Fundamentos de Programación

Cuaderno de Trabajo 3a

Ejercicios resueltos

1. Cuando se incorporan las calificaciones de los alumnos a las actas universitarias es necesario expresar la calificación numérica y también la textual. Teniendo en cuenta que las calificaciones se relacionan según el siguiente criterio:

Rango de valores	Calificación textual
0<=nota<5	Suspenso
5<=nota<=6.5	Aprobado
6.5<nota<=8.5	Notable
8.5<nota<=10	Sobresaliente

haz un subprograma que reciba una calificación numérica comprendida entre 0 y 10 y devuelva la calificación textual.

SOLUCION PROPUESTA:

```
def nota_textual(nota):
        """
           float -> str
           OBJ: retorna el equivalente textual a una calificación numérica
           PRE: 0 <= nota <=10
        """     if 0 <=
    nota < 5:
            nota_texto = "Suspenso"
    elif 5 <= nota <= 6.5:
            nota_texto = "Aprobado"
    elif 6.5 < nota <= 8.5:
    nota_texto = "Notable"     else:
            nota_texto = "Sobresaliente"
    return nota_texto
```

Nota: *Observa que no hemos comprobado si la nota se encuentra en los límites correctos, ya que tal circunstancia se controla con la precondición.*

Ejercicios propuestos

1. Escribe un código que pide al vendedor el importe de una compra: Si la compra es superior a 100EUR se aplica un descuento del 5% si se paga al contado, pero si el pago es con tarjeta sólo se aplica el 2%. Asegúrate de que el importe de la compra es un número válido antes de proceder a los cálculos (pista: usa `try` para comprobar que es posible convertir la entrada a un `float`).

```
try:
    importe = float(input('Introduce el importe de la compra: '))
except:
    print('ERROR: El importe no es un número')
else:
    if importe > 100:
        tarjeta = input('¿Ha pagado con tarjeta (si/no)? ')
        if tarjeta == 'si':
            print(f'Tiene que pagar {importe*0.98}')
        elif tarjeta == 'no':
            print(f'Tiene que pagar {importe*0.95}')
        else:
            print('La respuesta no es correcta')
    else:
        print(f'Tiene que pagar {importe}')
```

2. Escribe un subprograma que lea una hora en notación de 24 horas y la devuelva en notación de 12 horas (ejemplo: las 18:30 serán las 6:30 PM). Pruébalo utilizando valores introducidos por el usuario, pero no olvides validad las entradas para asegurarte de que se trata de valores en el rango correcto.

```
def convertir_hora(hora, minutos):
    """ int, int --> str
    OBJ: Convierte una hora en formato 24h a 12h
    """
    resultado = ''
    if 0 <= hora <= 23 and 0 <= minutos <= 59:
        if hora == 0:
            resultado = f'12:{minutos} AM'
        elif hora > 12:
            resultado = f'{hora-12}:{minutos} PM'
        else:
            resultado = f'{hora}:{minutos} AM'
    return resultado


# Programa principal
try:
    hora = int(input('Introduce la hora: '))
    min = int(input('Introduce los minutos: '))
except:
    print('El valor introducido no es entero')
else:
    hora12 = convertir_hora(hora,min)
    if hora12 == '':
        print('La hora no es válida')
    else:
        print(f'La hora en formato 12h es {hora12}')
```

3. Como habrás observado, muy a menudo necesitamos validar la información introducida por el usuario. Programa una función "validar_entero" que se asegure de que una entrada del usuario es un entero. Ten en cuenta que dicha entrada puede ser cualquier cosa, por ejemplo, un valor real, un booleano o incluso un texto como "Hola". Nuestra función recibirá un texto y retornará verdadero si es un entero (dando por válida la entrada) o falso (rechazando la entrada).

```
def validar_entero(cadena):
    """ str --> bool
    OBJ: Comprueba que la cadena se puede convertir a entero
    """
    es_entero = False
    try:
        int(cadena)
        es_entero = True
    except:
        pass
    return es_entero

# Programa principal
a = input('Introduce un número: ')
if validar_entero(a):
    print('Es un número.')
    x = int(a)
else:
    print('No es un número.')
```

4. Reprograma lo hecho en el ejercicio 2 para reutilizar el validador de enteros del ejercicio 3. Obviamente, tendrás que seguir validando que los valores de hora y minutos se encuentran en el rango correcto.

