1. Diagramas de flujo - Ejercicios

Debes realizar los siguientes ejercicios utilizando la aplicación "yEd Graph Editor". Esta aplicación es gratuita, puedes descargarla desde el enlace:

https://www.yworks.com/products/yed/download#download

También puedes utilizar la versión online a través de enlace:

https://www.yworks.com/yed-live/

Deberás de entregar en la actividad de Classroom el archivo en formato graphml y una imagen en formato jpg de cada uno de los ejercicios.

---

1. Diseña el diagrama de flujo para sumar dos números leídos por teclado y escribir el resultado por pantalla.
2. Diseña un diagrama de flujo que permita leer 2 números diferentes y nos diga cuál es el mayor de los 2 números.
3. Crea un diagrama de flujo de procesos en el que se almacenen 3 números en 3 variables A, B y C. El diagrama debe decidir cuál es el mayor y cuál es el menor.
4. Realiza el diagrama de flujo para que nos calcule la hipotenusa de un triángulo rectángulo, conocidos sus dos catetos.
5. Diagrama de Flujo para sumar 100 números leídos por teclado.
6. Modifica el diagrama de flujo anterior para que permita sumar N números. El valor de N se debe leer previamente por teclado.
7. Crea un diagrama de flujo que permita escribir los 100 primeros pares.
8. Modifica el diagrama de flujo anterior para que permita sumar los N primeros impares. ¿Y para sumar los múltiplos de 3?
9. Diseña un diagrama de flujo que simule el funcionamiento de un reloj.
10. Crea un diagrama de flujo que lea N números, calcule y Escribe la suma de los pares y el producto de los impares.
11. Introducida una serie de N números a través del teclado, mostrar por pantalla el máximo de todos ellos.
12. Un año es bisiesto si es múltiplo de 4, exceptuando los múltiplos de 100, que sólo son bisiestos cuando son múltiplos además de 400, por ejemplo, el año 1900 no fue bisiesto, pero el año 2000 si lo será. Hacer un organigrama que dado un año A nos diga si es o no bisiesto.
13. Dados dos números enteros positivos N y D, se dice que D es un divisor de N si el resto de dividir N entre D es 0. Se dice que un número N es perfecto si la suma de sus divisores (excluido el propio N) es N. Por ejemplo 28 es perfecto, pues sus divisores (excluido el 28)

son: 1, 2, 4, 7 y 14 y su suma es 1+2+4+7+14=28. Hacer un organigrama que dado un número N nos diga si es o no perfecto.

2. Ejercicios desde la línea de comandos

Deberás de entregar en la actividad de Classroom un único archivo de texto (Word, Writer, GoogleDocs…) en el que habrás insertado las capturas de pantalla de tus respuestas a cada uno de los ejercicios propuestos.

Realiza los siguientes ejercicios utilizando la consola de Windows. Realiza una captura de pantalla de cada una de las soluciones y guarda las mismas en un archivo de texto que entregarás a tu profesor al terminar.

1. Utilizando variables con nombres apropiados, utiliza Python para calcular la superficie de la base de un cilindro y su volumen. Ambos valores deberán ser guardados en dos variables diferentes.
2. Utiliza el modificador necesario y una única función print() para generar el mensaje "Bienvenido a Python" de tal forma que cada palabra esté en una línea de texto diferente.
3. Utilizando una única función print(), genera el siguiente mensaje por pantalla: \nothing\=nada en Inglés.
4. Utiliza los modificadores necesarios para crear el siguiente mensaje tabulado:

```
Rafael Nadal:   6   3   6       5   40
Andy Murray:    4   6   7       3   15
```

5. Crea una variable con el nombre miNombre y almacena en ella tu nombre. Utiliza ahora una función print() para crear la expresión "Encantado de conocerte miNombre". Donde miNombre ha de ser el valor guardado en la variable.
6. Utiliza la opción de la tres comillas para generar con una única función print() el siguiente mensaje.

```
País: España
Superficie: 505,990 km2
Población: 46,77 millones de habitantes
Densidad de población: 92,43 habitantes/km2
```

7. Utilizando una única función print(), en la cual la longitud total de todos sus argumentos contenidos entre los paréntesis no puede ser mayor de 13 caracteres, genera el mensaje 'LaLaLaLaLaLaDoDo'
8. Crea una variable a la que llamaremos nombreCompleto y asígnale como valor tu nombre y apellidos completos. Utiliza la función len() para calcular la longitud de la cadena de texto asignada a la variable.
9. Utilizando la variable del ejercicio anterior utiliza la función print() para mostrar:
   a) El tercer carácter de la cadena de texto.
   b) El grupo de caracteres contenidos entre la posición 3 y 8 (ambos incluidos).
   c) El penúltimo carácter de la cadena de texto.
   d) El grupo de los cinco últimos caracteres de la cadena de texto.

10. Crea una variable tipo lista con el nombre datos. Los valores incluidos en los datos de la lista serán; Jesús, Sánchez, Pérez, 25, 1.84, 72. Estos datos se corresponden con el nombre, primer apellido, segundo apellido, edad, altura y peso de una persona. A través de sucesivas funciones print() muestra cada uno de los cinco índices almacenados en la variable.
11. Utilizando los datos introducidos en la variable del ejercicio anterior modifica el valor del segundo apellido y haz que ahora tome el valor Gracia.
12. El índice de masa corporal (IMC) de una persona se calcula dividiendo su masa (en kg) por el cuadrado de su altura (en m). Calcula el IMC de la persona descrita a través de la variable datos.
13. Utilizando una única orden, modifica el valor de la edad, la altura y la masa en la variable datos del ejercicio 10.
14. Crea una variable tipo lista con el nombre serie. Estará formada por cinco elementos que se corresponden con la proporción aritmética (1,6, 11, 16, 21). Una vez creada la variable, añade dos nuevos índices que se correspondan con los elementos siguientes de la serie.
15. Utiliza la función len() para determinar el número de índices en la variable del ejercicio anterior, antes y después de añadir los dos nuevos elementos.

3. Entrada y Salida de Datos

1. Crear una aplicación que le permita al usuario ingresar 2 números y luego mostrar en pantalla la suma de ambos.
2. Se desea elaborar un algoritmo que permita calcular y mostrar velocidad final, utilizando la siguiente formula: $\mathbf{v_t = v_o + gt}$. Donde:
   a) v_t : Velocidad instantánea(m/s)
   b) v_o: Velocidad inicial (m/s)
   c) g: Gravedad (considere g= 9.8 m/s^2)
   d) t: tiempo
3. Se desea tener un algoritmo que permita aceptar dos números enteros, seguidamente determinar y mostrar la suma, resta y multiplicación de estos números.
4. Calcular el perímetro y área de un rectángulo dada su base y su altura.
5. Calcular el perímetro y área de un círculo dado su radio.
6. Escribir un programa que le pida una palabra al usuario, para luego imprimirla 1000 veces, con espacios intermedios.
7. Realiza un programa que reciba una cantidad de minutos y muestre por pantalla a cuantas horas y minutos corresponde.
8. Se desea tener un algoritmo que permita determinar y mostrar el promedio que ha obtenido un alumno en un determinado curso, conociendo las notas de: tres prácticas, el examen parcial y el examen final.
9. Escribir un programa que pida al usuario su peso (en kg) y estatura (en metros), calcule el índice de masa corporal y lo almacene en una variable, y muestre por pantalla la frase “Tu índice de masa corporal es <imc>”, donde <imc> es el índice de masa corporal calculado.
10. Escribir un programa que pida al usuario dos números enteros y muestre por pantalla ”<n> entre <m> da un cociente <c> y un resto <r>”, donde <n> y <m> son los números introducidos por el usuario, y <c> y <r> son el cociente y el resto de la división entera respectivamente.
11. Escribir un programa que pregunte al usuario una cantidad a invertir, el interés anual y el número de años, y muestre por pantalla el capital obtenido en la inversión.
12. Una juguetería tiene mucho éxito en dos de sus productos: payasos y muñecas. Suele hacer venta por correo y la empresa de logística les cobra por peso de cada paquete así que deben calcular el peso de los payasos y muñecas que saldrán en cada paquete a demanda. Cada payaso pesa 112 g y cada muñeca 75 g. Escribir un programa que lea el número de payasos y muñecas vendidos en el último pedido y calcule el peso total del paquete que será enviado.
13. Imagina que acabas de abrir una nueva cuenta de ahorros que te ofrece el 4% de interés al año. Estos ahorros debido a intereses, que no se cobran hasta finales de año, se te añaden al balance final de tu cuenta de ahorros. Escribir un programa que comience leyendo la cantidad de dinero depositada en la cuenta de ahorros, introducida por el usuario. Después

el programa debe calcular y mostrar por pantalla la cantidad de ahorros tras el primer, segundo y tercer años. Redondear cada cantidad a dos decimales.

14. Una panadería vende barras de pan a 3.49€ cada una. El pan que no es el día tiene un descuento del 60%. Escribir un programa que comience leyendo el número de barras vendidas que no son del día. Después el programa debe mostrar el precio habitual de una barra de pan, el descuento que se le hace por no ser fresca y el coste final total.

4. if… elif… else…

Realiza todos los ejercicios de esta hoja en un único proyecto de Eclipse. Envía el resultado a través de Classroom como un archivo comprimido.

1. Escribe un programa que pida por teclado dos números enteros. A continuación, el programa debe calcular su división, escribiendo el cociente entero, en caso de que el resto no sea cero, habrá que mostrar también el valor del resto entero.

```
Divisor de números
Escribe el dividendo: 14
Escribe el divisor: 5
La división no es exacta. Cociente: 2 Resto: 4
```

```
Divisor de números
Escribe el dividendo: 20
Escribe el divisor: 4
La división es exacta. Cociente: 5
```

Nota: Se puede mejorar el programa haciendo que tenga en cuenta que no se puede dividir por cero:

```
Divisor de números
Escribe el dividendo: 20
Escribe el divisor: 0
No se puede dividir por 0
```

2. Escribe un programa que pida por teclado dos números y que indique cuál de ellos es el menor y cuál el mayor o bien si ambos números son iguales.

```
Comparador de números
Escribe un número: 23
Escribe otro número: 14.5
Menor: 14.5; Mayor: 23.0
```

```
Comparador de números
Escribe un número: 5.0
Escribe otro número: 5
Los dos números son iguales
```

3. Escribe un programa que pida por teclado en primer lugar el año actual y después un año cualquiera. A continuación, el programa ha de indicar cuántos años faltan para llegar o cuantos años han pasado al/desde ese segundo año.

```
Comparador de años
¿En qué año estamos?: 2020
Escribe un año cualquiera: 2025
Para llegar al año 2025 faltan 5 años.
```

```
Comparador de años
¿En qué año estamos?: 2020
Escribe un año cualquiera: 1997
```

```
Desde el año 1997 han pasado 23 años.
```

```
Comparador de años
¿En qué año estamos?: 2056
Escribe un año cualquiera: 2056
¡Son el mismo año!
```

Nota: Se puede mejorar el programa haciendo que cuando la diferencia sea exactamente un año y Escribe la frase en singular:

```
Comparador de años
¿En qué año estamos?: 2015
Escribe un año cualquiera: 2016
Para llegar al año 2016 falta 1 año.
```

4. Escribe un programa que pida dos números enteros y que a continuación indique si el número mayor es múltiplo del menor.

```
Comparador de múltiplos
Escribe un número: 48
Escribe otro número: 6
48 es múltiplo de 6.
```

```
Comparador de múltiplos
Escribe un número: 6
Escribe otro número: 48
48 es múltiplo de 6.
```

```
Comparador de múltiplos
Escribe un número: 6
Escribe otro número: 49
49 no es múltiplo de 6.
```

```
Comparador de múltiplos
Escribe un número: 6
Escribe otro número: 6
6 es múltiplo de 6.
```

5. Escribe un programa que pida tres números. El programa ha de indicar a continuación si estos tres valores son iguales, si dos de ellos son iguales o si son los tres distintos.

```
Comparador de tres números
Escribe un número: 6
Escribe otro número: 6
Escribe otro número más: 6
Ha escrito tres veces el mismo número.
```

```
Comparador de tres números
Escribe un número: 6
Escribe otro número: 6.5
Escribe otro número más: 6
Ha escrito uno de los números dos veces.
```

```
Comparador de tres números
Escribe un número: 4
```

```
Escribe otro número: 5
Escribe otro número más: 6
Los tres números que ha escrito son distintos.
```

6. Escribe un programa que pida un año y que determine si ese año es bisiesto o no.

> Recuerda que los años bisiestos son múltiplos de 4, pero los múltiplos de 100 no lo son, aunque los múltiplos de 400 sí.

```
Comprobador de años bisiestos
Escribe un año y le diré si es bisiesto: 2012
El año 2012 es un año bisiesto porque es múltiplo de 4.
```

```
Comprobador de años bisiestos
Escribe un año y le diré si es bisiesto: 2010
El año 2010 no es un año bisiesto.
```

```
Comprobador de años bisiestos
Escribe un año y le diré si es bisiesto: 2000
El año 2000 es un año bisiesto porque es múltiplo de 400.
```

```
Comprobador de años bisiestos
Escribe un año y le diré si es bisiesto: 1900
El año 1900 no es un año bisiesto porque es múltiplo de 100 sin ser múltiplo de 400.
```

7. Escribe un programa que solicite los coeficientes de una ecuación de primer grado (a.x + b = 0), y que a continuación calcule y muestre como resultado la solución a la ecuación.

