```
módulo fechas
exporta
  tipo fecha
  prodecimiento crear(ent d,m,a:entero; sal f:fecha; sal error:booleano)
  función día(f:fecha) devuelve entero
  función mes(f:fecha) devuelve entero
  función año(f:fecha) devuelve entero
  función iguales(f1,f2:fecha) devuelve booleano  {no haría falta... "=")
  función anterior(f1,f2:fecha) devuelve booleano
  función posterior(f1,f2:fecha) devuelve booleano
implementación
  tipo fecha = registro
                 eldía,elmes,elaño:entero
               freg

  procedimiento crear(ent d,m,a:entero; sal f:fecha; sal error:booleano)
  principio
    si d<1 or d>31 or m<1 or m>12 or a<1583 or
       (d=31 and (m=2 or m=4 or m=6 or m=9 or m=11)) or
       (m=2 and d=30) entonces
       {1582=año del inicio de la adopción del calendario gregoriano}
      error:=verdad
    sino
      si m=2 and d=29 and
         ((a mod 4/=0) or
          (a mod 4=0 and a mod 100=0 and a mod 400/=0)) entonces
        error:=verdad
      sino
        f.eldía:=d; f.elmes:=m; f.elaño:=a; error:=falso
      fsi
    fsi
  fin

  función día(f:fecha) devuelve entero
  principio
    devuelve f.eldía
  fin
  función mes(f:fecha) devuelve entero
  principio
    devuelve f.elmes
  fin
  función año(f:fecha) devuelve entero
  principio
    devuelve f.elaño
  fin

  función iguales(f1,f2:fecha) devuelve booleano
  principio
    {devuelve f1=f2}

    devuelve((f1.elaño=f2.elaño) and (f1.elmes=f2.elmes) and (f1.eldia=f2.eldia))
  fin

  función anterior(f1,f2:fecha) devuelve booleano
  principio
    devuelve (f1.elaño<f2.elaño) or
     ((f1.elaño=f2.elaño) and (f1.elmes<f2.elmes)) or
     ((f1.elaño=f2.elaño) and (f1.elmes=f2.elmes) and (f1.eldia<f2.eldia))
  fin
```

```
  función posterior(f1,f2:fecha) devuelve booleano
  principio
    devuelve not(iguales(f1,f2) or (anterior(f1,f2))
  fin
fin
```