5. Una universidad acaba de modificar su sistema de representación de la calificación de los alumnos. A partir de ahora, se calificarán como "A" las notas mayores o iguales a 8,5, "B" las mayores o iguales a 6,5, "C" las calificaciones mayores o iguales a 5, "D" las calificaciones mayores o iguales a 3,5, y "F" todas las demás. Programa una función que reciba una calificación numérica y retorne la letra con la nueva calificación, luego pruébala con valores introducidos por el usuario. Tal vez sea buena idea basarte en el ejercicio resuelto número 1 pero, a diferencia de lo que se hace allí (se controla la validez con una precondición), asegúrate de que la calificación introducida es válida.

```
def nota_a_texto(nota):
    """ float --> str
    OBJ: convierte una nota en formato decimal 0-10 a una escala A-F
    """
    nota_text = 'fuera de rango'
    if 0 <= nota <= 10:
        if nota >= 8.5:
            nota_text = 'A'
        elif nota >= 6.5:
            nota_text = 'B'
        elif nota >= 5:
            nota_text = 'C'
        elif nota >= 3.5:
            nota_text = 'D'
        else:
```

```
            nota_text = 'F'
    return nota_text

# Programa principal
try:
    mi_nota = float(input('Introduce una nota: '))
except:
    print('Error: no es un número')
else:
    print(f'La nota es: {nota_a_texto(mi_nota)}')
```

6. Escribe un algoritmo que tras leer tres enteros los escriba en pantalla en orden creciente. Como siempre, valida las entradas.

7. Programa una función que permita determinar si un determinado carácter es una vocal o no, utilizando la construcción `if-elif`. Repite el ejercicio utilizando esta vez el operador booleano **or** y observa las diferencias.

```
def es_vocal(caracter):
    """ str --> bool
    OBJ: Indica si un caracter es una vocal o no (incluye mayúsculas)
    """
    if caracter == 'a' or caracter == 'A':
        salida = True
    elif caracter == 'e' or caracter == 'E':
        salida = True
    elif caracter == 'i' or caracter == 'I':
        salida = True
    elif caracter == 'o' or caracter == 'O':
        salida = True
    elif caracter == 'u' or caracter == 'U':
        salida = True
    else:
        salida = False
    return salida

def es_vocal_or(caracter):
    """ str --> bool
    OBJ: Indica si un caracter es una vocal o no (incluye mayúsculas)
    """
    return caracter == 'a' or caracter == 'A' or caracter == 'e' or
caracter == 'E' or \
        caracter == 'i' or caracter == 'I' or caracter == 'o' or
caracter == 'O' or \
        caracter == 'u' or caracter == 'U'


# Programa principal
cadena = input('Introduce un caracter: ')
print(f'Es vocal (if-else): {es_vocal(cadena)}')
print(f'Es vocal (or): {es_vocal_or(cadena)}')
```

8. Codifica un subprograma que reciba un número entero, y si es entre 1 y 12 escriba un mensaje por pantalla indicando el mes a que dicho número corresponde. En caso contrario deberá mostrar un mensaje de error. Valida las entradas.

9. Obtén las raíces de una ecuación de segundo grado $ax^2+bx+c=0$ a partir de sus coeficientes a, b y c. Recuerda que si el discriminante es cero la raíz es única (b/2xa)

y que si el discriminante es negativo las raíces son imaginarias, en cuyo caso deberá indicarse con un mensaje "raíces imaginarias".

10. Haz un programa que pida al usuario un año, si lo introducido por el usuario es posterior a 1.582 imprimirá si es bisiesto, y si no, explicará el motivo. Recuerda reutilizar tu función **es_bisiesto** de cuadernos anteriores ¿Cuántas veces debes probar este ejercicio?

11. Escribe un programa que lea dos caracteres y averigüe si se introdujeron en el orden de su código ASCII o no. Se deberá además comprobar si el primero de ellos es una cifra, y en caso afirmativo indicar si es impar y si es o no primo. Modulariza la solución. En tu código no deben aparecer los valores de la tabla ASCII, porque produciría un código muy difícil de mantener, puedes usar **ord()** y **chr()** o simplemente saber que en la tabla ASCII todos los dígitos son consecutivos).

