Ejercicio 8: Porcentajes, IVA e inversiones

1. Escribir un algoritmo que calcula el precio con todos los impuestos incluidos (TII) para un precio sin impuestos y un porcentaje de IVA dado.

```
Algoritmo PRECIO
        PRECIO(P: REAL) : REAL
        # Calcular precio con todos los impuestos
Entrada:
        P: REAL
        # Precio sin impuestos
        i: REAL
        # Porcentaje IVA
Precondición:
        Datos de entrada >= 0
Variables:
         Pf: REAL
        # Precio final con el impuesto añadido
Realización:
        Pf = P x (i/100) : REAL
Postcondición:
        Resultado= REAL(Pf)
Fin PRECIO
```

2. Escribir un algoritmo que calcula el importe de los intereses generados por un capital invertido a un interés dado durante un tiempo dado, expresado en meses.

Algoritmo INTERÉS

```
        # Dada una cuenta bancaria y un Interés simple
        # calcular el importe de los intereses generados
Entrada:
        Ci: >= 0
        # Capital inicial
Variables:
        Cf: REAL
        # Capital final con los intereses añadidos
        T: REAL
        # Tiempo
        i: REAL
        # Porcentaje IVA
        Ta: REAL
        # Tiempo en años
        Ig: REAL
        # Interés generado
Realización:
        Ta = t/12: REAL
        Cf= Ci x (1 +(i/100)Ta) : REAL
        Ig =  Cf – Ci : REAL
Postcondición:
        Resultado = REAL(Ig)
Fin INTERÉS
```

Ejercicio 9: Media aritmética ponderada

1.Escribir un algoritmo que calcula la media aritmética de tres números dados.

Algoritmo MEDIA
Calcular la media de 3 números Entrada: n1, n2, n3: REAL # Números a introducir Realización: S= n1+ n2 + n3 :REAL # Suma de los 3 números Vf = S/3: REAL # Valor final Postcondición: Resultado= REAL(Vf) Fin MEDIA

2. La misma pregunta para una media ponderada cuando se dan los números y los coeficientes de ponderación.

```
Algoritmo MEDIA_PONDERADA
# Calcular la media ponderada
Entrada:
        n1, n2, n3: REAL
        # Números a introducir
        p1, p2, p3: REAL
        # Porcentajes de los números
Realización:
        Vf = n1 x p1+ n2 x p2 + n3 x p3 : REAL
        # Valor final de la media ponderada
Postcondición:
        Resultado= REAL(Vf)
Fin MEDIA_PONDERADA
```

Ejercicio 10: Área del triángulo

1.Escribir un algoritmo que calcula el área de un triángulo del que se da la medida de un lado y la de la altura relativa a este lado.

```
Algoritmo ÁREA_TRIÁNGULO
# Calcular el área del triángulo
Entrada:
        L: REAL
        # Lado 1
         H: REAL
        # Altura Lado2
Realización:
        Rf= (L x H)/2 : REAL
        # Resultado final
Postcondición:
        Resultado= REAL(Rf)
Fin ÁREA_TRIÁNGULO
```

2. ¿Se puede utilizar este algoritmo para un triángulo rectángulo si se dan las medidas de sus dos lados perpendiculares?

Sí, ya que al ser perpendiculares uno de los 2 lados actúa como la altura y el otro como la base respectivamente.

Algoritmo ÁREA_TRIÁNGULO_LADOS
Calcular el área del triángulo dados 2 lados perpendiculares Precondición: Ángulo entre Lado 1 y Lado 2 (θ = 90°) Entrada: L1, L2: REAL # Lados dados Realización: Rf= (L1 x L2)/2 : REAL # Resultado final Postcondición: Resultado= REAL(Rf) Fin ÁREA_TRIÁNGULO_LADOS

Ejercicio 11: Salario y horas extra

Escribir el algoritmo que calcula el importe de las horas extra que hay que pagar, a partir del salario mensual bruto y de la cantidad de horas extra.

```
Algoritmo Salario_Horas_Extra
# Establece la paga de horas extra “He” para un salario mensual bruto “SMB”
Entrada
        SMB: REAL
        # Importe del salario mensual bruto
        He: ENTERO
        # Cantidad de horas extra del mes a remunerar
Precondición:
        SMB > 0
        He ≥ 0
Variables:
        Hm: ENTERO (8)
        # Horas máximas
        P1 : REAL (1,25)
        # Precio de HorasMax1 primeras horas extra
        P2 : REAL (1,5)
        # Precio de las otras horas extra
        He1 : ENTERO
        # Cantidad de horas extra con Precio 1 %
        He2 : ENTERO
        # Cantidad de horas extra con Precio 2 %
        PH: REAL
        # Precio hora bruto de base
Realización:
        # Cálculo de la cantidad de horas
        He1 = inf(He, Hm)
        He2 = sup(He – Hm, 0)
        # Cálculo del pago de las horas extra
```

```
        Resultado = PH x (He1 x P1 + He2 x P2)
Postcondición:
        Fin He
        Impuestos del estado (I): REAL
        ResultadoFinal: REAL (SMB + Resultado) x I
        # Cálculo del resultado final “ResultadoFinal”
Fin Salario_Horas_Extra
```

Además he creado el algoritmo que establece el precio hora bruto

```
Algoritmo Precio_Hora_Bruto
        # Precio_Hora_Bruto (SMB: REAL) : REAL
        # El precio hora bruto “PHB” correspondiente al salario mensual bruto “SMB”
Precondición
        SMB> 0
Constante
        NSem: ENTERO (52)
        # Número de semanas de trabajo
        HSem: ENTERO (40)
        # Número de horas de trabajo semanales
Realización
        # Cálculo del precio hora “PH”
        Resultado = SMB x 12,0 / (HSem x NSem)
Postcondición:
        Resultado = salario_mensual_bruto x 12,0 / REAL(40 x 52)
Fin Precio_Hora_Bruto
```

Ejercicio 12: Cuenta de depósito

Se considera las cuentas de depósitos alojadas en un banco por los clientes. Solo se permite hacer una retirada si el saldo que queda en la cuenta no es negativo.

