```
package Laboratorio03 is
    function f1 (X: in Integer) return Integer;
    --post: resultado = 3(4-5x)-2(5-6x)

    function f2 (A: in Integer) return Integer;
    --post: resultado = 2a(4a+3)-4(3a-7)

    function f3 (C: in Integer) return Integer;
    --post: resultado = (5c-7)^2

    function f4 (P,Q: in Integer) return Integer;
    --post: resultado = (3p+2q)(p-3q)

    function f5 (A,B: in Integer) return Integer;
    --post: resultado = 5(2a-b)

    function f6 (P,Q: in Integer) return Integer;
    --post: resultado = (3p+2q)(p-3q)

    function f7 (X,Y: in Float) return Float;
    --post: resultado = (2x-y+3)/(x^2+2y)


    procedure Porcentaje(A,B,C: in out Integer);
    -- post A = A_in / (A_in + B_in + C_in)
    --      B = B_in / (A_in + B_in + C_in)
    --      C = C_in / (A_in + B_in + C_in)


    function Imc(Peso_Kg,Altura_M: in Float) return Float;
    -- post: Resultado = peso /altura^2

    procedure Sumar(D1,D2: in Character; Resultado: out Character; Llevada: in out
    Boolean);
    -- post: Resultado = unidades de la suma de D1 + D2;
    --       LLevada = true <--> la suma es mas que 9

    procedure Restar(D1,D2: in Character; Resultado: out Character; Llevada:in out
    Boolean);
    -- post: Resultado = unidades de la resta de D1 + D2;
    --       LLevada = true <--> la suma es menos que 0

    function Area_Triangulo1 (Base,Altura: in Float) return Float;
    -- Pre: Base, Altura > 0
    --Post: Resultado es el área de un triángulo de base Base y altura Altura

    function Area_Triangulo2 (A,B,C: in Float) return Float;
    -- Pre: A,B,C > 0 y los lados corresponden a un triángulo
    --Post: Resultado es el área del triángulo con las medidas descritas

    function Area_Triangulo3 (A,B,C: in Float) return Float;
    -- Pre: A,B,C > 0 y A y B son longitudes de dos lados y C es el angulo (en
    grados) que forman dichos lados
    --Post: Resultado es el área del triángulo con las medidas descritas

    function Area_triangulo4 (A,B,C: in Float) return Float;
    -- Pre: A,B,C > 0 y A es la longitud de un lado y B y C los angulos (en grados)
    contiguos a dicho lado
    --Post: Resultado es el área del triángulo con las medidas descritas

    function Area_triangulo5 (lado_A,angulo_Contiguo,Angulo_No_Contiguo: in Float)
    return Float;
    -- Pre: A,B,C > 0 y lado_A es la longitud de un lado
    --      Angulo_Contiguo es el ángulo (en grados) contiguo al lado A
    --      Angulo_no_contiguo es el angulo (en grados) no contiguo al lado A
    --Post: Resultado es el área del triángulo con las medidas descritas

    function Area_triangulo6 (X1,Y1,X2,Y2,X3,Y3: in Float) return Float;
    -- Pre: (X1,Y1),(x2,Y2) y (X3,Y3) son las coordenadas de los vértices del
    triangulo
    --Post: Resultado es el área del triángulo con esos vertices


```

```
68
69      procedure hora (segundos: in Natural; hh,mm,ss: out Natural) ;
70      -- post: hh,mm,ss son las horas minutos y segundos (respectivamente) de segundos
71
72
73      function segundos (hh,mm,ss: in natural) return Natural;
74      --post: resultado es el nÃºmero de segundos de hh, mm, y ss
75
76
77      procedure sumar (hh1,mm1,ss1,hh2,mm2,ss2: in Natural; hh,mm,ss: out Natural);
78      --post: hh,mm,ss es el resultado de sumar (hh1:mm1:ss1) con (hh2:mm2:ss2)
79
80
81      procedure entero_a_digitos (num: in Natural; unidades, decenas, centenas, u_millar
        : out Natural);
82      --post: unidades = unidades de num; decenas =  decenas de num; centenas =
        centenas de num; u_millar = unidades de millar de num
83
84
85      procedure swap (c1, c2: in out Character);
86      --post: c1 y c2 tienen los valores intercambiados
87
88
89      function valor (unidades, decenas, centenas, u_millar: in Natural) return Integer;
90      --post: resultado = el valor del entero con u_millar unidades de millar, centenas
