Estructuras de control: for

InCo - FING

Section 1

Repetición

Instrucciones de repetición

Permiten que una sentencia sea ejecutada varias veces.

Pascal posee 3 instrucciones de repetición:

for	controlada por una variable que toma valores en un rango especificado.
while	controlada por una expresión booleana. La repetición continúa mientras la expresión se mantenga verdadera.
repeat	controlada por una expresión booleana. La repetición termina cuando la expresión se hace verdadera.

La instrucción `for`. Ejemplo.

Primer caso: Repetir una instrucción un número constante de veces:

```
const ASTERISCO = '*';
      VECES     = 10;
var   i : integer;
begin
   for i:= 1 to VECES do
      write(ASTERISCO)
end.
```

La instrucción `for` anterior es equivalente a:

```
write(ASTERISCO);write(ASTERISCO);write(ASTERISCO);
write(ASTERISCO);write(ASTERISCO);write(ASTERISCO);
write(ASTERISCO);write(ASTERISCO);write(ASTERISCO);
write(ASTERISCO)
```

Ejemplo (2)

Segundo caso: La instrucción a repetir utiliza la variable de control.

Desplegar los 10 primeros pares:

```
for i:= 1 to 10 do
   writeln(2*i)
```

La instrucción anterior es equivalente a:

```
writeln(2*1);writeln(2*2);writeln(2*3);
writeln(2*4);writeln(2*5);writeln(2*6);
writeln(2*7);writeln(2*8);writeln(2*9);
writeln(2*10)
```

Ejemplo (3)

Tercer caso: La cantidad de repeticiones es variable. Repetición anidada.

Dibujar un cuadrado de lado *a* , donde *a* es un número entero positivo que ingresa el usuario.

```
    a=10
**********
**********
**********
**********
**********
**********
**********
**********
**********
**********
**********
```

Ejemplo (3) (cont.)

```
readln(lado);

for i:= 1 to lado do
begin
   (* dibujar la i-esima fila *)
   for j:= 1 to lado do
      write('*');
   writeln  (* cambiar de linea *)
end;
```

Ejemplo (4)

Encontrar todos los números múltiplos de 3 que están comprendidos entre dos enteros dados.

```
readln(num_A,num_B);
for numero:= num_A to num_B do
   if numero mod 3 = 0 then
      writeln(numero)
```

Ejemplo (5)

Igual al anterior pero calculando la cantidad de múltiplos de 3.

```
var
   num_A,num_B,numero,contador: integer;
begin
   readln(num_A,num_B);
   contador:= 0; (* inicialización *)
   for numero:= num_A to num_B do
      if numero mod 3 = 0 then
         contador:= contador + 1; (* incremento *)
   (* mostrar el total *)
   writeln('La cantidad de divisores de 3 es: ',
           contador)
end.
```

Ejemplo (6)

Calcular la multiplicación de dos enteros usando sumas:

```
var i,a,b,producto : integer;
    negativo       : boolean;
begin
   readln(a,b);
   negativo:= b < 0;
   b:= abs(b);

   producto:= 0; (* inicialización *)
   for i:= 1 to b do
      producto:= producto + a; (* acumulación *)

   (* correccion del signo *)
   if negativo then
      producto:= - producto;
   (* mostrar el resultado *)
   writeln('El producto es: ', producto)
end.
```

Sintaxis de la instrucción `for`

BNF

```
sentencia_for ::=
   'for' identificador := expresión1 ('to'|'downto') expresión2
   'do' instrucción
```

- `to`, `downto`, `do` son palabras reservadas de Pascal.
- *identificador* debe ser una variable declarada de algún tipo **escalar** (integer, char, boolean, subrango, enumerado)
- *expresión1* y *expresión2* son expresiones del mismo tipo que el identificador.

Semántica de la instrucción `for`

La instrucción:

```
for v := e1 to e2 do instrucción
```

Se ejecuta de la siguiente forma:

- Se evalúan las expresiones *e1* y *e2*. Sean sus valores *v1* y *v2*.
- Se ejecuta la siguiente secuencia:

```
v:= v1; instrucción;
v:= succ(v); instrucción;
...;
v:= v2; instrucción
```

Observación: Si v1 $>$ v2, la instrucción no ejecuta nada.

Semántica de la instrucción `for` (cont.)

La instrucción:

```
for v := e1 downto e2 do instrucción
```

Se ejecuta de la siguiente forma:

- Se evalúan las expresiones *e1* y *e2*. Sean sus valores *v1* y *v2*.
- Se ejecuta la siguiente secuencia:

```
v:= v1; instrucción;
v:= pred(v); instrucción;
...;
v:= v2; instrucción
```

Observación: Si v1 $<$ v2, la instrucción no ejecuta nada.

Observaciones

- Se considera un **error**:
  que la instrucción del `for` ***modifique*** *la variable de control.*
- El valor de la variable de control luego de terminar el `for` es **indefinido**.
- Una instrucción `for` **siempre** termina.
- La variable de control **no** puede ser `real`.
- Se recomienda que la instrucción del `for` **no modifique** las expresiones e1 y e2.