DSIC
DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

BOLETIN DE EJERCICIOS
TEMA 1

Extraídos y adaptados de "Ramón Mollá Vayá; Inmaculada García García; Laura Sebastiá Tarín; Jon Ander Gómez Adrián; Jose Miguel Alonso Abalos; David Guerrero López; Miguel Ángel Martín Caro(2005). ***Problemas resueltos en lenguaje C***. Editorial UPV. 8497058836"

1. Una empresa de venta de recambios de automóviles necesita un programa que calcule y muestre el precio final en euros de un producto. Para ello, se debe aplicar la siguiente fórmula:

$$\text{precio\_neto} = \text{precio\_coste} * \frac{100 + \text{margen}}{100}$$

   El precio de coste en euros y el margen en tanto por ciento que desea obtener la empresa para el producto se introducirán por teclado.

Análisis

Una de las aplicaciones principales de un programa informático es la automatización de un proceso que transforma unos datos de entrada en unos datos de salida. Este proceso puede ser de muchos tipos: cálculos más o menos complejos, clasificaciones, ordenaciones, etc.

Cuando se va a utilizar un programa para automatizar un proceso, la primera actividad a realizar debe ser determinar cuáles son los datos de entrada al programa y los datos de salida que éste debe producir. En el caso que se va a resolver, se observa que los datos de entrada serán el precio de coste en euros y el margen que la empresa desea obtener, ya que a partir de estos datos, se calculará el precio neto, que es precisamente el dato de salida que el programa debe obtener.

En segundo lugar, se debe identificar el proceso que el programa debe realizar para transformar los datos de entrada en los datos de salida. En este problema, el enunciado ya indica la fórmula que se debe aplicar para obtener el precio neto.

Resumiendo:

1. datos de entrada: precio de coste y margen
2. datos de salida: precio neto
3. proceso: aplicar la fórmula $\text{precio\_neto} = \text{precio\_coste} * \frac{100 + \text{margen}}{100}$

Ahora se debe plasmar esto en un programa. Las tareas básicas que un programa debe realizar son:

1. Leer los datos de entrada y almacenarlos en variables
2. Obtener los datos de salida, que se almacenarán en otras variables, mediante la aplicación del proceso indicado sobre los datos de entrada
3. Escribir por pantalla los datos de salida

Así, el programa que resuelva este problema deberá:

1. Leer el precio de coste y el margen y almacenarlos en variables, que se llamarán, por ejemplo, `precio_coste` y `margen`.
2. Obtener el precio neto, almacenándolo en la variable `precio_neto`, mediante la aplicación de la fórmula:

$$\text{precio\_neto} = \text{precio\_coste} * \frac{100 + \text{margen}}{100}.$$

3. Escribir por pantalla el contenido de la variable `precio_neto`.

2. Escribir un programa que lea una cadena de caracteres en minúsculas e imprima en pantalla la misma cadena en mayúsculas.

Análisis

Al igual que en el problema anterior, se debe determinar en primer lugar, cuáles son los datos de entrada que recibirá el programa y cuáles los datos de salida que deberá producir. En este caso, hay un único dato de entrada, una cadena de caracteres en minúscula, que el proceso en cuestión debe transformar en la cadena en mayúscula correspondiente, que es el dato de salida. Así:

1. datos de entrada: cadena en minúscula
2. datos de salida: la misma cadena en mayúscula
3. proceso: transformar la cadena en minúscula a la correspondiente cadena en mayúscula

Por tanto, el programa deberá:

1. Leer una cadena de caracteres en minúscula que se almacenará en la variable `minuscula`
2. Transformar esta cadena en minúscula a la correspondiente cadena en mayúscula, que se almacenará en la variable `mayuscula`
3. Escribir en pantalla la cadena en mayúscula obtenida, es decir, el contenido de la variable `mayuscula`

Para realizar el segundo paso, se puede utilizar el método `upper()`, tal y como se explica en los apuntes.