> Una ecuación de primer grado puede no tener solución, tener una solución única, o que todos los números sean solución. La fórmula de las soluciones es x = -b / a

Estos son algunos ejemplos de posibles respuestas:

a	b	Solución
0	3	Sin solución
4.2	21	Una solución: -5
0	0	Todos los números son solución

```
Ecuación a x + b = 0
Escribe el valor del coeficiente a: 4.2
Escribe el valor del coeficiente b: 21
La solución de la ecuación es: -5.0
```

8. Escribe un programa que pida los coeficientes de una ecuación de segundo grado ($a x^2 + b x + c = 0$) y que a continuación muestre la/s soluciones.

> Una ecuación de segundo grado puede no tener solución, tener una solución única, tener dos soluciones o que todos los números sean solución.

Estos son algunos ejemplos de posibles respuestas.

a	b	c	Solución
1	-2	2	Sin solución real
2	-7	3	Dos soluciones: 0.5 y 3.0

1	2	1	Unasolución: -1.0
0	0	5	Sin solución
0	0	0	Todos los números son solución
0	3	2	Una solución: -0.666…

```
Ecuación a x² + b x + c = 0
Escribe el valor del coeficiente a: 2
Escribe el valor del coeficiente b: -7
Escribe el valor del coeficiente c: 3
Las soluciones de la ecuación son 3.0 y 0.5
```

9. Escribe un programa que en primer lugar pregunte si se quiere calcular el área de un triángulo o la de un círculo.
   a) Si se contesta que se quiere calcular el área de un triángulo, el programa tiene que pedir entonces la base y la altura y escribir el área.
   b) Si se contesta que se quiere calcular el área de un círculo, el programa tiene que pedir entonces el radio y escribir el área.
   c) En ambos casos el programa debe ser capaz de calcular y mostrar el resultado adecuado.

```
Cálculo de áreas - Elige una figura geométrica:
a) Triángulo
b) Círculo
¿Qué figura quieres calcular (Escribe T o C)? T
Escribe la base: 3
Escribe la altura: 5.5
Un triángulo de base 3.0 y altura 5.0 tiene un área de 8.25
```

```
Cálculo de áreas - Elige una figura geométrica:
a) Triángulo
b) Círculo
¿Qué figura quieres calcular (Escribe T o C)? C
Escribe el radio: 2
Un círculo de radio 2.0 tiene un área de 12.566370614359172
```

10. Escribe un programa que pida una distancia en centímetros y que Escribe esa distancia en kilómetros, metros y centímetros (escribiendo solamente las unidades necesarias).

Distancia en cm	Distancia en km, m y cm
100	1 m
240005	2 km, 400 m, 5 cm
67	67 cm
300004	3 km, 4 cm

La dificultad del ejercicio se puede reducir o aumentar según la forma de presentar el resultado:

- sin separador entre unidades: 2 km 400 m 5 cm
- separando con comas las unidades: 2 km, 400 m, 5 cm
- separando con comas y con la conjunción 'y' en la última unidad: 2 km, 400 m y 5 cm

```
Convertidor de centímetros a kilómetros, metros y centímetros
```

```
Escribe una distancia en centímetros: 43210
43210 centímetros son 432 m, 10 cm.
```

5. Bucle for (1)

1. Utilizando cinco bucles tipo for y en cada uno de ellos el tipo range() con tres argumentos, escribe el código Python necesario para que se muestre por pantalla la siguiente información tabulada con las siguientes cinco series aritméticas:

```
1   2   3   4   5   6   7   8   9   10
2   4   6   8   10  12  14  16  18  20
20  23  26  29  32  35  38  41  44  47
10  14  18  22  26  30
40  35  30  25  20  15  10  5   0
```

2. Crea un programa que muestre el mismo resultado que en el ejercicio anterior, pero utilizando ahora bucles tipo for con tipos range() de **dos** argumentos.

3. Crea un programa que muestre la tabla del ejercicio número 1, utilizando bucles tipo for con tipos range() que tengan solamente un argumento.

4. Escribe el código necesario para generar las siguientes siete secuencias de números utilizando bucles for.

```
1   4   9   16  25  36  49  64  81  100
2   5   10  17  26  37  50  65  82  101
8   27  64  125 216 343
2   6   12  20  30  42  56
1   10  100 1000 10000
1.0 0.1 0.01 0.001 0.0001
1   -1  1   -1  1   -1  1   -1
```

5. Escribe un programa que pida dos números enteros, el segundo ha de ser mayor o igual que el primero. A continuación, se debe mostrar por pantalla una lista con todos los números enteros comprendidos entre los números introducidos, indicando en cada caso si el número escrito es par o impar.

```
Escribe un número entero: 6
Escribe un número entero mayor o igual que 6: 2
¡Te he pedido un número entero mayor que 6!
```

```
Escribe un número entero: 4
Escribe un número entero mayor o igual que 4: 8
El número 4 es par
El número 5 es impar
El número 6 es par
El número 7 es impar
El número 8 es par
```

```
Escribe un número entero: 5
Escribe un número entero mayor o igual que 5: 5
El número 5 es impar
```

6. Escribe un programa que pida dos números enteros, el segundo ha de ser mayor o igual que el primero. A continuación, el programa debe mostrar como resultado la suma de todos los

enteros comprendidos entre el primero y el segundo incluidos ambos. Observa el formato del resultado en el modelo:

```
Escribe un número entero: 7
Escribe un número entero mayor que 7: 7
¡Te he pedido un número entero mayor que 7!
```

```
Escribe un número entero: 30
Escribe un número entero mayor que 30: 32
La suma desde 30 hasta 32 es 93
30 + 31 + 32 = 93
```

7. Escribe un programa que pida por pantalla un número entero y que a continuación calcule su factorial. En número ha de ser mayor que cero.

> El factorial de un entero n (n!) es el producto de los enteros desde el 1 hasta dicho número n.

```
Escribe un número entero mayor que cero: -5
¡Le he pedido un número entero mayor que cero!
```

```
Escribe un número entero mayor que cero: 5
El factorial de 5 es: 120
```

8. Escribe un programa que permita sumar números, la aplicación debe funcionar de la siguiente forma:
   a) En primer lugar, el programa preguntará por la cantidad de números que se van a introducir
   b) A continuación, el programa debe pedir cada uno de esos valores (pueden ser decimales)
   c) Por último el programa calculará la suma de todos ellos y mostrará el resultado por pantalla.

```
¿Cuántos valores vas a introducir? -1
¡Imposible!
```

```
¿Cuántos valores vas a introducir?5
Escribe el número 1: 25
Escribe el número 2: 30
Escribe el número 3: 10.5
Escribe el número 4: 14
Escribe el número 5: 23
La suma de los números que has escrito es 102.5
```

9. Diseña un programa que detecte números negativos, la aplicación debe funcionar de la siguiente forma:
   a) En primer lugar, el programa preguntará por la cantidad de números que se van a introducir.
   b) A continuación, el programa ha de pedir cada uno de esos valores (pueden ser decimales)
   c) Por último el programa indicará cuántos de esos números son negativos.

```
¿Cuántos valores vas a introducir? -1
¡Imposible!
```

```
¿Cuántos valores vas a introducir?2
Escribe el número 1: 56
Escribe el número 2: -22
Has escrito 1 número negativo.
```

```
¿Cuántos valores vas a introducir?5
Escribe el número 1: 56
Escribe el número 2: -22
Escribe el número 3: 98
Escribe el número 4: -30
Escribe el número 5: -30
Has escrito 3 números negativos.
```

10. Diseña un programa que pregunte por la cantidad de números que se van a introducir. A continuación, la aplicación debe de mostrar como resultado el mayor, el menor y la media aritmética de todos ellos.

```
¿Cuántos valores vas a introducir? -1
¡Imposible!
```

```
¿Cuántos valores vas a introducir? 5
Escribe el número 1: 25
Escribe el número 2: 100
Escribe el número 3: 7
Escribe el número 4: 90
Escribe el número 5: 14
El mínimo de los valores introducidos es 7.0
El máximo de los valores introducidos es 100.0
La media de los valores introducidos es 47.2
```

11. Crea un programa que pida un valor entero mayor que cero y calcule todos sus divisores, mostrando el resultado con el formato indicado en el siguiente ejemplo (necesitarás utilizar una variable tipo lista).

```
Escribe un número entero mayor que cero: -5
¡El número introducido debe ser un entero mayor de cero!
```

```
Escribe un número entero mayor que cero: 200
Los 12 divisores de 200 son 1, 2, 4, 5, 8, 10, 20, 25, 40, 50, 100 y 200.
```

12. Tomando como punto de partida el programa anterior, escribe el código necesario para que el programa determine si el número es primo o no, el resultado podría ser:

```
Escribe un número entero mayor que cero: -5
¡El número introducido debe ser un entero mayor de cero!
```

```
Escribe un número entero mayor que cero: 200
200 no es un número primo.
Los 12 divisores de 200 son 1, 2, 4, 5, 8, 10, 20, 25, 40, 50, 100 y 200.
```

```
Escribe un número entero mayor que cero: 7
7 es un número primo.
```

6. Bucle for (2) – Variables independientes

1. Crea un programa que pida dos números enteros. Utilizando caracteres producto (*), el programa ha de dibujar un rectángulo. La longitud de la base viene dada por el valor del primer número y la de la altura por el valor del segundo

```
--- D I B U J O  D E  R E C T A N G U L O S ---
¿Qué dimensiones tiene el rectángulo?
        - Base: 6
        - Altura: 3
*       *       *       *       *       *
*       *       *       *       *       *
*       *       *       *       *       *
```

2. Modifica el programa anterior de tal forma que los caracteres utilizados en el perímetro del rectángulo sean caracteres producto (*) y los interiores sean símbolos +.

```
--- D I B U J O  D E  R E C T A N G U L O S ---
¿Qué dimensiones tiene el rectángulo?
        - Base: 5
        - Altura: 4
*       *       *       *       *
*       +       +       +       *
*       +       +       +       *
*       *       *       *       *
```

3. Modifica el programa anterior de tal forma que los caracteres utilizados en el perímetro del rectángulo sean caracteres producto (*) y el interior esté vacío.

```
--- D I B U J O  D E  R E C T A N G U L O S ---
¿Qué dimensiones tiene el rectángulo?
        - Base: 5
        - Altura: 4
*       *       *       *       *
*                               *
*                               *
*       *       *       *       *
```

4. Modifica el código del ejercicio anterior para conseguir que el programa pregunte al usuario por el número de rectángulos que se desea dibujar. Sigue el modelo:

```
--- D I B U J O  D E  S E R I E S  D E  R E C T A N G U L O S ---
¿Qué dimensiones tiene cada rectángulo?
        - Base: 5
        - Altura: 3
¿Cuántos rectángulos quieres dibujar?         2
*       *       *       *       *
*                               *
*       *       *       *       *

*       *       *       *       *
*                               *
*       *       *       *       *
```

5. Modifica el código anterior. Ahora los rectángulos han de estar conectados entre sí: El lado inferior del rectángulo superior sea también el lado superior del rectángulo inferior.

```
--- D I B U J O  D E  S E R I E S  DE  R E C T A N G U L O S ---
¿Qué dimensiones tiene cada rectángulo?
        - Base: 5
        - Altura: 3
¿Cuántos rectángulos quieres dibujar?        3
*     *     *     *     *
*                       *
*     *     *     *     *
*                       *
*     *     *     *     *
*                       *
*     *     *     *     *
```

