```
-- AUTORA: Elvira Mayordomo Cámara
-- PROYECTO: módulo de declaración del TAD listas con punto de interés a partir
-- de pilas limitadas
-- FICHERO: listaspi.ads
-- FECHA: 16-10-03

-- especificación listaspi2
--    parámetro formal
--          género elemento
--          constante n:nat
--    fpf
--    usa booleanos, pilas2(elemento,n)
--          género lpi
--    operaciones
--          <_,_>: pila pila → lpi
--          listaVacía: → lpi
--          parcial insertar: lpi elemento → lpi
--          borrar: lpi → lpi
--                parcial actual: lpi → elemento
--          esvacía: lpi → bool
--          parcial principio: lpi → lpi
```

```
--          parcial avanzar: lpi → lpi
--          alfinal: lpi → bool
--      dominios de definición p1,p2:pila; e:elemento
--          tamaño(p1)<n → insertar(<p1,p2>,e)
--          actual(<p1,apilar(p2,e)>)
--          tamaño(p1)+tamaño(p2)<=n → principio(<p1,p2>)
--          tamaño(p1)<n → avanzar(<p1,apilar(p2,e)>)
--          avanzar(<p1,pilaVacía>)
--      ecuaciones p1,p2:pila; e:elemento
--          listaVacía = <pilaVacía, pilaVacía>
--          insertar(<p1,p2>,e) = <apilar(p1,e),p2>
--          borrar(<p1,pilaVacía>) = <p1,pilaVacía>
--          borrar(<p1,apilar(p2,e)>) = <p1,p2>
--          actual(<p1,apilar(p2,e)>) = e
--          esvacía(<pilaVacía, pilaVacía>) = verdad
--          esvacía(<apilar(p1,e),p2>) = falso
--          esvacía(<p1,apilar(p2,e)>) = falso
--          principio(<apilar(p1,e),p2>) = principio(<p1,apilar(p2,e)>)
--          principio(<pilaVacía,p1>) = <pilaVacía,p1>
--          avanzar (<p1,pilaVacía>) = <p1,pilaVacía>
--          avanzar(<p1,apilar(p2,e)>) = <apilar(p1,e),p2>
```

```
--          alfinal(<p1,pilaVacía>) = verdad
--          alfinal(<p1,apilar(p2,e)> = falso
-- fespec

with pilas4;
generic
      type elemento is private;
      capacidad:integer;
package listaspi is
      package mispilas is new pilas4(elemento,capacidad);
      use mispilas;
      type lpi is limited private;
      -- coste en memoria: una lista de cualquier tamaño
      -- ocupa O(capacidad)
      procedure crear(p1,p2:in pila;l:out lpi);
      -- Post: l=<p1,p2>
      -- coste en tiempo: para una lista de n datos tarda O(n)
      -- Esta operación es auxiliar y no es necesario implementarla
      procedure listaVacia(l:out lpi);
      -- Post: l=listaVacía
      -- coste en tiempo O(1)
```

procedure insertar(l:**in out** lpi; e:**in** elemento);
-- Pre: l=<p1,p2> $\wedge$ ($\neg$*esllena*(p1))
-- Post: l=*insertar*(<p1,p2>,e)
-- coste en tiempo O(1)
procedure borrar(l:**in out** lpi);
-- Pre: l=l0
-- Post: l=*borrar*(l0)
-- coste en tiempo O(1)
function actual(l:lpi) **return** elemento;
-- Pre: $\neg$*alfinal*(l)
-- Post: actual(l)=*actual*(l)
-- coste en tiempo O(1)
function esVacia(l:lpi) **return** boolean;
-- Post: esVacia(l)= *esvacía*(l)
-- coste en tiempo O(1)
procedure principio(l:**in out** lpi);
-- Pre: l=<p1,p2> $\wedge$ (*tamaño*(p1)+*tamaño*(p2)<=capacidad)
-- Post: l=*principio*(<p1,p2>)
-- coste en tiempo: si la lista tenía m datos antes del punto de interés
-- tarda O(m)
procedure avanzar(l:**in out** lpi);

```
        -- Pre: l=<p1,p2> ∧ (esvacía(p2) ∨ (¬esllena(p1))
        -- Post: l=avanzar(l0)
        -- coste en tiempo O(1)
        function esFinal(l:lpi) return boolean;
        -- Post: esFinal(l)= alfinal(l)
        -- coste en tiempo O(1)
private
        type lpi is
                record
                        piladcha:pila;
                        pilaizda:pila;
                end record;
end listaspi;
```