresó 0’
      imprimir:‘Estuvo ausente’
    nota = 1: imprimir:‘Estudie más’
    nota = 2: imprimir:‘Estudie más’
    nota = 3: imprimir:‘Estudie más’
    nota = 4: imprimir:‘Muy bien!’
    nota = 5: imprimir:‘Muy bien!’
    nota = 6: imprimir:‘Muy bien!’
    nota = 7: imprimir:‘Muy bien!’
    nota = 8: imprimir:‘Muy bien!’
    nota = 9: imprimir:‘Muy bien!’
    nota = 10: imprimir:‘Excelente!’
  en otro caso
    imprimir:‘Nota equivocada’
  fin caso
FIN HACER
FIN PROGRAMA
```

2. Estructuras de control. Diferencias.

Existen varias estructuras repetitivas en Pascal, pero sólo citaremos las vistas en pseudocódigo:

2.1. FOR (Repetir Para)

```
PROGRAM repetir;

VAR
  i,nota,cantalu: integer;

BEGIN
  writeln(‘Ingrese cantidad de alumnos de
    la comisión’);
  readln(cantalu);

  FOR i:= 1 TO cantalu DO
  begin
    writeln(‘Ingrese nota’);
    readln(nota);

    if nota>=4 then
      writeln(‘¡¡¡Aprobaste!!!’)
    else
      writeln(‘¡No podes desaprobar esta
        materia! ¡Sos una bestia!’)

  end;

END.
```

```
PROGRAMA repetir

VARIABLES
  i,nota,cantalu: entero

HACER
  imprimir:‘Ingrese cantidad de alumnos de
    la comisión’
  leer(cantalu)

  REPETIR PARA i:= 1,cantalu,1

    imprimir(‘Ingrese nota’)
    leer(nota);

    si (nota>=4) entonces
      imprimir:‘¡¡¡Aprobaste!!!’)
    sino
      imprimir:‘¡No podes desaprobar esta
        materia! ¡Sos una bestia!’)
    fin si

  FIN REPETIR PARA
FIN HACER
FIN PROGRAMA
```

2.1. WHILE (Repetir Mientras)

Otro ejemplo, ahora sin saber la cantidad de notas a cargar, de manera que preguntaremos todas las veces si quedan más notas. Para ello, utilizaremos una variable que se leerá y será la condición para la repetición.

```
PROGRAM repetir;

VAR
  resp: string;
  nota: integer;

BEGIN
  resp:= 'si';
  WHILE (resp = 'si') DO
  begin
    writeln('Ingrese nota');
    readln(nota);

    if (nota >= 4) then
      writeln('¡¡¡Aprobaste!!!')
    else
      writeln('No aprobaste, a estudiar!')

    writeln('Hay más notas? (si/no)');
    readln(resp);
  end;
END.
```

```
PROGRAMA repetir

VARIABLES
  resp: caracter
  nota: entero

HACER
  resp:= 'si'
  REPETIR MIENTRAS (resp = 'si')

    imprimir 'Ingrese nota'
    leer(nota)

    si (nota >= 4) entonces
      imprimir:'¡¡¡Aprobaste!!!'
    sino
      imprimir:'No aprobaste, a estudiar!'
    fin si

    imprimir:'Hay más notas? (si/no)'
    leer(resp)
  FIN REPETIR MIENTRAS
FIN HACER
FIN PROGRAMA
```

3. Consideraciones para trabajar en Turbo Pascal 7.0 de Borland.

Bueno, ya tenemos una base para empezar a realizar nuestros propios programas en Pascal, pero sin embargo seguramente se toparán con lo siguiente:

3.1. Compilación

Bueno, supongamos que ya escribimos alguno de los ejemplos de programas dados… ¿Como comprobamos que escribimos bien el programa (sin errores de sintaxis)? ¿Y como ejecutamos el programa? Explicaremos como compilar y correr (ejecutar) un programa en Pascal.

Inicialmente, si utilizamos la barra de menús, en el menú “Compile”, con la opción “Compile” o “Make” podemos **compilar** nuestro código y darnos cuenta **si hemos escrito sin errores** para la interpretación de Pascal, lo que no quiere decir que el programa esté funcionando correctamente, pero es un gran avance el saber que al menos se puede correr.

Luego se puede ejecutar nuestro programa, utilizando el menú “Run” y la opción de su mismo nombre. Ahora sí, interactuando con el programa en ejecución, podremos probarlo y saber si realmente está funcionando correctamente. En fin, hacer un “Run” significa compilarlo antes, si es que esto no fue realizado manualmente, por lo que si se desea probar directamente el código escrito, podemos utilizar sólo esta opción.

De manera resumida y cómoda, **para chequear nuestro código podemos compilarlo presionando** la tecla **F9 y correrlo presionando Ctrl+F9.**

3.2. Help

Muchos editores de lenguajes de programación, poseen en su sección de Ayuda, una amplia guía para el programador. No es la excepción esta versión de Pascal, que pese a estar en inglés no creemos que sea dificultoso entenderlo. En fin, siempre será bueno tener a mano la combinación de teclas **Shift+F1 para acceder** directamente **al "Turbo Help Index" y Alt+F1 para regresar** a la última pantalla consultada.

3.3. Parche

Quizás más de uno ha sentido hablar del "parche" para Pascal. En el próximo fascículo incluiremos la utilización de procedimientos y funciones; en especial usando una "librería" (conjunto de procedimientos y funciones) llamada CRT, que nos permite utilizar por ejemplo, colores en los textos que visualizamos en pantalla y otras.

Resulta que al utilizar esta librería en las PCs mayores a 200 MHz de procesador, aparece el siguiente mensaje al ejecutar nuestro programa: **"Error 200: Division by zero.".**

Para resolver este problema, **es necesario instalar un parche**, que pueden descargar del sitio WEB. Debemos notar que hay varios parches en Internet y que por comentarios de alumnos anteriores, a algunos les funcionó uno y a otros otro de los que encontramos en la red.