12. Escribe un programa que muestre un menú en pantalla que permita calcular el seno, coseno, tangente, cotangente, secante y cosecante de un ángulo. El menú se mostrará hasta que el usuario decida salir. El menú quedará:

```
1. Seno
2. Coseno
...
0. Salir
Elija una opción:
```

```
import math

opcion = -1
while opcion != 0:
    print('\t1. Seno')
    print('\t2. coseno')
    print('\t3. Tangente')
    print('\t4. Cotangente')
    print('\t5. Secante')
    print('\t6. Cosecante')
    print('\t0. Salir')
    try:
        opcion = int(input('Elija una opción: '))
    except:
        print('La opción no es válida')
    else:
        if 1 <= opcion <= 6:
            try:
                angulo = float(input('Introduzca un angulo (en radianes): '))
            except:
                print('El ángulo introducido no es válido')
            else:
                if opcion == 1:
                    print(f'El seno es {math.sin(angulo)}')
                elif opcion == 2:
                    print(f'El coseno es {math.cos(angulo)}')
                elif opcion == 3:
                    print(f'La tangente es {math.tan(angulo)}')
                elif opcion == 4:
                    print(f'La cotangente es {math.atan(angulo)}')
                elif opcion == 5:
                    print(f'La secante es {1 / math.cos(angulo)}')
                elif opcion == 6:
                    print(f'La cosecante es {1 / math.sin(angulo)}')
```

13. Una compañía de alquiler de automóviles desea adquirir un programa para emitir sus facturas, con las siguientes consideraciones:

a. Se factura una cantidad fija de 100EUR si no se rebasan los 300 Km.
b. Si la distancia recorrida es mayor que 300 Km, entonces:
    i. Si la distancia es menor o igual que 1.000 Km, se cobrarán 100EUR más el kilometraje que exceda de 300 Km, a razón de 30 céntimos/Km.
    ii. Si la distancia es mayor que 1.000 Km, se cobrarán los 100EUR más el kilometraje a razón de 30 céntimos/Km para los kilómetros entre el 300 y el 1.000 y 20 céntimos/Km para el resto.

14. En el baremo para la adjudicación de plazas en las Escuelas Infantiles Públicas se otorgan puntos en función de los ingresos de la unidad familiar, según se recoge en la siguiente tabla:

Puntos	Ingresos/mes € desde (incluido)	Hasta € (excluido)
1	5000	---
2	3500	5000
3	1800	3500
4	0	1800

Construye un subprograma que devuelva los puntos de un niño según los ingresos de su unidad familiar.

```
def baremacion_escuelas_infantiles(ingresos):
    """ float --> int
    OBJ: Indica el número de puntos en el baremo según ingresos
    """
    puntos = -1
    if ingresos >= 5000:
        puntos = 1
    elif ingresos >= 3500:
        puntos = 2
    elif ingresos >= 1000:
        puntos = 3
    elif ingresos >= 0:
        puntos = 4
    return puntos


# Probador
print(f"{baremacion_escuelas_infantiles(5000)}")
print(f"{baremacion_escuelas_infantiles(3500)}")
print(f"{baremacion_escuelas_infantiles(1800)}")
print(f"{baremacion_escuelas_infantiles(0)}")
```

15. Un monomio está caracterizado por un coeficiente (real) y una x elevada a un exponente (entero y positivo), si bien puede no llevar x. Como es sabido, un polinomio es una suma de monomios (Ejemplo: **+2.0x**3 + x + 4.0**)

Para escribir en pantalla un monomio, hay que tener en cuenta que:

a. Si el coeficiente es 0 no deberá imprimirse el monomio.
b. Si el coeficiente es 1 no imprimiremos el coeficiente.

c. Si el exponente es 1 no escribiremos el exponente

d. Si el exponente es 0 no escribiremos ni el polinomio ni la x.

Haz un subprograma que imprima un monomio. Advierte que Python no escribe el "+" en los números positivos y ahora hará falta hacerlo expresamente.