3. Escribir un programa que lea los valores de los catetos de un triángulo rectángulo y calcule cuál es la hipotenusa, el área y el perímetro del triángulo mediante las siguientes expresiones:

$$h = \sqrt{c_1^2 + c_2^2} \qquad A = \frac{c_1 + c_2}{2} \qquad p = h + c_1 + c_2$$

Análisis

Este problema consiste en obtener distintos cálculos de un triángulo rectángulo a partir de sus catetos. Es decir, en este caso, hay dos datos de entrada (cada uno de los catetos) y tres datos de salida (hipotenusa, área y perímetro). El proceso para obtener los datos de salida a partir de los datos de entrada se resume en el enunciado en forma de expresiones a aplicar. Por tanto, en el primer nivel de análisis se determina:

1. datos de entrada: los dos catetos del triángulo rectángulo
2. datos de salida: hipotenusa, área y perímetro de este triángulo
3. proceso: obtener la hipotenusa, el área y el perímetro utilizando las expresiones proporcionadas en el enunciado

En cuanto a la parte del proceso, es importante darse cuenta de que para calcular la hipotenusa y el área únicamente se utilizan los datos de entrada, mientras que en el cálculo del perímetro se necesita también uno de los datos de salida. Esto determina un orden entre las operaciones, es decir, se deberá calcular antes la hipotenusa que el perímetro.

De este modo, el programa deberá:

1. Leer los valores de los catetos del triángulo rectángulo y almacenarlos en las variables `cateto1` y `cateto2`.
2. Calcular la hipotenusa y el área y almacenar los resultados en las variables `hipotenusa` y `area` respectivamente.
3. Calcular el perímetro utilizando el contenido de las variables `cateto1`, `cateto2` e `hipotenusa` y almacenar el resultado en la variable `perimetro`.
4. Escribir por pantalla el contenido de las variables `hipotenusa`, `area` y `perimetro`.

Una cuestión importante es determinar cómo se pueden escribir las expresiones que aparecen en el enunciado. Tanto en la expresión para calcular el área y como para calcular el perímetro solamente aparecen operaciones como la suma y la división, que ya se han implementado en problemas anteriores. Pero para resolver la expresión $h = \sqrt{c_1^2 + c_2^2}$ es necesario el cálculo del cuadrado y de la raíz cuadrada de un número.

El cuadrado de un número `x` es sencillo de implementar: basta con multiplicar `x` por sí mismo, es decir, `cuadrado_x = x*x`

Sin embargo, para calcular la raíz cuadrada de un número `x` es necesario utilizar una instrucción especial que ofrece el lenguaje: `sqrt`. Por tanto, la instrucción que calculará la raíz cuadrada de `x` es: `raiz_cuadrada_x = sqrt(x)`

Además, para poder utilizar esta instrucción especial se debe añadir al principio del programa la instrucción de importación del módulo `math` . Esto permite utilizar en el programa un conjunto de operaciones matemáticas como seno, coseno, potencia, etc.

En este caso, la expresión a escribir en C es: $h = \sqrt{c_1^2 + c_2^2}$. Si se escribe paso a paso, se obtiene:

```
cuadrado_cateto_1 = cateto1*cateto1
cuadrado_cateto_2 = cateto2*cateto2
suma_cuadrados_catetos = cuadrado_cateto_1+cuadrado_cateto_2
hipotenusa = sqrt(suma_cuadrados_catetos)
```

Sin embargo, es posible simplificar este segmento de código eliminando la variable `suma_cuadrados_catetos`:

```
cuadrado_cateto_1 = cateto1*cateto1
cuadrado_cateto_2 = cateto2*cateto2
hipotenusa = sqrt(cuadrado_cateto_1+cuadrado_cateto_2)
```

Todavía es posible simplificarlo más, calculando el cuadrado de los catetos directamente al obtener la hipotenusa:

```
hipotenusa=sqrt(cateto1*cateto1 + cateto2*cateto2)
```

4. La empresa que fabrica un modelo de máquinas expendedoras de refrescos necesita un programa para estas máquinas que realice el cálculo de cuántas monedas de cada tipo debe devolver. Para ello, en primer lugar, se introducirá por teclado la cantidad a devolver en euros (múltiplo de 5 céntimos, que es la moneda más pequeña de la que se dispone), es decir, se tecleará 1.85 para 1 euro con 85 céntimos. Este programa escribirá en pantalla cuántas monedas de cada tipo hay que devolver teniendo en cuenta que:
    - Se dispone de monedas de 50 céntimos, 20 céntimos, 10 céntimos y 5 céntimos.
    - Siempre se dispone de las monedas necesarias de cada tipo.
    - Se debe devolver el menor número de monedas posible, es decir, intentar devolver con las de mayor valor.

