RESOLUCIÓN – PARCIAL 2 – 1º CUATRIMESTRE 2008

EJERCICIO

Codifique el programa Pascal que realice lo siguiente:

- Cargar una matriz AZULEJO de caracteres de M filas y N columnas desde teclado.
- Crear una matriz PARED de X filas e Y columnas que contenga el contenido de la matriz AZULEJO repetido tantas veces como entre. En el caso de que el espacio que sobre en la matriz PARED no alcance para colocar el contenido de la matriz AZULEJO completo, colocar la porción de AZULEJO que entre.
- Imprimir la matriz PARED de a una fila por renglón.
- Defina todas las constantes, tipos y variables necesarias. Es imprescindible modularizar correctamente para aprobar.

AZULEJO

#			1
	#		.
		#	M
1	..	N	

PARED

1	#			#			#			#
.		#			#			#		
.			#			#			#	
.	#			#			#			#
.		#			#			#		
.			#			#			#	
X	#			#			#			#
	1	.	.	.	.	.	.	.	.	y

ESTRATEGIAS

Los módulos de carga de matriz AZULEJO y de impresión de matriz PARED son sencillos y no requieren de estrategia de resolución alguna. Sin embargo, la inclusión de AZULEJO en PARED sí conviene planificarla dado que hay unas cuantas posibilidades de efectuarse.

La tarea consiste en recorrer la matriz AZULEJO e ir copiando elemento por elemento en la matriz PARED. Esto puede modularizarse o no, pero veremos ambos casos.

Si no modularizamos, por cada vez que copiamos un AZULEJO en la PARED, debemos conservar las posiciones finales para saber dónde pegar un nuevo AZULEJO y también, verificar que no nos excedamos de los límites de la PARED. Luego debemos volver a la posición inicial del AZULEJO para recomenzar la copia en PARED.

Hay muchas variantes posibles, pero hay que tener en claro la independencia de los índices que recorren a AZULEJO de los que recorren PARED, puesto que no son los mismos, debido a que ambas matrices poseen diferentes dimensiones.

Así las cosas, la dificultad se plantea cuando un AZULEJO no tiene espacio suficiente para copiarse por completo en la PARED, entonces debemos ser cuidadosos en detener la copia ni bien alcancemos los límites X e Y de la PARED, para avanzar a la siguiente fila o columna, según estemos haciendo la copia.

Veamos la implementación de algunas variantes de resolución.

```
program Parcial2_2008;
      const
       m=2; n=2; x=3; y=3;

      type
        az=array[1..m,1..n] of char; pd=array[1..x,1..y] of char;

      //--------------MODULO 1-----------------------------
      PROCEDURE ARMA_AZULEJO(var Azulejo:az);
      var
        i,j: integer;
      begin
        for i:=1 to m do
            for j:=1 to n do
                begin
                   writeln('Ingrese un motivo');
                   readln(Azulejo[i,j]);
                end;
      end;

      //---------------MODULO 2-----------------------------
      PROCEDURE ARMA_PARED(Azulejo:az; var Pared:pd);
      var
        fa, ca, fp, cp: integer; //fa: fila azulejo,    fp: fila pared
      begin                      //ca: columna azulejo, cp: columna pared
          fp:=1;
          while (fp<=x) do
          begin
              fa:=1;
              while ((fa<=m) and (fp<=x)) do
              begin
                cp:=1;
                while (cp<=y) do
                begin
                   ca:=1;
                   while ((ca<=n) and (cp<=y))do
                   begin
                        Pared[fp,cp]:=Azulejo[fa,ca];
                        ca:=ca+1;
                        cp:=cp+1;
                   end;
                end;
                fa:=fa+1;
              end;
       end;
      end;

      //-----------------MODULO 3-----------------------------
      PROCEDURE MOSTRAR_PARED(Pared:pd);
      var
         fp, cp: integer;
      begin
           for fp:=1 to x do
           begin
              for cp:=1 to y do
                  write(Pared[fp,cp]);
              writeln;
           end;
      end;
```

```
//-------------PROGRAMA PRINCIPAL-----------------
var
  mat_azulejo: az;
  mat_pared: pd;

begin
  ARMA_AZULEJO(mat_azulejo);
  ARMA_PARED(mat_azulejo, mat_pared);
  MOSTRAR_PARED(mat_pared);
end.

//--------------- FIN -----------------------------
```

VARIANTE 2: (sólo se muestra el módulo de copia, el resto es igual)

```
//---------------MODULO 2-----------------------------

PROCEDURE ARMA_PARED(Azulejo:az; var Pared:pd);
var
  fa, ca, fp, cp: integer;
begin
    fa:=1;
    for fp:=1 to x do
    begin
         for cp:=1 to y do
         begin
              if (fa<=m) and (ca<=n) then
              begin
                   PARED[fp,cp]:=AZULEJO[fa,ca];
                   ca:=ca+1;
              end
              else
                  if (fa>m) then
                  begin
                       ca:=1;
                       fa:=1;
                       PARED[fp,cp]:=AZULEJO[fa,ca];
                       ca:=ca+1;
                   end
                   else
                       if (ca>n) then
                       begin
                            ca:=1;
                            PARED[fp,cp]:=AZULEJO[fa,ca];
                            ca:=ca+1;
                        end;
              end;
         end;
    fa:=fa+1;
    ca:=1;
    end;
end;
```

VARIANTE 3: (sólo se muestra el módulo de copia, el resto es igual)

```
//---------------MODULO 2-----------------------------

      PROCEDURE ARMA_PARED(Azulejo:az; var Pared:pd);
      var
        fa, ca, fp, cp: integer;
      begin
          fa:=1;
          for fp:=1 to x do
          begin
             ca:=1;
             if (fa<=m) then
             begin
               for cp:=1 to y do
               begin
                    if (ca<=n) then
                    begin
                         Pared[fp,cp]:=Azulejo[fp,ca];
                         ca:=ca+1;
                    end
                    else
                    begin
                         ca:=1;
                         Pared[fp,cp]:=Azulejo[fp, ca];
                         ca:=ca+1;
                    end;
               end;
               fa:=fa+1;
             end
             else
                 fa:=1;
          end;
```