4.2.4. Mejorando el programa de los menús

Al acabar la sección dedicada a sentencias condicionales presentamos este programa:

circulo_4.py — circulo.py

```
1  from math import pi
2
3  radio = float(raw_input('Dame el radio de un círculo: '))
4
5  print 'Escoge una opción: '
6  print 'a) Calcular el diámetro.'
7  print 'b) Calcular el perímetro.'
8  print 'c) Calcular el área.'
9  opcion = raw_input('Teclea a, b o c y pulsa el retorno de carro: ')
10
11 if opcion == 'a':
12     diametro = 2 * radio
13     print 'El diámetro es', diametro
14 elif opcion == 'b':
15     perimetro = 2 * pi * radio
16     print 'El perímetro es', perimetro
17 elif opcion == 'c':
18     area = pi * radio ** 2
19     print 'El área es', area
20 else:
21     print 'Sólo hay tres opciones: a, b o c. Tú has tecleado', opcion
```

Y al empezar esta sección, dijimos que cuando el usuario no introduce correctamente una de las tres opciones del menú nos gustaría volver a mostrar el menú hasta que escoja una opción válida.

En principio, si queremos que el menú vuelva a aparecer por pantalla cuando el usuario se equivoca, deberemos repetir desde la línea 5 hasta la última, así que la sentencia **while** deberá aparecer inmediatamente después de la segunda línea. El borrador del programa puede quedar así:

circulo_13.py — circulo.py

```
1  from math import pi
2
3  radio = float(raw_input('Dame el radio de un círculo: '))
4
5  while opcion < 'a' or opcion > 'c':
6      print 'Escoge una opción: '
7      print 'a) Calcular el diámetro.'
8      print 'b) Calcular el perímetro.'
9      print 'c) Calcular el área.'
10     opcion = raw_input('Teclea a, b o c y pulsa el retorno de carro: ')
11     if opcion == 'a':
12         diametro = 2 * radio
13         print 'El diámetro es', diametro
14     elif opcion == 'b':
15         perimetro = 2 * pi * radio
16         print 'El perímetro es', perimetro
17     elif opcion == 'c':
18         area = pi * radio ** 2
19         print 'El área es', area
20     else:
21         print 'Sólo hay tres opciones: a, b o c. Tú has tecleado', opcion
```

Parece correcto, pero no lo es. ¿Por qué? El error estriba en que *opcion* no existe la primera vez que ejecutamos la línea 5. ¡Nos hemos olvidado de inicializar la variable

opcion! Desde luego, el valor inicial de *opcion* no debería ser `'a'`, `'b'` o `'c'`, pues entonces el bucle no se ejecutaría (piensa por qué). Cualquier otro valor hará que el programa funcione. Nosotros utilizaremos la cadena vacía para inicializar *opcion*:

circulo_14.py — circulo.py

```
from math import pi

radio = float(raw_input('Dame el radio de un círculo: '))

opcion = ''
while opcion < 'a' or opcion > 'c':
    print 'Escoge una opción: '
    print 'a) Calcular el diámetro.'
    print 'b) Calcular el perímetro.'
    print 'c) Calcular el área.'
    opcion = raw_input('Teclea a, b o c y pulsa el retorno de carro: ')
    if opcion == 'a':
        diametro = 2 * radio
        print 'El diámetro es', diametro
    elif opcion == 'b':
        perimetro = 2 * pi * radio
        print 'El perímetro es', perimetro
    elif opcion == 'c':
        area = pi * radio ** 2
        print 'El área es', area
    else:
        print 'Sólo hay tres opciones: a, b o c. Tú has tecleado', opcion
```

EJERCICIOS

▶ 114 ¿Es correcto este otro programa? ¿En qué se diferencia del anterior? ¿Cuál te parece mejor (si es que alguno de ellos te parece mejor)?

circulo_15.py — circulo.py

```
from math import pi

radio = float(raw_input('Dame el radio de un círculo: '))

opcion = ''
while opcion < 'a' or opcion > 'c':
    print 'Escoge una opción: '
    print 'a) Calcular el diámetro.'
    print 'b) Calcular el perímetro.'
    print 'c) Calcular el área.'
    opcion = raw_input('Teclea a, b o c y pulsa el retorno de carro: ')
    if opcion < 'a' or opcion > 'c':
        print 'Sólo hay tres opciones: a, b o c. Tú has tecleado', opcion

if opcion == 'a':
    diametro = 2 * radio
    print 'El diámetro es', diametro
elif opcion == 'b':
    perimetro = 2 * pi * radio
    print 'El perímetro es', perimetro
elif opcion == 'c':
    area = pi * radio ** 2
    print 'El área es', area
```