6. Compliquemos las cosas un poco más. Partiendo del ejercicio número 4, ahora los rectángulos han de estar a la misma altura y separados entre sí.

```
--- D I B U J O  D E  S E R I E S  DE  R E C T A N G U L O S ---
¿Qué dimensiones tiene cada rectángulo?
        - Base: 5
        - Altura: 4
¿Cuántos rectángulos quieres dibujar?        2
*     *     *     *     *     *     *     *     *     *
*                       *     *                       *
*                       *     *                       *
*     *     *     *     *     *     *     *     *     *
```

7. Modifica el código anterior para que ahora los rectángulos compartan los lados laterales:

```
--- D I B U J O  D E  S E R I E S  DE  R E C T A N G U L O S ---
¿Qué dimensiones tiene cada rectángulo?
        - Base: 5
        - Altura: 4
¿Cuántos rectángulos quieres dibujar?        3
*     *     *     *     *     *     *     *     *     *     *     *     *
*                       *                       *                       *
*                       *                       *                       *
*     *     *     *     *     *     *     *     *     *     *     *     *
```

8. Traca final. Vamos a dibujar una matriz de rectángulos

```
--- D I B U J O  D E  S E R I E S  DE  R E C T A N G U L O S ---
¿Qué dimensiones tiene cada rectángulo?
        - Base: 5
        - Altura: 3
        - Número de columnas de rectángulos:        3
        - Número de filas de rectángulos:           2
*     *     *     *     *     *     *     *     *     *     *     *     *
*                       *                       *                       *
*     *     *     *     *     *     *     *     *     *     *     *     *
*                       *                       *                       *
*     *     *     *     *     *     *     *     *     *     *     *     *
```

7. Bucle for (3) – Variables dependientes

1. Escribe un programa que utilizando el carácter producto (*), dibuje triángulos rectángulos del tipo mostrado en el siguiente ejemplo. El programa necesitará conocer el número de caracteres que forman los catetos.

```
Medida de los catetos: 4
*
*       *
*       *       *
*       *       *       *
```

2. Modifica el código anterior para conseguir que el triángulo sea del tipo:

```
Medida de los catetos: 4
*       *       *       *
*       *       *
*       *
*
```

3. Crea un programa que cree dibujos del tipo mostrado en el ejemplo. El programa ha de preguntar por el número de filas que tenga la estructura:

```
Número de filas: 4
-       -       -       *
-       -       *
-       *
*
```

4. Crea un programa que cree dibujos del tipo mostrado en el ejemplo. El programa ha de preguntar por el número de filas que tenga la estructura:

```
Número de filas: 4
-       -       -       *
-       -       *       *
-       *       *       *
*       *       *       *
```

5. Crea un programa que cree dibujos del tipo mostrado en el ejemplo. El programa ha de preguntar por el número de columnas que tenga la estructura:

```
Número de columnas: 3
*
*       *
*       *       *
*       *
*
```

6. Crea un programa que cree dibujos del tipo mostrado en el ejemplo. El programa ha de preguntar por el número de filas que tenga la estructura:

```
Número de filas: 4
                  *
              *   *   *
          *   *   *   *   *
      *   *   *   *   *   *   *
```

8. Bucle while (1)

1. Escribe un programa que pida dos números enteros. El segundo número ha de ser mayor que el primero. En caso de no serlo, el programa volverá a solicitar el segundo número hasta que se cumpla la condición. Para terminar el programa debe mostrar los dos números por pantalla.

```
Escribe un número entero: 6
Escribe un número entero mayor que 6: 6
6 no es mayor que 6. Vuelve a intentarlo: 1
1 no es mayor que 6. Vuelve a intentarlo: 8

Los números que has introducido son el 6 y el 8.
Programa finalizado
```

2. Escribe un programa que pida un primer número. A partir de entonces el programa continuará pidiendo números hasta que el número introducido sea menor que el número inicial. En ese momento terminará la ejecución del programa.

```
Escribe un número: 4.5
Escribe un número mayor que 4.5: 5
Escribe otro número mayor que 4.5:4.5
Escribe otro número mayor que 4.5: 2

2.0 es menor que 4.5.
Programa finalizado
```

3. Escribe el código de un programa que pida un número entero positivo. A continuación, el programa debe de pedir por pantalla esa cantidad de números positivos. El programa debe continuar pidiendo números positivos hasta que se hayan introducido la cantidad de números indicada en la primera variable. Como resultado final se indicará cuantos de los números introducidos cumplen la condición de ser positivos.

```
¿Cuántos números positivos vas a introducir?: 3

Escribe el primer número: 10
Escribe otro número: 2
Escribe otro número: -5
Escribe otro número: 2

Has escrito 3 números positivos. Programa finalizado.
```

```
¿Cuántos números positivos vas a introducir?: 0
La cantidad debe ser mayor que 0. Inténtalo de nuevo: 1

Escribe el primer número: 1

Has escrito 1 número positivo. Programa finalizado.
```

4. Escribe un programa que pida números positivos. El programa debe finalizar cuando el usuario introduzca un número negativo, en ese momento se debe mostrar la suma de todos los números positivos que se hayan introducido.

```
Escribe un número: 11
Escribe otro número: 3
Escribe otro número: 7
```

```
Escribe otro número: -1

Los números positivos introducidos suman 21.
Programa terminado
```

```
Escribe un número: -9

Los números positivos introducidos suman 0.
Programa terminado
```

5. Escribe un programa que pida por pantalla un valor positivo. A continuación, el programa pedirá números hasta que la suma de todos los números introducidos supere el valor del número inicial, mostrando el resultado de la suma.

```
Escribe la cantidad límite: 13

Escribe un número: 5.2
Escribe otro número: -1
Escribe otro número: 9

Has superado el límite. La suma de los números introducidos es 13.2.
Programa finalizado
```

```
Escribe la cantidad límite: -3
El número debe ser mayor que 0. Inténtalo de nuevo: 0
El número debe ser mayor que 0. Inténtalo de nuevo: 5.5

Escribe un número: 7.2

Has superado el límite. La suma de los números introducidos es 7.2.
Programa finalizado
```

6. Crea un programa que debe comenzar pidiendo dos números enteros (mínimo y máximo). A continuación, el programa debe ir pidiendo números enteros situados entre esos valores máximo y mínimo. El programa concluye cuando escribamos un número no comprendido entre los dos valores iniciales, mostrando la cantidad de números escritos incluidos en el intervalo abierto definido.

```
Escribe un número: 6
Escribe un número mayor que 6: 4
4 no es mayor que 6. Inténtalo de nuevo: 50

Escribe un número entre 6 y 50: 45
Escribe otro número entre 6 y 50: 6
Escribe otro número entre 6 y 50: 4

Ha escrito 2 números entre 6 y 50.
Programa terminado
```

```
Escribe un número: 8
Escribe un número mayor que 8: 20

Escribe un número entre 8 y 20: 45

No has escrito ningún número entre 6 y 50.
Programa terminado
```

```
Escribe un número: -10
Escribe un número mayor que -10: -5
```

```
Escribe un número entre -10 y -5: -8
Escribe un número entre -10 y -5: 100

Ha escrito 1 número entre -10 y -5.
Programa terminado
```

7. Escribe el código que pida números múltiplos de cinco mientras el usuario indique que quiere seguir introduciendo números. Para indicar que queremos introducir un nuevo número, el usuario deberá contestar S a una pregunta.

```
Escribe un número múltiplo de 5: 10
¿Quieres escribir otro número múltiplo de 5? (S/N) S
Escribe otro número múltiplo de 5: 15
¿Quieres escribir otro número múltiplo de 5? (S/N) N
Has escrito 2 números múltiplos de 5.
Programa terminado
```

```
Escribe un número múltiplo de 5: 6
6 no es un número múltiplo de 5. Inténtalo de nuevo: 35
¿Quieres escribir otro número múltiplo de 5? (S/N) x
Has escrito 1 número múltiplo de 5.
Programa terminado
```

8. Realiza la descomposición en factores primos de un número dado por teclado.

```
Escribe un número mayor que 1: 500
Descomposición en factores primos: 2 2 5 5 5
Escribe un número mayor que 1: 521
Descomposición en factores primos: 521
Escribe un número mayor que 1: 1
1 no es un número mayor que 1. Inténtalo de nuevo: 720
Descomposición en factores primos: 2 2 2 2 3 3 5
```

9. Instrucciones break, continue, pass, else

1. Crea un programa que solicite una palabra por teclado. El programa a continuación debe mostrar cada una de las letras de esa palabra en una fila distinta. Si alguna de las letras es la h, el programa se debe detener.

```
Introduce la palabra que quieres deletrear: PIE
P
I
E
```

```
Introduce la palabra que quieres deletrear: Python
P
y
t
```

2. Modifica el programa anterior para que en caso de que el programa deletree la palabra completa, al final escriba de nuevo la palabra deletreada.

```
Introduce la palabra que quieres deletrear: PIE
P
I
E
PIE
```

```
Introduce la palabra que quieres deletrear: Python
P
y
t
```

3. Utilizando un bucle for y un único if sencillo con la condición (numero%3==0), crea un programa que pida un número entero y muestre una lista con todos los números enteros menores o iguales que el introducido que no sean múltiplos de 3.

```
Introduce un número entero: 4
1
2
4
```

4. Repite el ejercicio anterior sustituyendo el bucle for por un bucle while.

5. Crea un programa con el siguiente código:

```
for numero in range(10,20):
    for i in range(2,numero):
        if numero%i == 0:
            j=numero/i
            print (numero,'=',i,' x ',j)
            break
    else:
        print (numero, 'es un número primo')
```

Añade los comentarios necesarios para explicar la función de cada una de las líneas de código

10. Funciones (1)

1. Escribe el siguiente código y añade los comentarios necesarios para explicar su funcionamiento.

```
def subrutina():
    global a
    print(a)
    a += 10
    return

a = 33
subrutina()
print(a)
```

2. ¿Qué ocurre en el programa anterior si eliminamos la segunda línea de código (global a)? Añade al código los comentarios necesarios para explicar este cambio de funcionamiento.

3. Crea un nuevo módulo y escribe en él el siguiente código añadiendo los comentarios necesarios para explicar su funcionamiento.

```
def subrutina():
    global a
    print(a)
    a = 21
    return

subrutina()
a = 20
print(a)
```

4. Añade los comentarios necesarios para explicar el funcionamiento del código:

```
def funcion():
    global a
    a = 10
    print(a)
    return

a = 33
funcion()
print(a)
```

5. Escribe el siguiente código y añade los comentarios necesarios para explicar su funcionamiento (difícil).

```
def subrutina():
    a = b
    print(a)
    return

a = 4
b = 3
subrutina()
print(a)
```

6. Añade los comentarios necesarios para explicar el funcionamiento del código:

```
def subrutina_1():
    a = 20
    print(a)
    return

def subrutina_2():
    global a
    a = 30
    print(a)
    return

a = 10
subrutina_1()
print(a)
subrutina_2()
print(a)
```

7. Añade los comentarios necesarios para explicar el funcionamiento del código:

```
def subrutina():
    def sub_subrutina():
        a = 5
        print(a)
        return

    a = 4
    sub_subrutina()
    print(a)
    return

a = 3
subrutina()
print(a)
```

8. Escribe una función `repite_hola` que reciba como parámetro un número entero `n` y escriba por pantalla el mensaje `"Hola"n`veces.

```
¿Cuántos saludos necesitas? 2
hola
hola
```

9. Escribe una función `repite_saludo` que reciba como parámetro un número entero `n` y una cadena `saludo` y escriba por pantalla el valor de `saludo n` veces.

```
¿Cuántos saludos necesitas?2
¿Qué saludo deseas mostrar? Buenos días
Buenos días
Buenos días
```

10. Define la función print_asegundos(horas, minutos, segundos) con tres parámetros (horas, minutos y segundos) que imprima por pantalla la transformación a segundos de una medida de tiempo expresada en horas, minutos y segundos:

```
¿Horas? 2
¿Minutos? 5
¿Segundos? 7
7507 segundos
```

11. Funciones (2)

En esta sección vamos a utilizar funciones externas al bloque de código principal. Para facilitar la corrección de los ejercicios crearemos en primer lugar un proyecto en el que guardaremos todos los módulos con funciones.

