Ejercicio 1

Parte a)

```
function EsPosterior (f1, f2: Fecha): boolean;
begin
   EsPosterior := (f1.aa > f2.aa) or
                  (f1.aa = f2.aa) and (f1.mm > f2.mm) or
                  (f1.aa = f2.aa) and (f1.mm = f2.mm) and (f1.dd > f2.dd)
end;
```

Parte b)

```
function IndMaxFecha (arre: Fechas; pos: integer): integer;
var i, indMax: integer;
begin
   indMax := 1;
   for i := 2 to pos do
      if EsPosterior (arre[i], arre[indMax]) then
         indMax := i;
   IndMaxFecha := indMax;
end;
```

Parte c)

```
procedure OrdenarFechas (VAR arre: Fechas);
var
   i, indMax: 1..MAX;
   temp : Fecha;
begin
   for i := MAX downto 2 do
   begin
      indMax  := IndMaxFecha (arre, i);
      temp    := arre[indMax];
      arre[indMax] := arre[i];
      arre[i] := temp
   end
end;
```

Ejercicio 2

```
procedure normalizarArreglo (var arrTope : ArregloTope; N: Integer);

var
    i, valor : Integer;

begin
    (* si hay mas de N elementos en el arreglo,
       elimino los primeros hasta dejar el tope en N, manteniendo el orden *)
    if N < arrTope.tope then
       for i:= 1 to N do
          arrTope.valores[i] := arrTope.valores[i+(arrTope.tope-N)]
    else
    begin
       (* si hay menos de N elementos, busco el primer valor negativo
          y completo con ese valor hasta N.
          si no hay ningun negativo, completo con 0*)

       (* busco primer negativo o uso 0*)
       i := 1;
       while (i <= arrTope.tope) and (arrTope.valores[i] >= 0) do
                    i := i + 1;
       if i <= arrTope.tope then
          valor := arrTope.valores[i]
       else
          valor := 0;

       (* completo arreglo hasta N con valor *)
       for i := arrTope.tope + 1 to N do
          arrTope.valores[i] :=  valor
    end;
    arrTope.tope := N
end;
```

Ejercicio 3

```
function suma_saldos_persona (valor: Integer; lista: listaPersonas): Integer;
var cont : Integer;
begin
  cont := 0;

  while lista <> NIL do
  begin
     if lista^.tiene_saldo then
       if (lista^.saldoIRPF + lista^.saldoIASS > valor) then
        cont := cont + 1;
     lista := lista^.sig
  end;

  suma_saldos_persona := cont
end;
```

Ejercicio 4

```
=> 0
1 0 0
2
6

=> 1
2 2 2
3
8

=> 2
3 4 4
4
10

=> 3
4 6 6
5
12

=> 4
5 8 8
6
14

=> 5
6 10 10
7
16

=> 6
7 12 12
8
18

=> 7
8 14 14
9
20

=> 8
9 16 16
10
22

=> 9
10 18 18
11
24
```