Es habitual que los programas con menú repitan una y otra vez las acciones de presentación del listado de opciones, lectura de selección y ejecución del cálculo. Una opción del menú permite finalizar el programa. Aquí tienes una nueva versión de `circulo.py` que finaliza cuando el usuario desea:

circulo_16.py — circulo.py

```
from math import pi

radio = float(raw_input('Dame el radio de un círculo: '))

opcion = ''
while opcion != 'd':
    print 'Escoge una opción: '
    print 'a) Calcular el diámetro.'
    print 'b) Calcular el perímetro.'
    print 'c) Calcular el área.'
    print 'd) Finalizar.'
    opcion = raw_input('Teclea a, b o c y pulsa el retorno de carro: ')
    if opcion == 'a':
        diametro = 2 * radio
        print 'El diámetro es', diametro
    elif opcion == 'b':
        perimetro = 2 * pi * radio
        print 'El perímetro es', perimetro
    elif opcion == 'c':
        area = pi * radio ** 2
        print 'El área es', area
    elif opcion != 'd':
        print 'Sólo hay cuatro opciones: a, b, c o d. Tú has tecleado', opcion

print 'Gracias por usar el programa'
```

EJERCICIOS

▶ 115 El programa anterior pide el valor del radio al principio y, después, permite seleccionar uno o más cálculos con ese valor del radio. Modifica el programa para que pida el valor del radio cada vez que se solicita efectuar un nuevo cálculo.

▶ 116 Un vector en un espacio tridimensional es una tripleta de valores reales (x, y, z). Deseamos confeccionar un programa que permita operar con dos vectores. El usuario verá en pantalla un menú con las siguientes opciones:

```
1) Introducir el primer vector
2) Introducir el segundo vector
3) Calcular la suma
4) Calcular la diferencia
5) Calcular el producto escalar
6) Calcular el producto vectorial
7) Calcular el ángulo (en grados) entre ellos
8) Calcular la longitud
9) Finalizar
```

Puede que necesites que te refresquemos la memoria sobre los cálculos a realizar. Si es así, la tabla 4.1 te será de ayuda:

Tras la ejecución de cada una de las acciones del menú éste reaparecerá en pantalla, a menos que la opción escogida sea la número 9. Si el usuario escoge una opción diferente, el programa advertirá al usuario de su error y el menú reaparecerá.

Las opciones 4 y 6 del menú pueden proporcionar resultados distintos en función del orden de los operandos, así que, si se escoge cualquiera de ellas, deberá mostrarse un

Operación	Cálculo
Suma: $(x_1, y_1, z_1) + (x_2, y_2, z_2)$	$(x_1 + x_2, y_1 + y_2, z_1 + z_2)$
Diferencia: $(x_1, y_1, z_1) - (x_2, y_2, z_2)$	$(x_1 - x_2, y_1 - y_2, z_1 - z_2)$
Producto escalar: $(x_1, y_1, z_1) \cdot (x_2, y_2, z_2)$	$x_1x_2 + y_1y_2 + z_1z_2$
Producto vectorial: $(x_1, y_1, z_1) \times (x_2, y_2, z_2)$	$(y_1z_2 - z_1y_2, z_1x_2 - x_1z_2, x_1y_2 - y_1x_2)$
Ángulo entre (x_1, y_1, z_1) y (x_2, y_2, z_2)	$\frac{180}{\pi} \cdot \arccos\left(\frac{x_1x_2 + y_1y_2 + z_1z_2}{\sqrt{x_1^2 + y_1^2 + z_1^2}\sqrt{x_2^2 + y_2^2 + z_2^2}}\right)$
Longitud de (x, y, z)	$\sqrt{x^2 + y^2 + z^2}$

Tabla 4.1: Recordatorio de operaciones básicas sobre vectores.

nuevo menú que permita seleccionar el orden de los operandos. Por ejemplo, la opción 4 mostrará el siguiente menú:

```
1) Primer vector menos segundo vector
2) Segundo vector menos primer vector
```

Nuevamente, si el usuario se equivoca, se le advertirá del error y se le permitirá corregirlo.

La opción 8 del menú principal conducirá también a un submenú para que el usuario decida sobre cuál de los dos vectores se aplica el cálculo de longitud.

Ten en cuenta que tu programa debe contemplar y controlar toda posible situación excepcional: divisiones por cero, raíces con argumento negativo, etcétera. (Nota: La función arcocoseno se encuentra disponible en el módulo *math* y su identificador es *acos*.)