1. Crea un proyecto pyDev nuevo con el nombre repositorio. A partir de ahora guardaremos en la carpeta src de este proyecto cualquier módulo que contenga funciones a utilizar en los ejercicios de este apartado.
2. Crea un programa que solicite tres valores de tiempo con el formato h:m:s. Donde h son horas, m minutos y s segundos. A través de una función se debe convertir cada uno de estos valores a segundos. El programa mostrará como resultado la suma de los tres valores en formato de segundos.
3. Modifica el programa anterior para conseguir que el resultado de la suma de los tres tiempos se muestre en formato de h:m:s

```
Introduce el primer tiempo
horas: 5
minutos: 57
segundos: 3
Introduce el segundo tiempo
horas: 4
minutos: 4
segundos: 15
Introduce el tercer tiempo
horas: 0
minutos: 3
segundos: 59
Tiempo total: 10 h: 5 m: 17 s
```

4. Un cliente nos plantea un problema: Necesita un programa que facture el uso de un teléfono. El usuario informará de:

- La tarifa por segundo que se va a aplicar.
- Número de comunicaciones realizadas.
- Duración de cada comunicación expresada en horas, minutos y segundos.

Como resultado deberemos informar la duración en segundos y coste de cada comunicación.

Intenta utilizar las funciones almacenadas en la carpeta repositorio

```
¿Cuánto céntimos cuesta 1 segundo de comunicación?: 0.2
¿Cuántas llamadas hay que facturar?: 2
¿Cuántas horas?: 0
¿Cuántos minutos?: 1
¿Cuántos segundos?: 10
Duración: 70 segundos. Coste: 14.0 c€.
¿Cuántas horas?: 0
¿Cuántos minutos?: 2
¿Cuántos segundos?: 5
Duración: 125 segundos. Coste: 25.0 c€.
```

5. Vamos a mejorar el programa anterior:
   - El coste de cada llamada se mostrará con el formato xx€, yycen.
   - Al terminar el programa mostrará el tiempo total consumido en formato hh:mm:ss y el coste total.

```
¿Cuánto céntimos cuesta 1 segundo de comunicación?: .2
¿Cuántas llamadas hay que facturar?: 2
¿Cuántas horas?: 0
¿Cuántos minutos?: 5
¿Cuántos segundos?: 23
Duración: 323 segundos. Coste: 64 c€.
¿Cuántas horas?: 0
¿Cuántos minutos?: 59
¿Cuántos segundos?: 1
Duración: 3541 segundos. Coste: 708 c€.
Duración total: 1h 4m 24s
Coste total: 7€,72cen
```

6. Crea un módulo de funciones con el nombre area.py y guárdalo dentro del proyecto repositorio. Define en este módulo las funciones necesarias para calcular la superficie de cualquier cuadrado, rectángulo, trapecio, triángulo, círculo y romboide.
7. Crea un programa que permita calcular superficies de cualquier tipo de cuadrado, rectángulo, trapecio, triángulo, círculo y romboide. El programa mostrará un menú inicial con las distintas opciones, solicitará los valores necesarios, calculará el resultado utilizando la función necesaria definida en el ejercicio anterior y mostrará el resultado por pantalla.

```
Selecciona una opción
a) Superficie de un cuadrado
b) Superficie de un rectángulo
c) Superficie de un trapecio
d) Superficie de un triángulo
e) Superficie de un círculo
f) Superficie de un romboide
?:7
Opción incorrecta. Vuelve a intentarlo?: e
¿Longitud del radio?: 6.23
Superficie: 121.93432330583002
```

Ejercicios obtenidos del libro:

https://librosweb.es/libro/algoritmos_Python/

12. Bucle while(2) – Módulo random

1. Escribe un programa que simule el funcionamiento de un dado, generando un número entero al azar entre 1 y 6. A continuación el programa preguntará si queremos generar un nuevo número aleatorio. Si contestamos 'S' o 's' el programa volverá a ejecutarse y en cualquier otro caso finalizará.

```
GENERADOR DE NÚMEROS ALEATORIOS
3
Para generar un nuevo número pulsa S o s, otra tecla para terminar: s
6
Para generar un nuevo número pulsa S o s, otra tecla para terminar: n
Programa terminado.
```

2. Modifica el programa anterior de tal forma que tras cada lanzamiento del dado el programa también muestre la suma conseguida en todos los lanzamientos.

```
LANZANDO UN DADO
----------------------------
Tirada actual: 4            Total acumulado: 4
Para generar un nuevo número pulsa S o s, otra tecla para terminar: s
Tirada actual: 2            Total acumulado: 6
Para generar un nuevo número pulsa S o s, otra tecla para terminar: S
Tirada actual: 4            Total acumulado: 10
Para generar un nuevo número pulsa S o s, otra tecla para terminar: n
Puntuación final: 10 - Programa terminado
```

3. Modifica el programa anterior para que ahora se muestren dos números al azar (dos jugadores) entre 1 y 6. Al terminar el juego el programa debe declarar ganador al jugador que haya obtenido más puntos.

```
LANZANDO DOS DADOS
-------------------------------
Primer jugador:     Tirada actual: 2     Total acumulado: 2
Segundo jugador:    Tirada actual: 5     Total acumulado: 5
Para generar un nuevo número pulsa S o s, otra tecla para terminar: s
Primer jugador:     Tirada actual: 4     Total acumulado: 6
Segundo jugador:    Tirada actual: 1     Total acumulado: 6
Para generar un nuevo número pulsa S o s, otra tecla para terminar: n
************************************************************
Primer jugador y segundo jugador han empatado a  6 puntos
************************************************************
```

```
LANZANDO DOS DADOS
-------------------------------
Primer jugador:     Tirada actual: 5     Total acumulado: 5
Segundo jugador:    Tirada actual: 5     Total acumulado: 5
Para generar un nuevo número pulsa S o s, otra tecla para terminar: s
Primer jugador:     Tirada actual: 4     Total acumulado: 9
Segundo jugador:    Tirada actual: 5     Total acumulado: 10
Para generar un nuevo número pulsa S o s, otra tecla para terminar: n
************************************************************
```

```
Vencedor: Segundo jugador.     Resultado final:     Jugador1: 9 - Jugador2: 10
```

4. Escribe un programa que simule un juego en el que dos jugadores lanzan su propio dado. Tras cada tirada se mostrará el valor obtenido en el lanzamiento y el total obtenido hasta ese instante por cada jugador. Tras cada tirada se preguntará a cada uno de los jugadores si quieren volver a lanzar el dado o si desean terminar su partida.
5. A partir del instante en que un jugador haya decidido no jugar el programa no debe volver a preguntarle.

```
LANZANDO DOS DADOS II
---------------------------------
Primer jugador:          Tirada actual: 4          Total acumulado: 4
Segundo jugador:         Tirada actual: 4          Total acumulado: 4
Jugador 1: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
Jugador 2: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
Primer jugador:          Tirada actual: 2          Total acumulado: 6
Segundo jugador:         Tirada actual: 3          Total acumulado: 7
Jugador 1: Para lanzar el dado pulsa S o s, otra tecla para terminar: n
Jugador 2: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
Primer jugador:          Tirada actual: 0          Total acumulado: 6
Segundo jugador:         Tirada actual: 2          Total acumulado: 9
Jugador 2: Para lanzar el dado pulsa S o s, otra tecla para terminar: n
************************************************************
Vencedor: Segundo jugador.     Resultado final: Jugador1: 6 – Jugador2: 9
************************************************************
```

6. Modifica el programa anterior para que el programa declare como ganador al jugador que haya obtenido más puntos sin superar los 21 puntos.

   Ten en cuenta que existen muchas situaciones diferentes:

   a) Un jugador gana al obtener más puntos que el otro sin tener más de 21 puntos.

   b) Un jugador gana cuando ha obtenido menos puntos que el otro pero el otro ha obtenido más de 21 puntos.

   c) Los dos jugadores empatan obteniendo una misma puntuación menor de 21.

   d) Los dos jugadores pierden cuando han obtenido más de 21 puntos.

```
LANZANDO DOS DADOS II
---------------------------------
Primer jugador:          Tirada actual: 5          Total acumulado: 5
Segundo jugador:         Tirada actual: 3          Total acumulado: 3
Jugador 1: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
Jugador 2: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
-------------------------------------------------------------------------------------------------
Primer jugador:          Tirada actual: 4          Total acumulado: 9
Segundo jugador:         Tirada actual: 6          Total acumulado: 9
Jugador 1: Para lanzar el dado pulsa S o s, otra tecla para terminar: n
Jugador 2: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
-------------------------------------------------------------------------------------------------
Primer jugador:          Tirada actual: 0          Total acumulado: 9
Segundo jugador:         Tirada actual: 1          Total acumulado: 10
```

```
Jugador 2: Para lanzar el dado pulsa S o s, otra tecla para terminar: n
-------------------------------------------------------------------------------------------------
**************************************************************
Vencedor: Segundo jugador.      Resultado final: Jugador1: 9 - Jugador2: 10
**************************************************************
```

7. Modifica el programa anterior de tal forma que cuando uno o los dos jugadores supere los 21 puntos, el juego de la partida por terminada y muestre el resultado final.

```
…
-------------------------------------------------------------------------------------------------
Primer jugador:          Tirada actual: 5          Total acumulado: 18
Segundo jugador:         Tirada actual: 4          Total acumulado: 20
Jugador 1: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
Jugador 2: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
-------------------------------------------------------------------------------------------------
Primer jugador:          Tirada actual: 6          Total acumulado: 24
Segundo jugador:         Tirada actual: 4          Total acumulado: 24
**************************************************************
Los dos jugadores han perdido por superar los 21 puntos
**************************************************************
```

```
…
-------------------------------------------------------------------------------------------------
Primer jugador:          Tirada actual: 6          Total acumulado: 14
Segundo jugador:         Tirada actual: 2          Total acumulado: 8
Jugador 1: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
Jugador 2: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
-------------------------------------------------------------------------------------------------
Primer jugador:          Tirada actual: 4          Total acumulado: 18
Segundo jugador:         Tirada actual: 2          Total acumulado: 10
Jugador 1: Para lanzar el dado pulsa S o s, otra tecla para terminar: s
Jugador 2: Para lanzar el dado pulsa S o s, otra tecla para terminar: n
-------------------------------------------------------------------------------------------------
Primer jugador:          Tirada actual: 6          Total acumulado: 24
Segundo jugador:         Tirada actual: 0          Total acumulado: 10
**************************************************************
Vencedor: Segundo jugador.      El primer jugador sobrepasó los 21 puntos
**************************************************************
```

13. Módulo random

1. Escribe un programa que muestre tres números al azar entre 1 y 5. A continuación el programa preguntará si queremos generar un nuevo número aleatorio. Si contestamos 'n' el programa terminará, en cualquier otro caso volverá a ejecutarse.

```
---------------------------------
Tirada:        | 1 | 3 | 4 |
---------------------------------
Pulsa n para terminar, otra tecla para volver a jugar: s
---------------------------------
Tirada:        | 2 | 3 | 5 |
---------------------------------
Pulsa n para terminar, otra tecla para volver a jugar: n
Programa terminado
```

2. Modifica el código anterior para que tras cada tirada el programa indique si hemos conseguido ninguno, dos o tres números iguales.

```
---------------------------------
Tirada:        | 3 | 1 | 3 |
---------------------------------
Dos de los tres números son iguales
Pulsa n para terminar, otra tecla para volver a jugar: s
---------------------------------
Tirada:        | 3 | 5 | 2 |
---------------------------------
Los tres números son distintos
Pulsa n para terminar, otra tecla para volver a jugar: s
---------------------------------
Tirada:        | 1 | 1 | 1 |
---------------------------------
Los tres números son iguales
Pulsa n para terminar, otra tecla para volver a jugar: n
Programa terminado
```

3. Crea un programa que simule el funcionamiento de una máquina tragaperras. Debes conseguir:
   a) Inicialmente, el jugador indica cuánto dinero quiere jugar.
   b) El programa muestra tres números al azar del 1 al 5.
   c) Si los tres números son distintos, el jugador pierde todo su dinero y la partida termina.
   d) Si salen dos números iguales, el jugador duplica la apuesta.
   e) Si salen tres números iguales, el jugador multiplica por 5 la apuesta.
   f) El jugador indica si quiere seguir jugando o no.
   g) Al terminar la partida, el programa indica si se ha ganado o perdido.

```
¿Qué cantidad quieres apostar (€)?5
---------------------
```

```
Tirada:         | 2 | 2 | 2 |
---------------------
Los tres números son iguales.  Has multiplicado por cinco tu dinero.
¡Enhorabuena! Has ganado 20.0 €.  Ahora tienes 25.0 €.
Pulsa n para terminar, otra tecla para volver a jugar: s
---------------------
Tirada:         | 2 | 2 | 1 |
---------------------
Dos de los tres números son iguales.  Has duplicado tu dinero.
¡Enhorabuena! Has ganado 25.0 €.  Ahora tienes 50.0 €.
Pulsa n para terminar, otra tecla para volver a jugar: n
Enhorabuena has ganado 45.0 €.
```

```
¿Qué cantidad quieres apostar (€)?10
--------------------------------
Tirada:         | 5 | 4 | 3 |
--------------------------------
Los tres números son distintos.  Has perdido todo tu dinero.
Programa terminado.
```

