Ejercicio 1a

```
procedure imprimirSeguidas(a : Secuencia);
var i : integer;
begin
 for i := 1 to MAX_FICHAS - 1 do
          if (a[i] = a[i + 1]) then
                    ImprimirPar(i, i + 1)
end
```

Ejercicio 1b

```
function indiceBloqueN(a : Secuencia; n : Integer) : Integer;

  function largoBloque(a : Secuencia; posIni : Integer) : integer;
  (* Halla el largo de un bloque suponiendo que el bloque comienza en posIni *)
  var     i, largo : Integer;
          ficha : TipoFicha;
  begin
       ficha := a[posIni];
       largo := 1;
       i := posIni + 1;
       while (i <= MAX_FICHAS) and (a[i] = ficha) do begin
            i := i + 1;
            largo := largo + 1;
       end;
       largoBloque := largo;
 end;

var    i, largo : Integer;

begin
   i := 1;
   repeat
       largo := largoBloque(a, i);
       i := i + largo;
   until (i > MAX_FICHAS + 1 - n)  or (largo = n);

   if largo = n then
       indiceBloqueN := i - largo
   else
       indiceBloqueN := -1
end;
```

Ejercicio 2

Nota: En la solución se podía suponer que en avionesA hay por lo menos un avión con tipo tpasajeros y otro con tipo tcarga pues el promedio tiene sentido solo para uno o mas elementos.

```
procedure aterrizarAviones( avionesA: AvionesAterr;
                    var avionesP:  AvionesPasajeros;
                    var avionesC:  AvionesCarga;
                    var promedioP: real;
                    var promedioC: real);

var     pasajeros, nPasaj, nCarga, i: integer;
carga: real;

begin
      pasajeros:= 0;
      nPasaj := 0;
      nCarga := 0;
      carga  := 0;

      for i := 1 to maxAterr do begin
        case avionesA[i].tipo of
          tpasajeros : begin
                avionesP.tope := avionesP.tope+1;

                    avionesP.aviones[avionesP.tope].codigo := avionesA[i].codigo;
                    avionesP.aviones[avionesP.tope].pasajeros := avionesA[i].pasajeros;

                    pasajeros := pasajeros + avionesA[i].pasajeros;
                    nPasaj := nPasaj +1;
                  end;
          tcarga     : begin
                    avionesC.tope := avionesC.tope+1;

                    avionesC.aviones[avionesC.tope].codigo := avionesA[i].codigo;
                    avionesC.aviones[avionesC.tope].carga := avionesA[i].carga;

                    carga := carga + avionesA[i].carga;
                    nCarga := nCarga + 1;
                  end;
        end;
end;
promedioP :=  pasajeros / nPasaj;
promedioC := carga / nCarga;
```

```
end;
```

Ejercicio 3

```
procedure borrarExtremos(var l : ListaInt);

procedure borrarPrimero(var l : ListaInt);
var borrar : ListaInt;
begin
   if (l <> NIL) then begin
      borrar := l;
      l := l^.sig;
      DISPOSE(borrar);
   end;
end;

procedure borrarUltimo(var l : ListaInt);
var it : ListaInt;
begin
   if (l <> NIL) then begin
      if (l^.sig = NIL) then begin
         DISPOSE(l);
         l := NIL;
      end else begin
         it := l;
         while (it^.sig^.sig <> NIL) do
            it := it^.sig;
         DISPOSE(it^.sig);
         it^.sig := NIL;
      end;
   end;
end;

begin
  borrarPrimero(l);
  borrarUltimo(l);
end;
```