    Ejemplos:
    - Si se introduce la cantidad de 1 euro con 85 céntimos, el programa debe imprimir: 3 monedas de 50 céntimos, 1 moneda de 20 céntimos, 1 moneda de 10 céntimos, 1 moneda de 5 céntimos.
    - Si se introduce la cantidad de 1 euro con 20 céntimos, el programa debe imprimir: 2 monedas de 50 céntimos, 1 moneda de 20 céntimos, 0 monedas de 10 céntimos, 0 monedas de 5 céntimos.

Análisis

Este problema consiste en determinar cuántas monedas de cada tipo se necesitan para formar la cantidad que debe devolver la máquina expendedora, teniendo en cuenta que se debe utilizar el menor número posible de monedas de cada tipo. Esto supone que, hay un único dato de entrada (la cantidad a devolver), mientras que se tienen varios datos de salida (el número de monedas de cada tipo). Así:

- datos de entrada: cantidad a devolver
- datos de salida: cuántas monedas de 50 céntimos, cuántas de 20 céntimos, cuántas de 10 céntimos y cuántas de 5 céntimos se necesitan
- proceso: determinar cuántas monedas de cada tipo hacen falta para formar la cantidad a devolver

Una vez determinados los datos de entrada y de salida, habrá que pensar cómo es posible calcular el número de monedas de cada tipo que hacen falta. Un dato muy importante es el hecho de que se desea utilizar el menor número de monedas posible, lo que indica que se debe comenzar a emplear las monedas de mayor valor. Así, supongamos que la cantidad a devolver es 1.85 euros. Si se divide esta cantidad por 0.5 euros (que es el valor de la moneda de 50 céntimos) se obtiene 3.7, lo que significa que se necesitan 3 monedas de 50 céntimos y todavía queda cantidad por devolver. Es decir, ya se ha devuelto 1.50 euros con 3 monedas de 50 céntimos y quedan 0.35 euros (1.85 – 3*0.50) por devolver. Ahora si se divide 0.35 euros entre 0.2 euros (la moneda de 20 céntimos), se obtiene 1.75, por lo que se utiliza una moneda de 20 céntimos y todavía quedan 0.15 euros (0.35 – 1*0.20) por devolver. Si se divide 0.15 euros entre 0.1 euros (la moneda de 10 céntimos), el resultado es 1.5, lo que significa que se utiliza una moneda de 10 céntimos y quedan 0.05 euros por devolver, que al dividirlo por 0.05 euros (la moneda de 5 céntimos), se obtiene 1, es decir, una moneda de 5 céntimos y ya no queda nada por devolver.

Este proceso puede resumirse de la siguiente forma:

- dividir la cantidad a devolver por 0.5 para obtener el número de monedas de 50 céntimos a utilizar
- calcular lo que queda por devolver restando la cantidad a devolver menos el número de monedas de 50 céntimos utilizadas por 0.5 (que es su valor en euros)

- dividir la cantidad que queda a devolver por 0.2 para obtener el número de monedas de 20 céntimos a utilizar
- calcular lo que queda por devolver restando lo que quedaba por devolver menos el número de monedas de 20 céntimos utilizadas por 0.2 (que es su valor en euros)
- dividir la cantidad que queda a devolver por 0.1 para obtener el número de monedas de 10 céntimos a utilizar
- calcular lo que queda por devolver restando lo que quedaba por devolver menos el número de monedas de 10 céntimos utilizadas por 0.1 (que es su valor en euros)
- dividir la cantidad que queda a devolver por 0.05 para obtener el número de monedas de 5 céntimos a utilizar

Por tanto, el programa deberá:

1. Leer la cantidad a devolver en euros y almacenarlo en la variable `euros_a_devolver`. Copiar el contenido de `euros_a_devolver` en la variable `falta_por_devolver`, ya que todavía queda todo por devolver.
2. Dividir la variable `falta_por_devolver` entre 0.5 y almacenar la parte entera de la división en la variable `monedas_50c`.
3. Obtener la cantidad que queda por devolver restando la variable `falta_por_devolver` menos la variable `monedas_50c` por 0.5. Almacenar el resultado en la variable `falta_por_devolver`.
4. Dividir la variable `falta_por_devolver` entre 0.2 y almacenar la parte entera de la división en la variable `monedas_20c`.
5. Obtener la cantidad que queda por devolver restando la variable `falta_por_devolver` menos la variable `monedas_20c` por 0.2. Almacenar el resultado en la variable `falta_por_devolver`.
6. Dividir la variable `falta_por_devolver` entre 0.1 y almacenar la parte entera de la división en la variable `monedas_10c`.
7. Obtener la cantidad que queda por devolver restando la variable `falta_por_devolver` menos la variable `monedas_10c` por 0.1. Almacenar el resultado en la variable `falta_por_devolver`.
8. Dividir la variable `falta_por_devolver` entre 0.05 y almacenar la parte entera de la división en la variable `monedas_5c`.
9. Escribir en pantalla el contenido de las variables `monedas_50c`, `monedas_20c`, `monedas_10c` y `monedas_5c`.

Para obtener la parte entera de una cantidad, se puede utilizar las funciones `int` o `round`, tal y como se explica en los apuntes.

Ejercicios de expresiones

5. Traduce y evalúa las siguientes expresiones en Python:

   a) $2+(3*(6/2))$
   b) $\frac{4+6}{2+3}$
   c) $(4/2)^5$
   d) $(4/2)^{5+1}$
   e) $(-3)^2$
   f) $-(3^2)$

6. ¿Qué resultará de evaluar las siguientes expresiones?

   a) 1/2/4.0
   b) 1/2.0/4.0
   c) 1/2.0/4
   d) 1.0/2/4
   e) 4**.5
   f) 4.0**(1/2)
   g) 4.0**(1/2)+1/2
   h) 4.0**(1.0/2)+1/2.0
   i) 3e3/10
   j) 10/5e-3
   k) 10/5e-3+1
   l) 3/2+1

7. ¿Qué resultados se muestran al evaluar estas expresiones?

   a) True==True!=False
   b) $1 < 2 < 3 < 4 < 5$
   c) $(1 < 2 < 3)$ and $(4 < 5)$
   d) $1 < 2 < 4 < 3 < 5$
   e) $(1 < 2 < 4)$ and $(3 < 5)$

Ejercicios de entrada-salida y expresiones

8. Una compañía de refrescos comercializa tres productos: de cola, de naranja y de limón. Se desea realizar un programa que calcule las ventas realizadas de cada producto. Para ello, se leerá la cantidad vendida (máximo 5000000) y el precio en euros de cada producto y se mostrará un informe de ventas como el que sigue:

```
Producto      Ventas      Precio          Total
-------------------------------------------
Cola         1000000        0.17     170000.00
Naranja       350000        0.20      70000.00
Limon         530000        0.19     100700.00
                           TOTAL     340700.00
```

Para resolver este problema es necesario leer las ventas y el precio de cada uno de los productos y multiplicar los valores leídos para obtener el total. La salida debe mostrarse tabulada, utilizando especificaciones de formato más elaboradas en las instrucciones `print`.