4. Vamos a modificar la máquina anterior:
   a) Inicialmente, el jugador indica cuánto dinero quiere jugar.
   b) El programa muestra tres números al azar del 1 al 5.
   c) Si los tres números son distintos, el jugador pierde todo su dinero y la partida termina.
   d) Si sale un 5, se recupera la apuesta.
   e) Si salen dos 5, el jugador multiplica por cuatro la apuesta.
   f) Si salen tres 5, el jugador multiplica por diez la apuesta.
   g) Si salen dos números iguales que no sean 5, el jugador duplica la apuesta.
   h) Si salen un 5 y dos números iguales que no sean 5, el jugador multiplica por 3 la apuesta.
   i) Si salen tres números iguales que no sean cinco, el jugador multiplica por 5 la apuesta.
   j) El jugador indica si quiere seguir jugando o no.
   k) Al terminar la partida, el programa indica si se ha ganado o perdido.

```
¿Qué cantidad quieres apostar (€)? 10
--------------------------------
Tirada:         | 5 | 5 | 2 |
--------------------------------
Has conseguido dos cincos.  Has multiplicado por cuatro tu dinero.
¡Enhorabuena! Has ganado 40.0 €.  Ahora tienes 50.0 €.
Pulsa n para terminar, otra tecla para volver a jugar: s
--------------------------------
Tirada:         | 4 | 5 | 3 |
--------------------------------
Has conseguido un cinco.  Recuperas tu dinero.
¡Enhorabuena! Has ganado 0.0 €.  Ahora tienes 50.0 €.
```

```
Pulsa n para terminar, otra tecla para volver a jugar: s
--------------------------------
Tirada:         | 5 | 3 | 3 |
--------------------------------
Has conseguido un cinco y dos números iguales.  Has multiplicado por tres tu dinero.
¡Enhorabuena! Has ganado 100.0 €.  Ahora tienes 150.0 €.
Pulsa n para terminar, otra tecla para volver a jugar: s
--------------------------------
Tirada:         | 1 | 3 | 1 |
--------------------------------
Dos de los tres números son iguales.  Has duplicado tu dinero.
¡Enhorabuena! Has ganado 150.0 €.  Ahora tienes 300.0 €.
Pulsa n para terminar, otra tecla para volver a jugar: n
Enhorabuena has ganado 290.0 €.
```

14. Errores y excepciones

1. Localiza el error en el siguiente bloque de código. Crea una excepción para evitar que el programa se bloquee y que explique al usuario la causa y/o solución:

```
resultado = 10/0
```

```
No es posible dividir entre cero, debes introducir un número distinto.
```

2. Partimos de la siguiente lista:

```
lista=['gato', 'perro', 'ratón', 'pato', 'elefante']
```

Crea un programa que solicite el índice de un elemento de la lista y que a continuación muestre dicho elemento por pantalla.

```
['gato', 'perro', 'ratón', 'pato', 'elefante']
Introduce el índice del elemento de la lista que quieres mostrar por pantalla:3
pato
```

En caso de que el índice sea un número entero demasiado mayor (>4) se producirá un error. Crea una excepción para evitar que el programa se bloquee y que explique al usuario la causa y solución.

```
['gato', 'perro', 'ratón', 'pato', 'elefante']
Introduce el índice del elemento de la lista que quieres mostrar por pantalla:5
El índice se encuentra fuera del rango.
Debes utilizar un número mayor o igual que cero y menor que la longitud de la lista.
```

3. (Necesario haber visto diccionarios). Partimos del siguiente diccionario:

```
colores = { 'rojo':'red', 'verde':'green', 'negro':'black' }
```

Crea un programa que solicite un término del diccionario para mostrar su valor asociado por pantalla:

```
Introduce el nombre de un color en español: azul
azul en inglés se dice blue
```

En caso de que el término no esté incluido en el diccionario se producirá un error. Crea una excepción para evitar que el programa se bloquee y que explique al usuario la causa y solución.

```
Introduce el nombre de un color en español: amarillo
Error: El término amarillo no se encuentra en este diccionario, debes probar con otro que sí exista.
```

4. Crea un programa que solicite dos números enteros por pantalla y muestre el resultado de su suma. En caso de que uno de los valores introducidos sea una cadena de caracteres el programa debe gestionar el error informando de la causa y solución.

```
Introduce el primer valor: 5
Introduce el segundo valor: 4
5 + 4 = 9
```

```
Introduce el primer valor: 3
Introduce el segundo valor: 2.5
El valor introducido no es un número entero
```

5. Partiremos de la siguiente lista:

```
elementos = [1, 'pie', -2]
```

Realiza una función llamada agregar_una_vez(lista, elemento) que reciba como parámetros una lista y un elemento. La función debe añadir el elemento al final de la lista con la condición de no repetir ningún elemento. Además, si este elemento ya se encuentra en la lista se debe invocar un error de tipo ValueError que debes capturar y mostrar este mensaje en su lugar: Error: Imposible añadir elementos duplicados => [elemento].

Cuando tengas la función crea un programa que permita añadir elementos a la lista de forma indefinida. Intenta añadir los siguientes valores a la lista: 10, "pie", "Hola" y luego muestra su contenido.

Sugerencia: Puedes utilizar la sintaxis "elemento in lista", si el elemento está en la lista genera un valor True, si no lo está genera un valor False

```
La lista actual es:  [1, 'pie', -2]
Valor que quieres añadir a la lista: 10
La lista actual es:  [1, 'pie', -2, '10']
Valor que quieres añadir a la lista: pie
Error: Imposible añadir elementos duplicados => pie
La lista actual es:  [1, 'pie', -2, '10']
Valor que quieres añadir a la lista: Hola
La lista actual es:  [1, 'pie', -2, '10', 'Hola']
```

6. Escribe el código de una función que ha de ser capaz de aceptar números enteros positivos. La función ha de chequear que el número introducido cumple la condición dada. La función ha de ser capaz de gestionar los valores que no sean enteros o que sean enteros negativos o si se introduce un texto o si pulsamos "enter" sin haber introducido un valor, informando del error.
7. Mejora el programa anterior consiguiendo que en caso de excepción la función solicite un nuevo valor hasta que este tenga el formato correcto.

```
Introduce un número entero positivo: -5
El valor es número entero negativo
Introduce un número entero positivo: 5.2
El número es decimal
Introduce un número entero positivo: hola
El valor introducido no es un número
Introduce un número entero positivo: 4
4
```

8. Ejecuta el programa que desarrollaste en el ejercicio número 4 de la hoja 13. Módulo random e introduce como cantidad de dinero a apostar un valor negativo ¿qué ocurre? Prueba ahora a introducir una cadena de texto ¿algún problema? Gestiona estas dos excepciones de tal forma que el programa funcione de forma correcta. Es decir, cuando introduzcamos un valor no apropiado, el programa en lugar de interrumpirse debe

indicarnos cual es el error y solicitar un valor adecuado. Este proceso se debe repetir hasta que el usuario utilice el programa de forma correcta.

15. Diccionarios

Bloque de ejercicios progresivos. Es obligatorio incluir comentarios en todos ellos.

1. El 11 de julio de 2010 la selección española de fútbol gano su primer y único mundial. La alineación titular fue la siguiente:

1	POR	Iker Casillas	Capitán
15	DEF	Sergio Ramos	
3	DEF	Gerard Piqué	
5	DEF	Carles Puyol	
11	DEF	Joan Capdevila	
14	MED	Xabi Alonso	
16	MED	Sergio Busquets	
8	MED	Xavi Hernández	
18	MED	Pedro Rodríguez	
6	MED	Andrés Iniesta	
7	DEL	David Villa	

   Crea un diccionario al que llamaremos titulares. Los elementos de dicho diccionario serán los once jugadores que comenzaron el partido, añadidos en el orden mostrado en la tabla anterior, siendo el número de su dorsal el índice y su nombre el valor asociado.

2. Utilizando la función get() muestra de forma tabulada y ordenados por su número dorsal de menor a mayor los nombres de los once jugadores (el menor dorsal posible es el 1 y nunca podrá tener más de dos cifras.

3. Modifica el ejercicio anterior para que programa lea el número de elementos incluidos en el diccionario titulares, mostrando ese valor mediante la expresión 'Iniciaron el partido XX jugadores', siendo XX el número de elementos incluidos en el diccionario.

4. Añade al ejercicio anterior el código necesario mostrar una lista con todos los índices utilizados en la biblioteca y otra lista con todos los valores almacenados en ella (utiliza los métodos Name.keys() y Name.values()).

5. Partimos ahora del ejercicio 1. Añade el código necesario para que el programa cree una copia de la biblioteca "titulares" y asígnale el nombre plantilla. Muestra por pantalla el contenido de plantilla con el mismo formato que el indicado en el ejercicio 2.

6. Añade al programa un nuevo diccionario al que llamaremos suplentes. Los elementos de dicho diccionario serán los once jugadores suplentes el día del partido, siendo el número de su dorsal el índice y su nombre el valor asociado. Muestra como resultado el contenido de los dos diccionarios siguiendo el mismo formato que en los ejercicios 2 y 5.

7. Añade los elementos del diccionario suplentes al diccionario plantilla y muestra el contenido actualizado del diccionario plantilla siguiendo el formato de los ejercicios anteriores.

8. Durante el partido se produjeron tres sustituciones. Xabi Alonso, Pedrito y Villa abandonaron el campo. Navas, Fàbregas y Torres se incorporaron. Haz una copia del diccionario "titulares" y asígnale el nombre "final". Utilizando los métodos setdefault(key,valor) y pop(key), elimina del diccionario final a los tres jugadores que

fueron sustituidos y añade a los que se incorporaron. Muestra al final del ejercicio el contenido del diccionario final con el formato de los ejercicios anteriores.

9. Ahora tú eres el seleccionador y vas a hacer los cambios. Partimos de los diccionarios titulares y suplentes. El programa mostrará las dos listas con el formato habitual. A continuación te preguntará por el número del primer jugador que quieres sustituir. Tras contestar, el programa preguntará por el número del jugador que quieres que entre en su lugar. A continuación se mostrarán los contenidos actualizados de los dos diccionarios. Ten en cuenta que podemos hacer un máximo de tres cambios.
10. Un jugador que ha sido sustituido no puede volver a entrar al campo.

```
Jugando en el campo
----------------------------
1       -> Casillas
3       -> Pique
5       -> Puyol
6       -> Iniesta
7       -> Villa
8       -> Xavi Hernández
11      -> Capdevila
14      -> Xavi Alonso
15      -> Ramos
16      -> Busquets
18      -> Pedrito

Suplentes
-------------
4       -> Marchena
9       -> Torres
10      -> Fàbregas
12      -> Valdés
13      -> Mata
17      -> Arbeloa
19      -> Llorente
20      -> Javi Martínez
21      -> Silva
22      -> Navas
23      -> Reina
Introduce el número del jugador que quieres sustituir: 5
Jugador a sustituir: Puyol                  número: 5
Introduce el número del jugador que va a entrar: 17
Jugador sustituido: Puyol                   número: 5
Jugador que entra en su lugar: Arbeloa      número: 17

Jugando en el campo
----------------------------
1       -> Casillas
3       -> Pique
6       -> Iniesta
7       -> Villa
8       -> Xavi Hernández
```

```
11      -> Capdevila
14      -> Xavi Alonso
15      -> Ramos
16      -> Busquets
17      -> Arbeloa
18      -> Pedrito

Suplentes
-------------
4       -> Marchena
5       -> S-Puyol
9       -> Torres
10      -> Fàbregas
12      -> Valdés
13      -> Mata
19      -> Llorente
20      -> Javi Martínez
21      -> Silva
22      -> Navas
23      -> Reina
Introduce el número del jugador que quieres sustituir: 1
Jugador a sustituir: Casillas                 número: 1
Introduce el número del jugador que va a entrar: 12
Jugador sustituido: Casillas                  número: 1
Jugador que entra en su lugar: Valdés         número: 12

Jugando en el campo
---------------------------
3       -> Pique
6       -> Iniesta
7       -> Villa
8       -> Xavi Hernández
11      -> Capdevila
12      -> Valdés
14      -> Xavi Alonso
15      -> Ramos
16      -> Busquets
17      -> Arbeloa
18      -> Pedrito

Suplentes
-------------
1       -> S-Casillas
4       -> Marchena
5       -> S-Puyol
9       -> Torres
10      -> Fàbregas
13      -> Mata
19      -> Llorente
20      -> Javi Martínez
21      -> Silva
```

```
22      -> Navas
23      -> Reina
Introduce el número del jugador que quieres sustituir:12
Jugador a sustituir: Valdés                 número: 12
Introduce el número del jugador que va a entrar: 1
Casillas  no puede volver al campo
Introduce el número del jugador que quieres sustituir: 14
Jugador a sustituir: Xavi Alonso            número: 14
Introduce el número del jugador que va a entrar:21
Jugador sustituido: Xavi Alonso             número: 14
Jugador que entra en su lugar: Silva     número: 21

Jugando en el campo
---------------------------
3       -> Pique
6       -> Iniesta
7       -> Villa
8       -> Xavi Hernández
11      -> Capdevila
12      -> Valdés
15      -> Ramos
16      -> Busquets
17      -> Arbeloa
18      -> Pedrito
21      -> Silva

Suplentes
-------------
1       -> S-Casillas
4       -> Marchena
5       -> S-Puyol
9       -> Torres
10      -> Fàbregas
13      -> Mata
14      -> S-Xavi Alonso
19      -> Llorente
20      -> Javi Martínez
22      -> Navas
23      -> Reina
Ya has realizado las tres sustituciones
```

16. Ficheros

1. Crea un módulo en Python que incluya una función diseñada por ti. Esta función debe devolver la ruta absoluta del directorio donde se encuentra el módulo que realiza la llamada.