        centenas, decenas decenas y unidades unidades
91
92
93      function Shaw_Basho (x: in Natural) return Integer;
94      --post: resultado devuelve f(x) siendo X el polinomio de Shaw_basho
95
96
97      function multiplo_de_10 (x: in positive) return Boolean;
98      --post: resultado = true si X es multiplo de 10
99
100
101     procedure area_circulo (r: in Float; area: out Float);
102     --post: resultado = area del circulo de radio r
103
104
105     function volumen_cilindro (r: in Float; h: in Float) return Float;
106     --Post: resultado = volumen del cilindro de radio r y altura h
107
108
109     function volumen_esfera (r: in Float) return Float;
110     --Post: resultado = volumen de la esfera de radio r
111
112
113     procedure volumen_cilindro_semiesfera (r, h: in Float; volumen: out Float);
114     --Pre: r,h>0
115     --Post: Volumen de un cilindro de altura h y radio r coronado por una semiesfera
        de radio r
116
117     function multiplo_de(x,n: in Integer) return Boolean;
118     --post: True si y solo si X es multiplo de n
119
120     procedure a_trigonometrica(modulo_polar, argumento_polar: in Float; a,b: out Float
        );
121     --post: a es la primera coordenada del numero complejo; b es la segunda
        coordenada del numero complejo
122
123     function "+" (C: in Character; N: in Integer) return Character;
124     --pre: -256 < N < 256
125     --post: resultado es el caracter que resulta de avanzar N posiciones adelante (si
        positivo) o atras (si negativo)  a C.
126     --      Si nos salimos de la tabla, se recomienza por el principio (final, si es
        en inverso)
127
128     function es_bisiesto (anno: in Positive) return Boolean;
129     --post: resultado = true si y solo si anno es un anno bisiesto
130
131
132     procedure hoy (dd,mm,aa: out Positive);
```

```
133       --post: dd, mm aa son el dÃ-a mes y aÃ±o de hoy
134
135       function cuantos_dias(d1,m1,a1: in Positive; d2,m2,a2: in Positive) return Natural
          ;
136       --pre: d1,d1 <=31; m1,m2 <=12; a1,a2<=2100; d1-m1-a1 <= d2-m2-a2 (la fecha con 1
          es anterior o igual a la fecha con 2)
137       --post: resultado = numero de di-as entre d1-m1-a1 y d2-m2-a2
138
139       function Digito_Como_Entero(D: in Character) return Natural;
140       --pre: D entre '0' y '9'
141       --post: Resultado = el valor de D como entero
142
143       function Entero_Como_Digito(D: in Natural) return Character;
144       --pre: D entre 0 y 9
145       --post: Resultado = caracter que representa al digito D
146
147       procedure Suma (D1,D2: in Character; Resultado: out Character; Llevada: out
          Boolean);
148       --pre: D1,D2: dos dígitos
149       --post: Resultado = las unidades de D1+D2 (como enteros)
150       --      LLevada = true sii D1+D2>9
151
152       function Codificar_Fecha(D,M,A: in Positive) return Positive;
153       --pre: D,M,A>0.  D<=31, M<=12 (y reglas de concordancia del número de días con
          mes)
154       --post: Fecha es un entero en el que:
155       --      las cuatro cifras menos significativas son A,
156       --      las dos siguientes son M y
157       --      las siguientes (1 o 2) son D
158
159       procedure Decodificar_Fecha(F: in Positive; D,M,A: out Positive);
160       --pre: F es una fecha codificada:
161       --      las cuatro cifras menos significativas representan un año,
162       --      las dos siguientes corresponden a un mes (valores entre 1 y 12)
163       --      el resto (1 o 2) son un valor entre 1..31 (en conjuncion con el valor de
          M el limite está en 28,29,30,31)
164       --post: D es el valor de las primeras cifras
165       --      M es el valor de las dos siguientes
166       --      A es el valor del año
167
168   end Laboratorio03;
169
170
```