9. Escribir un programa que muestra el resultado de la ecuación de tercer grado $y = ax^3 + bx^2 + cx + d$ para tres valores de x. Para ello, debe leer el valor de los coeficientes (*a, b, c y d*) y el valor de *x* y mostrar por pantalla el resultado de la evaluación de la ecuación resultante.
10. Escribir un programa que permita resolver una ecuación de primer grado $ax + b = c$ introduciendo los coeficientes *a*, *b* y *c* por teclado.
11. Escribir un programa que calcule el área y el perímetro de un rectángulo y muestre el resultado en pantalla, sabiendo que:

$$area = base * altura \qquad perimetro = 2 * base + 2 * altura$$

12. Escribir un programa que lea el número que ha salido en el sorteo de la ONCE e imprima en pantalla la última cifra de este número.
13. Escribir un programa que simule a una calculadora sencilla. Este programa pedirá dos números por teclado y calculará la suma, la resta, el producto y la división de ambos.
14. Escribir un programa que calcule el área y el perímetro de una circunferencia de radio R introducido por el usuario. Las fórmulas a aplicar son:

$$area = \Pi R^2$$

$$perimetro = 2\Pi R$$

15. Escribir un programa que pida el valor de los tres lados de un triángulo y calcule el valor de su área y perímetro aplicando la siguiente expresión:

    A = raiz(s(s − a)(s − b)(s − c)),donde s = (a + b + c)/2

    Comprueba que el programa funciona correctamente con este ejemplo: si los lados miden a=3, b=5 y c=7, el perímetro será 15.0 y el área 6.49519052838.
16. El área de un triángulo se puede calcular a partir del valor de dos de sus lados, a y b, y del ángulo θ que estos forman entre sí con la fórmula:

$$A = \frac{1}{2} ab\ sin(\theta)$$

Escribir un programa que pida al usuario el valor de los dos lados (en metros), el ángulo que estos forman (en grados), y muestre el valor del área. Se debe tener en cuenta que la función *sin* trabaja en radianes. Comprueba que el programa funciona correctamente con este ejemplo: a = 1, b = 2, θ = 30; el resultado es 0.5.

17. Escribir un programa que lea los valores de tres resistencias eléctricas (en Ohmios, W) y muestre en pantalla el valor global de la resistencia formada por estas tres resistencias si:

    a) están conectadas en paralelo: $R = \dfrac{1}{1/R_1 + 1/R_2 + 1/R_3}$

    b) están conectadas en serie: $R = R_1 + R_2 + R_3$

18. El servicio de endocrinología de un hospital necesita un programa para calcular el peso recomendado de una persona. Escribir un programa que lea la altura en metros y la edad de una persona y realice el cálculo del peso recomendado según la siguiente fórmula:

    peso = (altura en centímetros – 100 + 10% de la edad) * 0.9

19. Un asesor nos ha solicitado un programa para calcular los pagos mensuales de una hipoteca, de manera que pueda asesorar a sus clientes sobre ello. El programa debe solicitar el capital del préstamo (C), el interés anual (I) y el número de años (N) de la hipoteca y debe escribir la cuota a pagar mensualmente. Para calcular esta cuota se utiliza la siguiente fórmula, donde R es el interés mensual:

$$cuota = \frac{C * R}{1 - \left(\frac{1}{1+R}\right)^N} \qquad R = \frac{I/100}{12}$$

20. Escribir un programa que pida al usuario una cantidad en euros Ci, una tasa de interés t% y un número de años n. Muestra por pantalla en cuanto se habrá convertido el capital inicial transcurridos esos años si cada a˜ño se aplica la tasa de interés introducida, aplicando la siguiente fórmula:

$$Cf = Ci * (1 + t/100)^n$$

    Comprueba que el programa funciona correctamente con este ejemplo: una cantidad de 10000 euros al 4.5% de interés anual se convierte en 24117.14 euros al cabo de 20 años.˜

21. Escribir un programa que muestre por pantalla la ecuación de una recta en un plano, $Ax + By + C = 0$, leyendo previamente las coordenadas de dos de sus puntos (x_1, y_1) y (x_2, y_2), sabiendo que:

$$A = y_2 - y_1 \qquad B = y_1(x_2 - x_1) - x_1(y_2 - y_1)$$

22. Escribir un programa que calcule un determinante de 2º orden, sabiendo que:

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = a * d - c * b$$

23. Escribir un programa que resuelva sistemas de ecuaciones del siguiente tipo:

$$\left.\begin{aligned} ax + by &= c \\ a'x + b'y &= c' \end{aligned}\right\}$$

Este tipo de sistemas puede resolverse aplicando las siguientes expresiones:

$$y = \frac{a'c - ac'}{a'b - ab'} \qquad x = \frac{c - by}{a}$$

24. En un Campeonato de Atletismo se miden las marcas de los corredores de las distintas categorías en segundos porque resultan más apropiados para almacenarlos en un soporte informático. Sin embargo, resulta muy incómodo para los aficionados leer las marcas en segundos. Por esta razón, se desea un programa que lea un tiempo expresado en segundos y muestre su equivalencia en horas, minutos y segundos. Por ejemplo, si se introduce 4550 segundos, debe mostrar: 1 hora, 15 minutos y 50 segundos.
25. Escribir un programa que obtiene el valor de *y* de la siguiente ecuación, para un valor de *x* introducido por teclado:

$$y = \frac{x^2 - 4}{2} + \frac{3x - 7x^4}{-5x^3} + 4x - 2$$

26. Un grupo de amigos se ha encontrado un maletín con dinero mientras jugaba en un edificio abandonado. En vista de que nadie lo reclama, deciden repartirlo entre varias ONGs. Escribir un programa que lea la cantidad de dinero (en euros) que hay en el maletín y el número de ONGs y escriba por pantalla cuánto corresponderá a cada una.
27. La franquicia de tiendas de ropa Raza nos ha pedido que realicemos un programa para su terminal punto de venta. El programa deberá solicitar el código del artículo (máximo 6 cifras) a vender, su precio en euros, la cantidad de artículos que se desean y el descuento a aplicar (en tanto por ciento) sobre el precio inicial. Con esos datos, el programa debe mostrar por pantalla un ticket como el que sigue:

---- RAZA ----

codigo producto	precio
cantidad	precio total
descuento	descuento sobre el precio total
21% IVA	21% precio total sin descuento
TOTAL	precio final

Por ejemplo, si compramos 3 camisetas de código 123456, cuyo precio son 15’50€ y nos hacen un descuento del 10%, el ticket será:

---- RAZA ----

123456	15.50€
3	46.50€
-10%	-4.65€
21% IVA	+9.76€
TOTAL	51.61€

28. Los gastos de despacho de una empresa se clasifican en cuatro ítems: papelería, ordenadores, teléfono y mensajería. Se necesita un programa que controle estos gastos, de forma que, se introduzca el presupuesto fijado en la Junta de Propietarios y el gasto realmente realizado para cada uno de estos ítems y se muestre por pantalla la desviación tanto positiva como negativa del gasto con respecto al presupuesto.

Ejercicios con funciones básicas

29. Escribir un programa que nos permita obtener la longitud y el área de una circunferencia a partir del radio. El programa pedirá el radio al usuario. Por cada valor que el usuario indique mostrará ambos resultados. Este programa se debe estructurar en funciones. Concretamente, se debe implementar una función para calcular la longitud y otra función para calcular el área. Ambas recibirán como parámetro el radio de la circunferencia y devolverán el resultado calculado. El resultado se deberá imprimir en el programa principal.

Análisis

En este caso, en primer lugar, repetimos el análisis realizado en casos anteriores. Por tanto:

- Datos de entrada: radio de la circunferencia
- Datos de salida: longitud y área de la circunferencia
- Proceso: aplicar las fórmulas correspondientes de la longitud y el área

La diferencia con programas anteriores es que se pide que el proceso de implemente en funciones y no en el programa principal. Es decir, se definirá una función para calcular la longitud y otra para calcular el área. Ambas recibirán como parámetro el radio de la circunferencia y calculará el resultado correspondiente, que devolverán al programa principal. En el programa principal se llamará a estas funciones para obtener y mostrar los resultados por pantalla.

En cada función, seguimos el análisis como si fuera un programa aparte. Por ejemplo, para la función del cálculo de la longitud, determinamos:

- Datos de entrada: radio de la circunferencia. En lugar de utilizar una instrucción de lectura, será un parámetro de la función, especificado en la cabecera:
  `def calcula_longitud(radio):`
- Datos de salida: longitud de la circunferencia. En lugar de mostrar el resultado por pantalla, se utilizará `return` para enviar el resultado al programa principal.
- Proceso: cálculo de la longitud de la circunferencia: `longitud = 2*math.pi*radio`

Para la función para calcular el área se seguiría un proceso similar.