   Aunque al principio pueda parecer un poco confuso, esta función será muy útil en los próximos ejercicios, ya que te permitirá definir fácilmente la ubicación de los ficheros con los que trabajarás.

2. Crea un archivo con el nombre 1.txt y guárdalo en el mismo directorio en que vayas a guardar este ejercicio. El contenido del archivo ha de ser:

   I, I will be king
   And you, you will be queen
   Though nothing, will drive them away
   We can beat them, just for one day
   We can be heroes, just for one day

3. Crea un módulo que pregunte por el título de la canción contenida en el archivo 1.txt del ejercicio anterior y por el nombre de su compositor.

   A continuación, el programa debe crear una nueva versión del archivo inicial a la que asignaremos por nombre el título de la canción y como extensión .txt.

   En esta nueva versión, la primera línea de texto ha de informarnos del título de la canción seguido del nombre del autor. A continuación, se mostrará la letra de la canción.

   ```
   Título de la canción: xxxxxx
   Nombre del autor: yyyyyyyyyyy
   xxxxxx - yyyyyyyyyyy
   I, I will be king
   And you, you will be queen
   Though nothing, will drive them away
   We can beat them, just for one day
   We can be heroes, just for one day
   ```

4. Crea ahora un módulo que lea el contenido del archivo 1.txt y muestre por pantalla los caracteres entre el 10 y el 17 en una línea. A continuación se deben mostrar esos mismos caracteres en columna y mayúsculas (utilizar el método upper()).

5. Crea una copia del fichero 1.txt y asignale el nombre 4.txt. Utiliza los métodos tell and seek para **sustituir los caracteres** comprendidos en este nuevo fichero entre las posiciones 10 y 17 por su traducción al español (no puedes escribir la línea completa, solamente sustituir la expresión indicada).

6. Crea un módulo que lea de una en una cada una de las líneas del archivo 1.txt. ,pregunte por su traducción, vaya escribiendo la letra en español en un archivo que llamaremos traduc.txt y vaya pasando a las líneas siguientes hasta terminar.

7. Para determinar la precisión de un arma de fuego se ha realizado es siguiente experimento:

   Quinientos tiradores han disparado el arma una única vez, apuntando al centro de una diana. Las coordenadas de dicho centro son (4,4). En el fichero coordenadas.txt se han

recogido las coordenadas de los quinientos disparos en el formato (x.xx,y.yy) (11 dígitos), siendo x.xx las coordenadas en el eje de las x e y.yy las coordenadas en el eje de las y. Ambas distancias son positivas y su valor máximo ha de ser menor de 10.

Abre el archivo coordenadas.txt y muestra por pantalla un listado en la que aparezcan las coordenadas de cada disparo en la forma "X = xx.x – Y = yy.y".

```
X = 1.67 - Y = 0.88
X = 2.69 - Y = 3.35
X = 3.88 - Y = 7.95
…
```

8. Partiremos de nuevo del fichero coordenadas.txt. Vamos a ir leyendo las coordenadas de cada uno de los disparos. Tras leer cada una de las coordenadas la pantalla ha de mostrar el valor de las coordenadas medias acumuladas incluyendo todos los disparos realizados hasta ese instante. ¿Qué puedes decir del funcionamiento de esta arma?

```
Disparo 1 :   x= 4.81 y= 1.55     xmed= 4.810 ymed= 1.550
Disparo 2 :   x= 7.40 y= 7.27     xmed= 6.105 ymed= 4.410
Disparo 3 :   x= 5.89 y= 1.82     xmed= 6.033 ymed= 3.547
…….
```

```
# Truco para dar formato a los números decimales.
#Formateamos 5.1234554321
print "{0:.2f}".format(5.1234554321)
#El resultado sería
5.12
#¿Y si queremos tres?
print "{0:.3f}".format(5.1234554321)
#El resultado sería
5.123
```

9. Modifica el programa anterior para que tras cada disparo también se muestre la distancia media a la diana:

Distancia entre dos puntos: $d(a,b)=\sqrt{(a_x-b_x)^2+(a_y-b_y)^2}$

```
Disparo 1 :   x= 4.81 y= 1.55     xmed= 4.810 ymed= 1.550     Error: 2.580
Disparo 2 :   x= 7.40 y= 7.27     xmed= 6.105 ymed= 4.410     Error: 2.145
Disparo 3 :   x= 5.89 y= 1.82     xmed= 6.033 ymed= 3.547     Error: 2.083
…
```

10. Crea un módulo que pregunte a un alumno por la calificación obtenida en cada una de las siguientes materias: Matemáticas, Lengua, Historia y TIC II. Las calificaciones serán números enteros. El resultado se debe ir guardando en una lista de nombre "notas". Cada elemento de la lista será a su vez una lista con dos componentes, el primero será el nombre de la materia y el segundo la calificación. Muestra el contenido de la lista "notas" y la nota media de todas las materias.

11. Modifica el programa anterior. Utilizando el módulo pickle crea el archivo notas.pkl. Este archivo ha de guardar la información relativa al nombre de las asignaturas y sus calificaciones.

12. Utilizando el módulo pickle abre el archivo notas.pkl. Lee la información contenida en el mismo y muestra por pantalla de forma tabulada la información contenida en él.

```
Matemáticas ->    10
Lengua      ->    9
Historia    ->    5
TIC II      ->    7
Calificación media obtenida: 7.75
```

13. En este ejercicio vamos a crear un encriptador de mensajes. La clave para crear la encriptación va a ser sumar un valor fijo a cada uno de los caracteres ASCII del texto a codificar. El programa debe pedir un texto que introduciremos a través de un input. A continuación solicita la clave de encriptación que será el número entero que hay sumar la código ASCII de cada carácter del texto a encriptar. Por último mostraremos por pantalla el texto encriptado.

chr(num):	Devuelve el carácter asociado al valor "num" en código ASCII.
ord('x'):	Devuelve el código ASCII del carácter 'x'.

```
Introduce el texto a encriptar: En un lugar de la marcha de cuyo nombre no me quiero acordar
Introduce la clave de encriptación: 5
Js%zs%qzlfw%ij%qf%rfwhmf%ij%hz~t%strgwj%st%rj%vznjwt%fhtwifw
```

14. Modifica el programa anterior. Utilizando el módulo pickle el programa ha de guardar el mensaje encriptado. Sólo quien conozca la clave de encriptación podrá leer el mensaje.

15. Crea un programa que conociendo la clave de encriptación lea cualquier archivo creado utilizando el programa del ejercicio anterior.

```
Texto a desencriptar
Hshu'[|ypun'klzjpmyú'sh'tèx|puh'Lupnth5
Introduce la clave de desencriptación: 7
Texto desencriptado:
Alan Turing descifró la máquina Enigma.
```

16. Crea un programa que utilizando el módulo shelve gestione una base de datos con los alumnos de un centro. Se debe almacenar su código (código de matrícula), nombre, dirección y curso. Como clave usarás el código del alumno. Almacena diez estudiantes en el almacén.

17. Crea un programa que recorra el fichero anterior y muestre los nombres de los alumnos, ordenados por curso / nombre.

18. Tomando como punto de partida el ejercicio 4 de la hoja 12 (Módulo random) modifica el código del programa para que utilizando el módulo shelve se guarden en un archivo independiente los datos de las tres mejores puntuaciones conseguidas en el juego, así como el nombre de los jugadores. Esta información se debe mostrar al comienzo de la partida y al finalizar la misma, ya actualizada, si el jugador ha conseguido entrar en la lista.

17. POO – Creando clases y objetos sencillos

1. Crea un programa. En el programa crea una clase, a la que llamarás "libro", que defina las características de los libros de tu biblioteca personal. La clase debe de tener ***dos propiedades:***

   - propietario: El valor de esta propiedad ha de ser tu nombre completo.
   - read: Esta propiedad tomará inicialmente el valor lógico False e indicará si un libro ha sido ya leído o todavía no.

Crea un objeto tipo "libro" y muestra por pantalla el valor de sus dos propiedades.

```
Andrés Sánchez
False
```

2. Partiendo del programa anterior añade a la clase "libro" ***dos métodos***:

   - El primer método **informará** de si un libro ha sido ya leído o todavía no. Para ello ha de mostrar uno de los dos mensajes siguientes:
     - Si la propiedad read tiene el valor False: "Todavía no has leído este libro".
     - Si la propiedad read tiene el valor True: "Ya has leído este libro".
   - El segundo método **realizará la acción** de cambiar el valor de la propiedad read de False a True.

A continuación, el programa realizará las siguientes opciones:

- Crea un objeto "libro".
- Muestra por pantalla el estado de las propiedades propietario y read al iniciar la ejecución de programa.
- Ejecuta el método que informa sobre si hemos leído el libro o no.
- Ejecuta el método que cambia el valor de la propiedad read a True
- Muestra de nuevo las propiedades el objeto que acabas de crear.
- Ejecuta el método que informa sobre si hemos leído el libro o no.

```
Andrés Sánchez
False
Todavía no has leído este libro
Andrés Sánchez
True
Ya has leído este libro
```

3. Crea un programa. En el programa crea una clase, a la que llamarás "triangulo". La clase debe de tener ***cuatro propiedades:***

   - tipo: tomará el valor "Triángulo"

- lados: asígnale el valor 3
- base: asígnale el valor 0
- altura: asígnale el valor 0

Crea un objeto tipo "triangulo" y muestra por pantalla el valor de sus cuatro propiedades.

```
3
Triángulo
0
0
```

4. Partiendo del programa anterior crea un segundo objeto triángulo y muestra las propiedades de los dos objetos por pantalla. Enriquece un poco el formato de las órdenes print para mostrar de manera más clara la información.

```
-------- Propiedades del primer triángulo--------
Número de lados: 3
Tipo de polígono: Triángulo
Base: 0
Altura: 0

-------- Propiedades del segundo triángulo--------
Número de lados: 3
Tipo de polígono: Triángulo
Base: 0
Altura: 0
```

5. Partimos de nuevo del ejercicio anterior. Una vez que has mostrado los datos de los dos triángulos modifica los valores de la base y la altura del primer triángulo (no en la clase triángulo, debes cambiarlos en el objeto). Los nuevos valores para el primer objeto serán:

- base= 5
- altura=4

Muestra de nuevo las propiedades de los dos objetos.

```
-------- Propiedades del primer triángulo--------
Número de lados: 3
Tipo de polígono: Triángulo
Base: 0
Altura: 0

-------- Propiedades del segundo triángulo--------
Número de lados: 3
Tipo de polígono: Triángulo
Base: 0
Altura: 0

********** DATOS ACTUALIZADOS **********
-------- Propiedades del primer triángulo--------
```

```
Número de lados: 3
Tipo de polígono: Triángulo
Base: 5
Altura: 4

-------- Propiedades del segundo triángulo--------
Número de lados: 3
Tipo de polígono: Triángulo
Base: 0
Altura: 0
```

6. Crea un nuevo programa. Define en él de nuevo la clase "triángulo" con las mismas propiedades que en los ejercicios anteriores:

   - -tipo: tomará el valor "Triángulo"
   - lados: asígnale el valor 3
   - base: asígnale el valor 0
   - altura: asígnale el valor 0

Crea un objeto triangulo y asígnale el valor 5 a la longitud de la base y 4 a la longitud de la altura.

