

# Programación recursiva en Prolog: Quadrees

Curso 2002/03, Fecha:20/12/2002

**Enunciado 1 (quadrees)** *Un quadtree puede representarse de forma recursiva como:*

- *Un cuadrado vacío mediante la constante vacio*
- *Un rectángulo de un color fijo, por ejemplo: `rect(rgb(1,0,0))`*
- *Una división en cuatro cuadrantes, por ejemplo: `d(vacio, rect(rojo), rect(verde), vacio)`*

*El siguiente predicado `wqt(QT)` almacena el quadtree `QT` es un fichero VRML*

```
wqt(QT):-wqt('qt.wrl',QT).
```

```
wqt(F,OT):-open(F,write,S),
             cabecera(S),
             viewPoint(S),
             wqt(S,0,0,64,OT),
             close(S).
```

```
wqt(S,-,-,-,vacio):-write(S,'').
```

```
wqt(S,X,Y,D,rect(C)):-
```

```
    putBox(S,X,Y,0,C,D,D,1).
```

```
wqt(S,X,Y,D,d(C1,C2,C3,C4)):-
```

```
    (D=<1->write(S,''))
```

```
    ; D2 is D/2, D4 is D/4,
```

```
      XP is X+D4, XN is X-D4,
```

```
      YP is Y+D4, YN is Y-D4,
```

```
      wqt(S,XN,YP,D2,C1),
```

```
      wqt(S,XP,YP,D2,C2),
```

```
      wqt(S,XN,YN,D2,C3),
```

```
      wqt(S,XP,YN,D2,C4)
```

```
    ).
```

```
cabecera(S):-
```

```
    write(S,'#VRML V2.0 utf8\n\n').
```

```
viewPoint(S):-
```

```
    write(S,'Viewpoint { description "Punto" \n'},
```

```

write(S, '                orientation 0 0 1 0\n'),
write(S, '                position 0 0.2 150 } \n').

putSphere(S,X,Y,Z,C,R):-
  translate(S,X,Y,Z),
  color(S,C), sphere(S,R), endColor(S),
  endTranslate(S).

putBox(S,X,Y,Z,C,SX,SY,SZ):-
  translate(S,X,Y,Z),
  color(S,C),
  box(S,SX,SY,SZ),
  endColor(S),
  endTranslate(S).

translate(S,X,Y,Z):-
  write(S, ' Transform { translation  }'), write3(S,X,Y,Z), nl(S),
  write(S, ' children [ ').

endTranslate(S):-
  write(S, ' ] }\n').

color(S,rgb(R,G,B)):-
  write(S, ' Shape { appearance Appearance  }'),
  write(S, ' { material Material { diffuseColor  }'),
  write3(S,R,G,B),
  write(S, ' } } ').

endColor(S):-
  write(S, ' } ').

box(S,X,Y,Z):-
  write(S, ' geometry Box { size  }'),
  write3(S,X,Y,Z),
  write(S, ' } ').

sphere(S,R):-
  write(S, ' geometry Sphere { radius  }'),
  write(S,R),
  write(S, ' } ').

write3(S,X,Y,Z):-
  format(S, '~2f ~2f ~2f ', [X,Y,Z]).

  Un ejemplo de llamada sería ?-wqt(d(vacio,rect(rgb(1,0,0)), rect(rgb(0,1,0)), vacio)).
  Construir diversos quadrees

```

Nota: Si se utiliza el SWI-Prolog editor es necesario deshabilitar la opción Integrated Swi-Prolog window en el menú Configuration para que el programa anterior pueda realmente escribir en el fichero

**Enunciado 2 (escalera)** Construir un predicado *escalera*( $X, Y$ ) que se cumple si  $Y$  es un cuadtrees formado al colocar el cuadtrees  $X$  en la esquina superior izquierda y en la esquina inferior derecha, dejando los otros dos cuadrantes vacíos.

**Enunciado 3 (escalN)** Construir un predicado *escalN*( $N, X, Y$ ) que se cumple si  $Y$  es un cuadtrees formado al repetir  $2^N$  veces el cuadtrees  $X$  en diagonal.

**Enunciado 4 (repiteN)** Construir un predicado *repite*( $N, X, Y$ ) que se cumple si  $Y$  es un cuadtrees formado al repetir el  $N$  veces el cuadtrees  $X$  en cada cuadrante.

**Enunciado 5 (gira)** Construir un predicado *gira*( $X, Y$ ) que se cumple si  $Y$  es un cuadtrees formado al girar  $X$  90 grados.

**Enunciado 6 (diags)** Construir un predicado *diags*( $N, X, Y$ ) que se cumple si  $Y$  es un cuadtrees formado por repetir el cuadtrees  $X$  en las diagonales girando  $90^\circ$  cada cuadrante.

**Enunciado 7 (inserta)** Construir un predicado *inserta*( $X, Q, R$ ) que se cumple si  $R$  es el resultado de insertar el cuadtrees  $X$  en un cuadro vacío del cuadtrees  $Q$

**Enunciado 8 (insertaLs)** Construir un predicado *insertaLs*( $Ls, Q, R$ ) que se cumple si  $R$  es el resultado de insertar la lista de cuadtrees  $Ls$  en cuadros vacíos del cuadtrees  $Q$

**Enunciado 9 (galeria)** Construir una galería de cuadtrees

**Enunciado 10 (octrees(Opcional))** Tomando como referencia el predicado *wqt*, construir un predicado *wot* que permita representar octrees. Construir una galería con octrees generados en Prolog