Finalmente, se debe llamar a las funciones desde el programa principal pasando el parámetro correspondiente y recoger el valor que devuelven. Por ejemplo:

```
r = float(input("Introduce el radio:"))
l = calcula_longitud(r)
print ("La longitud de la circunfencia es: ", l)
```

Es importante destacar que el nombre de las variables del programa principal y de la función no es necesario que coincidan.

En este pequeño ejemplo no se aprecian las ventajas que aportan las funciones en cuanto a modularidad de los programas, y en cuanto a ahorro de líneas de programa. Pero creemos que sí es un buen ejemplo del uso de funciones.

30. Sabiendo que el área de la superficie de una esfera de radio **r** se obtiene son la fórmula $S=4\pi\mathbf{r}^2$, y que el volumen es $V=^4/_3\pi\mathbf{r}^3$. Añadir dos funciones adicionales al programa anterior para mostrar también la superficie y el volumen de una esfera con el mismo radio.
31. Las resistencias eléctricas suelen ir identificadas por un código de colores que permite marcar cada resistencia con su valor (en Ohmios, Ω) y su Tolerancia (en %). Este código de colores viene representado en la siguiente tabla:

Dígito	Color	Multiplicador	Tolerancia
	Ninguno		20%
	Plata	0.01	10%
	Oro	0.1	5%
0	Negro	1	
1	Marrón	10	
2	Rojo	10^2	2%
3	Naranja	10^3	
4	Amarillo	10^4	
5	Verde	10^5	
6	Azul	10^6	
7	Violeta	10^7	
8	Gris		
9	Blanco		

El código que suele emplearse en las resistencias es un código de 4 colores, es decir, cada resistencia está marcada con 4 bandas y cada una de ellas puede ser de diferente color. Cada banda tiene un significado, que depende de cada color:

Las primeras 2 bandas indican un número de 2 dígitos: Esos dos dígitos vienen dados por el color de esas bandas, según la columna "Dígito" de la tabla.

La tercera banda es un valor por el que se multiplicará el número obtenido por las bandas anteriores. Una vez multiplicados ambos valores, obtenemos el valor de la resistencia en Ohmios (Ω).

La cuarta banda indica la tolerancia de la resistencia y, como puede verse en la tabla, no puede ser de cualquier color.

Ejemplo: Unas resistencias con los siguientes colores, tienen los siguientes valores de resistencia y tolerancia:

Verde-Azul-Amarillo-Oro 560 k Ω, 5%

Rojo-Negro-Rojo-Rojo 2 k Ω, 2%

Rojo-Rojo-Marrón-Plata 220 Ω, 10%

Según todo lo anterior, se pide implementar un programa que permita calcular la resistencia y la tolerancia de una resistencia, sabiendo los códigos de colores y suponiendo que los colores Oro, Plata y Ninguno tomarán los valores 10, 11 y 12 respectivamente. Para ello, se deben implementar las siguientes funciones:

- Una función que muestre por pantalla el dígito que le corresponde a cada color (incluyendo los dígitos 10, 11 y 12).
- Una función que tendrá, como mínimo, cuatro argumentos, que serán números naturales, y que indicarán el color de las bandas según la columna "Dígito". Esta función escribirá en pantalla el valor correspondiente de la resistencia y de la tolerancia.
- Una función que leerá cuatro valores naturales correspondientes a las cuatro bandas, validando que el dígito introducido es posible para esa banda. Es decir, el valor 4 no puede aparecer en la cuarta banda. Esta función llamará a la función anterior para obtener en pantalla el valor de la resistencia y de la tolerancia.

El programa comenzará mostrando el dígito que le corresponde a cada color usando la primera función. Después leerá de teclado cuatro números que corresponderán a los colores de las cuatro bandas y, tras esta lectura mostrará los datos de la resistencia con esos colores en las bandas, utilizando las funciones anteriores. Este proceso se repetirá indefinidamente hasta que se lea un valor negativo como color de una banda.