Crea un segundo objeto triángulo. El programa debe preguntar al usuario por la longitud de la base y de la altura de este segundo triángulo. Establece estos valores como estado de las propiedades base y altura del segundo triángulo.

Muestra las propiedades de los dos triángulos por pantalla.

```
---- Datos segundo triángulo ----
Introduce la longitud de la base: 5.4
Introduce la longitud de la altura: 3.27

-------- Propiedades del primer triángulo--------
Número de lados: 3
Tipo de polígono: Triángulo
Base: 5
Altura: 4

-------- Propiedades del segundo triángulo--------
Número de lados: 3
Tipo de polígono: Triángulo
Base: 5.4
Altura: 3.27
```

7. Partimos del ejercicio anterior. Vamos añadir un procedimiento a la clase triangulo. Este procedimiento ha de calcular la superficie del triángulo. Ten en cuenta que tendrás que utilizar las propiedades base y altura de cada objeto para calcular dentro del procedimiento el valor de la superficie.

Consejo: Para referirte a una propiedad del objeto dentro del procedimiento tendrás que utilizar el parámetro self tal y como se explica en el punto 29.1.2 de teoría. Por ejemplo para referirte a la base tendrás que utilizar la sintaxis self.base.

Una vez creado el procedimiento que calcula la superficie de los objetos de la clase triangulo. Añade el código necesario para que se muestre la superficie de los dos triángulos. Este valor se tiene que obtener llamando al procedimiento creado en la primera parte del ejercicio.

```
---- Datos segundo triángulo ----
Introduce la longitud de la base: 5.4
Introduce la longitud de la altura: 3.27

-------- Propiedades del primer triángulo--------
Base: 5
Altura: 4
El area es: 10.00

-------- Propiedades del segundo triángulo--------
Base: 5.4
Altura: 3.27
El area es: 8.83
```

8. Modifica el código anterior para que el programa nos informe de cual de los dos triángulos tiene una superficie mayor:

```
---- Datos segundo triángulo ----
Introduce la longitud de la base: 5
Introduce la longitud de la altura: 8

-------- Propiedades del primer triángulo--------
Base: 5
Altura: 4
El área del primer triángulo es: 10.00

-------- Propiedades del segundo triángulo--------
Base: 5.0
Altura: 8.0
El área del segundo triángulo es: 20.00

El segundo triángulo tiene una superficie mayor que el primero.
```

```
---- Datos segundo triángulo ----
Introduce la longitud de la base: 2
Introduce la longitud de la altura: 10

-------- Propiedades del primer triángulo--------
Base: 5
Altura: 4
El área del primer triángulo es: 10.00

-------- Propiedades del segundo triángulo--------
```

```
Base: 2.0
Altura: 10.0
El área del segundo triángulo es: 10.00

Los dos triángulos tienen la misma superficie.
```

18. POO – Constructores y encapsulado de variables

1. Crea un programa que conste de una clase llamada Alumno. La clase tendrá dos propiedades, el nombre y la nota del alumno. Crearemos tres métodos para esta clase:

   - Un método que inicialice las propiedades del objeto. Este método debe recibir como parámetros el nombre y la nota del objeto.
   - Un método que imprima por pantalla los valores de las propiedades
   - Un método que muestre si el alumno ha aprobado o no.

   Crea un objeto, asígnale nombre y nota. A continuación, ejecuta el método que muestra sus datos y el método que nos informa sobre si ha aprobado o suspendido.

> A la hora de crear un objeto de esta clase debes incluir como parámetros (entre paréntesis) los valores de las propiedades del mismo en la llamada.

2. Modifica el programa anterior de tal manera que se cree un segundo objeto. En este caso las propiedades del segundo alumno se pedirán a través de teclado. Muestra por pantalla la información de los dos alumnos e indica si han aprobado o han suspendido.

```
Nombre del segundo alumno: Luisa
Nota del segundo alumno: 6.7

- Primer alumno
  ------------
Nombre: José
Nota: 7
El alumno ha aprobado

- Segundo alumno
----------------
Nombre: Luisa
Nota: 6.7
El alumno ha aprobado
```

3. Modifica el programa anterior de tal manera que una vez creado el segundo objeto alumno el programa muestre el nombre de la persona que tenga mayor nota.

```
Nombre del segundo alumno: Marta
Nota del segundo alumno: 8.3
Marta ha obtenido una nota más alta que José.
```

4. Elabora un programa que tenga una clase a la que llamaremos Persona. La clase tendrá como propiedades el nombre y la edad de una persona. Diseña los siguientes métodos:

   - Método para inicializar las propiedades del objeto.
   - Método para mostrar los datos del objeto.
   - Método que muestre por pantalla si la persona es mayor de edad o no.

Añade el código necesario para crear un objeto de la clase Persona, el programa ha de solicitar sus propiedades por pantalla y a continuación ejecutar los métodos que muestran las propiedades del objeto y que indique si el alumno es o no es mayor de edad.

```
Nombre de la persona: Luisa
Edad de Luisa: 17

Luisa
17
Luisa todavía no es mayor de edad
```

En el ejercicio anterior el método que analiza si la persona es o no es mayor de edad, imprime una cadena de texto. Muchas veces nos va a interesar no que el método escriba algo, si no que devuelva un valor a través de un return. Vamos a aplicar esta variación en el siguiente ejercicio.

5. Modifica el ejercicio anterior para que el método que estudia si la persona es mayor de edad o no lo es, devuelva un valor True si la persona es mayor de edad y False si no lo es. El método no debe escribir nada por pantalla. Crea un nuevo objeto Persona, las propiedades de este objeto se definirán a través del teclado. El programa preguntará a continuación al usuario si quiere jugar a un determinado juego. Si el usuario contesta que no, el programa terminará. Si el usuario contesta que Si el programa del debe dar la bienvenida o decirle que no puede jugar en función de que el usuario sea mayor de edad o no lo sea. Para esto último debes utilizar el método que determina si la persona es mayor de edad o si no lo es.

```
Nombre de la persona: Marta
Edad de Marta: 31
¿Quieres jugar una partida de 'Python: The Game' (S/N)?a
¿Quieres jugar una partida de 'Python: The Game' (S/N)?S
Bienvenido
```

```
Nombre de la persona: Luisa
Edad de Luisa: 17
¿Quieres jugar una partida de 'Python: The Game' (S/N)?S
No tienes edad suficiente para jugar a 'Python: The Game'.
```

6. Elabora un programa y en él define la clase Triangulo. La clase tendrá cuatro propiedades:

- lados: Número de lados.
- lado1: Longitud de uno de los lados
- lado2: Longitud de otro de los lados
- lado3: Longitud del tercer lado

Una de estas propiedades ha se estar encapsulada ya que no puede tomar otro valor más que el definido en el constructor. ¿Cuál de ellas es? Tenlo en cuenta para al definir el constructor.

Define ahora los siguientes métodos:

- Método que muestra por pantalla las cuatro propiedades del objeto.
- Método que muestre por pantalla la longitud del lado más largo.
- Método que muestra por pantalla el tipo de triángulo del que se trata (equilátero, isósceles o escaleno).

Crea un objeto triángulo definiendo la longitud de sus valores a través de ordenes input. A continuación, ejecuta los métodos que muestran por pantalla la longitud del lado más largo y el tipo de triángulo con el que estamos trabajando. Por último, muestra por pantalla el valor de la propiedad "lados" del objeto.

```
Longitud del primer lado:5
Longitud del segundo lado:6
Longitud del tercer lado:7
Número de lados: 3
Lado 1: 5.0
Lado 2: 6.0
Lado 3: 7.0
El lado más largo mide:7.0
Triángulo escaleno
```

7. Vamos a comprobar que sentido tiene haber encapsulado la propiedad "lados" en la clase anterior. Partiendo del código anterior, añade al final del mismo una línea de código que asigne un valor cualquiera a propiedad lado3.

```
miTriangulo.lado3=33333333
```

Vuelve a mostrar ahora las propiedades del objeto, ¿qué ha sucedido?

```
Longitud del primer lado:5
Longitud del segundo lado:6
Longitud del tercer lado:7
Número de lados: 3
Lado 1: 5.0
Lado 2: 6.0
Lado 3: 7.0
El lado más largo mide:7.0
Triángulo escaleno
Número de lados: 3
Lado 1: 5.0
Lado 2: 6.0
Lado 3: 33333333
```

Como puedes ver, hemos modificado el valor de la propiedad lado3 desde fuera del código de la clase. Intenta modificar ahora de la misma forma el valor de la propiedad número de lados y muestra el resultado por pantalla otra vez ¿Ocurre lo mismo?

8. **Desafío voluntario:** Una regla básica de geometría establece que: Un triángulo no se podrá construir si la suma de la longitud de dos cualquiera de sus lados es menor o igual que la longitud del tercero. Modifica el método de construcción de los triángulos en los ejercicios anteriores para que en caso de que los lados no cumplan la condición anterior no se cree el objeto, sino que se genere un mensaje de error. Al final y como en los ejercicios anteriores se ha de mostrar por pantalla la información del objeto ejecutando los métodos correspondientes.

```
Longitud del primer lado:4
Longitud del segundo lado:5
Longitud del tercer lado:6
Número de lados: 3
Lado 1: 4.0
Lado 2: 5.0
Lado 3: 6.0
El lado más largo mide:6.0
Triángulo escaleno
```

Para hacer las cosas más interesantes, mira lo que ocurre cuando el triángulo no se puede construir:

```
Longitud del primer lado:4
Longitud del segundo lado:5
Longitud del tercer lado:9
No es posible construir el objeto
Traceback (most recent call last):
  File "D:\Google Drive\eclipse-workspace\17. POO Constructores y encapsulado de vaiables\src\constructores_08.py", line 48, in <module>
    trian.propiedades()
  File "D:\Google Drive\eclipse-workspace\17. POO Constructores y encapsulado de vaiables\src\constructores_08.py", line 14, in propiedades
    print("Número de lados:", self.__lados)
AttributeError: 'Triangulo' object has no attribute '_Triangulo__lados'
```

Como las últimas líneas de código intentan trabajar con un objeto que no se ha creado de forma correcta (no se cumplía la condición geométrica), se muestra un error.

Gestiona esta excepción de forma adecuada de tal manera que no se genere el error.

```
Longitud del primer lado:5
Longitud del segundo lado:4
Longitud del tercer lado:10
No es posible construir el objeto
```

19. Encapsulación de variables II

9. Elabora un programa que defina una clase llamada Circulo(). La clase ha de tener tres métodos:

- Un constructor para elaborar los objetos de esta clase. El constructor debe utilizar como parámetro el valor del radio.
- Un método que devuelva el área del círculo.
- Un método que devuelva el perímetro del circulo.

Crea un objeto tipo Circulo y muestra por pantalla su área y su perímetro.

```
Radio del círculo:6
El círculo tiene una superficie: 18.85
El círculo tiene un perímetro: 37.70
```

10. Escribe un programa que contenga una clase a la que llamaremos Rectangulo(). Los objetos de esta clase quedarán definidos por tres propiedades:

- lados: Número de lados.
- base: Longitud de la base del rectángulo.
- altura: Longitud de la altura del rectángulo.
- Ninguna de estas variables debe estar encapsulada.

Define los siguientes métodos:

- Método constructor que crea instancias (objetos) Rectangulo recibiendo como parámetros los valores de la base y la altura.
- Método que muestra por pantalla las tres propiedades del objeto.
- Método que calcule el área del rectángulo.
- Método que calcule el perímetro del rectángulo.

Crea un objeto rectángulo definiendo la longitud de sus valores a través de ordenes input.

Muestra el valor de las propiedades número de lados, longitud de la base y longitud de la altura del objeto ejecutando el método que hemos definido para ello.