1. Definir el tipo de datos CUENTA.

2. Definir las operaciones aplicables.

En determinadas circunstancias y para determinados clientes, la banca autoriza un descubierto limitado y temporal.

3. Volver a hacer las definiciones previas para permitir estos descubiertos.

Primero, he creado el algoritmo que se encarga de crear la cuenta bancaria.

```
Algoritmo Abrir_cuenta
Abrir_cuenta(C: CUENTA ; Si: REAL)
# Crear la cuenta “C” con un saldo inicial “Si”
Precondición
        Si > 0
Realización
        Cs = Si
        # El saldo de la cuenta (Cs) es igual al Saldo inicial (Si)
Postcondición
        Des = 0
      # El descubierto no está autorizado
        Antiguo(Si) = Si
        Cs = Si
Fin Abrir_cuenta
```

Segundo, he creado el algoritmo que permite abonar dinero a la cuenta.

```
Algoritmo Abonar
Abonar(C: CUENTA ; D : REAL)
        # Dinero “D” de la cuenta
        # Saldo “Sal”
Precondición
        Cs > 0
        D > 0
Realización
        Cs= CS+ D
Postcondición
        # El descubierto autorizado “Des” y el Dinero “D” no se modifican
        Antiguo(C).D = D
        Antiguo(C).Des = Des
        # El saldo aumenta con el dinero “D”
        Cs = Antiguo(C).Sal + D
Fin Abonar
```

Tercero, he creado el algoritmo que permite consultar la cuenta.

```
Algoritmo Consultar
Consultar(C : CUENTA) : REAL
        # El saldo “Sal” de la cuenta “C”
Precondición
        C.Sal > 0
Realización
        Resultado = C.Sal
Postcondición:
        Resultado = REAL(C.Sal)
Fin Consultar
```

Cuarto, he creado el algoritmo que permite cargar la cuenta con débito.

```
Algoritmo Cargar
Cargar(c : CUENTA ; Deb : REAL)
        # Carga la cuenta "C" con el débito "Deb"
Precondición
         Cs > 0
        Deb > 0
        Cs+ C.Des ≥ Deb ≥ 0
Realización
        abonar(C, -Deb)
Postcondición
        # El descubierto autorizado "Des" y el débito "Des" no se modifican
        Antiguo(C).Des = Des
        Antiguo(Deb) = Deb
         # Al saldo "Sal" se le resta el débito "Deb"
        C.Sal = Antiguo(C).Sal - Deb
Fin Cargar
```

Quinto, he creado el algoritmo que permite ver si la cuenta es deudora.

```
Algoritmo Deudora
Deudora (C : Cuenta) : Booleano
        # ¿Es la cuenta "C" deudora?
Precondición
        C.Sal > 0
Realización
        Resultado = (-C.Des≤ C.Sal ≤ 0)
Postcondición:
        Resultado = REAL(-C.Des≤ C.Sal ≤ 0)
fin Deudora
```

Sexto, he creado el algoritmo que permite ver si la cuenta es acreedora.

```
Algoritmo Acreedora
Acreedora (C: CUENTA) : BOOLEANO
        # ¿Es la cuenta "C" acreedora?
Precondición
        C.Sal > 0
Realización
        Resultado = (C.Sal > 0)
Postcondición:
        Resultado = REAL (C.Sal > 0)

Fin Acreedora
```

Séptimo, he creado el algoritmo que permite abrir una cuenta con el descubierto autorizado.

```
Algoritmo Abrir_Descubierto_Autorizado
Abrir_cuenta(C: CUENTA ;Si: REAL ;DesMax: REAL)
        #  Abrir una cuenta "C" con un saldo inicial "Si" y un descubierto máximo
        # "DesMax"
Precondición
        Si > 0
        DesMax ≥ 0
Realización
        C.Des = DesMax
        C.Sal = Si
Postcondición:
        C.Des = REAL(DesMax)
        C.Sal = REAL(Si)
Fin Abrir_Descubierto_Autorizado
```

Y por último, he creado el algoritmo que permite abrir una cuenta con el descubierto autorizado con una duración establecida.

Algoritmo Abrir_Descubierto_Autorizado_Duración
Algoritmo Abrir_cuenta # Abrir una cuenta “C” con un saldo inicial “Si” y un descubierto # máximo “DesMax” durante una duración máxima “DurMax” y una fecha # descubierta “Fd” Entrada C : CUENTA Si : REAL DesMax: REAL DurMax: FECHA Precondición Si > 0 DesMax ≥ 0 DurMax ≥ 0 Realización C.Des= DesMax C.Sal= Si C.DurMax = DurMax C.Fd= 0 Postcondición: C.Des= REAL(DesMax) C.Sal= REAL(Si) C.DurMax = REAL(DurMax) C.Fd= 0 fin Abrir_Descubierto_Autorizado_Duración