Muestra el valor del área y el perímetro del rectángulo ejecutando los métodos correspondientes.

```
Introduce el valor de la longitud de la base: 5
Introduce el valor de la longitud de la altura: 6
Número de lados: 4
Longitud de la base: 5.0
Longitud de la altura: 6.0
El rectángulo tiene una superficie de: 30.0
El rectángulo tiene un perímetro de: 22.0
```

11. Vamos a repasar algo la gestión de errores. Modifica el programa anterior para que las ordenes input solamente puedan aceptar valores numéricos reales mayores de cero. Si se introduce un valor negativo o una cadena de texto el programa continuará pidiendo el valor hasta que se introduzca con el formato correcto.

```
Introduce el valor de la longitud de la base: -5
La longitud ha de ser un valor numérico mayor que cero
Introduce el valor de la longitud de la base: 5
Introduce el valor de la longitud de la altura: siete
La longitud ha de ser un valor numérico mayor que cero
Introduce el valor de la longitud de la altura: 7

Número de lados: 4
Longitud de la base: 5.0
Longitud de la altura: 7.0
El rectángulo tiene una superficie de: 35.0
El rectángulo tiene un perímetro de: 24.0
```

12. Modifica el programa anterior de tal forma que el programa te pregunte por el número de lados que tiene el rectángulo. Cambia el valor de la propiedad número de lados de cuatro a tres. Ejecuta ahora el método que muestra la información del rectángulo. ¿Tiene sentido el resultado? ¿Tiene sentido que el usuario pueda cambiar el valor de esa propiedad? ¿Qué solución se te ocurre para este problema?

```
Introduce el valor de la longitud de la base: 5
Introduce el valor de la longitud de la altura: 4
Introduce el número de lados de tu rectángulo:2
El número de lados debe ser un entero mayor de 2
Introduce el número de lados de tu rectángulo:3.5
El número de lados debe ser un entero mayor de 2
Introduce el número de lados de tu rectángulo:3

Número de lados: 3
Longitud de la base: 5.0
Longitud de la altura: 4.0
```

Hemos podido cambiar esa propiedad a través de una orden del tipo rectan.lados=3. Sin embargo, no tiene sentido que esta orden se pueda programar desde fuera de la clase que define a los objetos rectángulo.

13. La solución al problema que nos hemos encontrado en el ejercicio anterior es encapsular la variable número de lados. Modifica el programa anterior definiendo el número de lados como una variable encapsulada (__lados). Modifica de forma parecida a como lo has hecho en el ejercicio anterior el valor de la propiedad __lados. El programa no dará ningún error, pero observa lo que aparece como resultado al mostrar las propiedades del rectángulo.

```
Introduce el valor de la longitud de la base: 5
Introduce el valor de la longitud de la altura: 4
Introduce el número de lados de tu rectángulo:3

Número de lados: 4
Longitud de la base: 5.0
Longitud de la altura: 4.0
```

```
El rectángulo tiene una superficie de: 20.0
El rectángulo tiene un perímetro de: 18.0
```

14. En cinemática se trabaja habitualmente con tres vectores: posición, velocidad y aceleración. Crea un programa con una clase a la que llamaremos Vector(). Cada objeto vector queda definido por tres propiedades:

- tipo: Esta propiedad solamente podrá tomar uno de los tres siguientes valores:
  - p (para indicar que es un vector de posición)
  - v (si es un vector que indica velocidad)
  - a (si es un vector aceleración)
- x: Valor tipo float que indica el valor de la coordenada x del vector.
- y: Valor tipo float que indica el valor de la coordenada y del vector.

Un vector nunca podrá cambiar de tipo una vez que se definido el mismo. Por lo tanto, esa propiedad deberá estar encapsulada.

Crea un objeto Vector() siguiendo las normas anteriores. Y muestra sus propiedades:

```
Tipo de vector (p,v,a):p
Coordenada x:2.3
Coordenada y:-7.13
tipo de vector: p
x= 2.3
y= -7.13
```

15. Partiendo del programa anterior intenta modificar las propiedades “tipo” y “coordenada x” de un objeto Vector() utilizando ordenes sencillas similares a miVector.tipo=”p”, miVector.x=33. Observa el resultado. ¿Qué conclusiones obtienes?

```
Tipo de vector (p,v,a):p
Coordenada x:3
Coordenada y:2
tipo de vector: p
x= 3.0
y= 2.0
Nuevo tipo para el vector:a
Nuevo valor para la coordenada x:5
tipo de vector: p
x= 5.0
y= 2.0
```

El valor de x ha cambiado sin problemas, el del tipo no. No lo ha hecho porque es una propiedad encapsulada y solamente será posible cambiar su valor a través de un método que esté definido dentro de la clase.

16. Modifica el programa anterior para que se pueda modificar el valor de la propiedad tipo de un vector a través de un método de la clase. Este método utilizará como parámetro el valor del nuevo tipo del vector.

```
Tipo de vector (p,v,a):p
```

```
Coordenada x:3
Coordenada y:2
tipo de vector: p
x= 3.0
y= 2.0
Nuevo tipo para el vector:a
Nuevo valor para la coordenada x:5
tipo de vector: a
x= 5.0
y=2.0
```

17. Partiendo de la clase definida en el ejercicio anterior. Crea un método que calcule el módulo del vector y luego muestra su valor por pantalla.

```
Introduce las coordenadas del primer vector:
Tipo de vector (p,v,a):p
Coordenada x:3
Coordenada y:2
Módulo del vector: 3.61
```

18. Crea un programa que permita sumar vectores. Debes crear el vector suma, asignarle por ejemplo el tipo "a" y mostrar sus valores por pantalla.

```
Introduce las coordenadas del primer vector:
Tipo de vector (p,v,a):p
Coordenada x:3
Coordenada y:6
Introduce las coordenadas del segundo vector:
Tipo de vector (p,v,a):p
Coordenada x:1
Coordenada y:3
El resultado de la suma es:
tipo de vector: a
x= 4.0
y= 9.0
```

19. **Desafío:** Modifica el programa anterior de tal forma que cuando los tipos de dos vectores sean diferentes y los intentemos sumar, el código nos informe de la imposibilidad de realizar la suma. A continuación, nos tienen que ofrecer la opción de cambiar el tipo de uno de los vectores y realizar la suma.

- Otros aspectos a tener en cuenta:

  - El tipo solo puede ser "p", "v" o "a". En caso de solicitar un tipo diferente el programa debe de informar de ello y volver a solicitar el tipo.
  - Las coordenadas han de ser valores tipo float, en caso de introducir una cadena de texto el programa debe informar de ello y volver a solicitar el valor.

La dificultad de este ejercicio es conseguir leer, de alguna manera, el valor del tipo de vector desde fuera del código de definición de la clase.

Resultado cuando los vectores son del mismo tipo:

```
Introduce las coordenadas del primer vector:
Tipo de vector (p,v,a):p
Coordenada x:3
Coordenada y:2.5
Introduce las coordenadas del segundo vector:
Tipo de vector (p,v,a):p
Coordenada x:-2.5
Coordenada y:1
El resultado de la suma es:
tipo de vector: a
x= 0.5
y= 3.5
```

Si los vectores son de distinto tipo:

```
Introduce las coordenadas del primer vector:
Tipo de vector (p,v,a):v
Coordenada x:12
Coordenada y:10
Introduce las coordenadas del segundo vector:
Tipo de vector (p,v,a):a
Coordenada x:-5
Coordenada y:-1
El primer vector es de tipo v y el segundo es de tipo a.
No es posible sumar dos vectores de dos tipos diferentes.
¿Quieres cambiar el tipo del primer vector a tipo a? (S)S
El resultado de la suma es:
tipo de vector: a
x= 7.0
y= 9.0
```

20. Encapsulado de métodos

Todos los ejercicios de esta hoja están relacionados. El objetivo final es que comprendas las ventajas que nos puede dar encapsular un método en determinadas situaciones.

1. Crea un programa con un clase a la que llamarás Usuario(). Los objetos de esta clase tendrán dos propiedades (nombre y password). Ambas propiedades se asignarán a los objetos de la clase a través de un constructor.

   Crea un objeto Usuario y muestra sus propiedades por pantalla.

```
Introduce el nombre del usuario:José
Introduce la contraseña del usuario:Hola
Datos del usuario:
José
Hola
```

2. Piensa un poco. En el ejercicio anterior no tiene sentido que a través de un simple print(usuario.password) podamos leer el valor de la propiedad contraseña. Encapsula la variable password para que no se pueda leer directamente su valor.

```
Introduce el nombre del usuario:José
Introduce la contraseña del usuario:Hola
```

```
Datos del usuario:
José
Traceback (most recent call last):
  File "D:\Google Drive\eclipse-workspace\19. POO Encapsulado de métodos\src\en_met_02.py", line 14, in <module>
    print(usuario1.__password)
AttributeError: 'Usuario' object has no attribute '__password'
```

3. Vamos a seguir modificando el programa anterior. Añade un procedimiento que permita mostrar las propiedades nombre y password (aunque password sea una variable encapsulada).

```
Introduce el nombre del usuario: José
Introduce la contraseña del usuario: Hola
Las propiedades del objeto José son:
Nombre de usuario: José
contraseña del usuario: Hola
```

4. En este caso queremos que se pueda cambiar la contraseña del usuario. Como la contraseña es una variable que hemos definido como encapsulada, no podremos cambiar directamente su valor. Habrá que hacerlo a través de un procedimiento que tome como parámetro el valor de la nueva contraseña.

```
Introduce el nombre del usuario: José
Introduce la contraseña del usuario: Hola

Las propiedades del objeto José son:
==============================
Nombre de usuario: José
contraseña del usuario: Hola
Introduce la nueva contraseña del objeto José: Adiós

Las propiedades del objeto José son:
==============================
Nombre de usuario: José
contraseña del usuario: Adiós
```

5. Vamos a partir de la clase definida en el ejercicio anterior. El programa al arrancar debe preguntar por los datos de un usuario (nombre y contraseña). A continuación, se mostrará un menú con cinco opciones:

   a. Mostrar el nombre del usuario.
   b. Mostrar el nombre y la contraseña del usuario.
   c. Cambiar el nombre del usuario.
   d. Cambiar la contraseña del usuario.
   e. Salir del programa.

Una vez realizada la acción indicada por una de las opciones “a, b, c, d” del menú, el programa debe mostrar de nuevo el menú inicial. El proceso continuará hasta que introduzcamos una opción diferente a “a, b, c, d”.

```
Introduce el nombre del usuario: José
Introduce la contraseña del usuario: Hola
 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:a
El nombre del usuario es: José

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:b

Las propiedades del objeto José son:
==================================
Nombre de usuario: José
contraseña del usuario: Hola

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:c
Introduce el nuevo nobmre del objeto José: Andrés

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:b

Las propiedades del objeto Andrés son:
====================================
Nombre de usuario: Andrés
contraseña del usuario: Hola

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
```

```
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:d
_Introduce la nueva contraseña del objeto Andrés: Adiós

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:b

Las propiedades del objeto Andrés son:
====================================
Nombre de usuario: Andrés
contraseña del usuario: Adiós

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:e
_Programa terminado
```

6. El programa anterior tiene una limitación importante. Cualquier usuario del programa puede acceder a las opciones b, c y d sin saber cual es la contraseña original del usuario. Crea un único método dentro de la clase que ejecute una tarea de comprobación. El método ha de funcionar de la siguiente forma:

- Ha de recibir un parámetro.
- Comparará el valor de ese parámetro con la contraseña actual.
- Si ambos valores son iguales, el método escribirá un mensaje tipo "Contraseña válida. Acceso concedido" y devolverá un valor True.
- Si ambos valores no son iguales, el método escribirá un mensaje tipo "Contraseña incorrecta" y de volverá un valor False.

El control de usuario se realizará de la siguiente forma:

- Si el usuario selecciona la opción a) o e). Se ejecutará la acción sin solicitar la contraseña.
- Si el usuario selecciona cualquiera de las otras opciones, el método asociado a esa acción comprobará que el usuario conoce la contraseña actual. En caso de que la conozca (recibiendo un valor True, desde el método de comprobación) se realizará la acción solicitada. En caso contrario (recibiendo un valor False desde el método de comprobación), no se realizará ninguna acción y se volverá al menú inicial.

```
Introduce el nombre del usuario: José
Introduce la contraseña del usuario: Hola

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:a
_El nombre del usuario es: José

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:b
_Introduce el valor de la contraseña actual:No la se
Contraseña incorrecta

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:b
_Introduce el valor de la contraseña actual:Hola
Contraseña válida.  Acceso concedido

Las propiedades del objeto José son:
==================================
Nombre de usuario: José
contraseña del usuario: Hola

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:c
_Introduce el nuevo nombre del objeto José: José Luis
Introduce el valor de la contraseña actual:No lo se
Contraseña incorrecta

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
```

```
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:c
_Introduce el nuevo nombre del objeto José: José Luis
Introduce el valor de la contraseña actual:Hola
Contraseña válida.  Acceso concedido

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:b
_Introduce el valor de la contraseña actual:Hola
Contraseña válida.  Acceso concedido

Las propiedades del objeto José Luis son:
========================================
Nombre de usuario: José Luis
contraseña del usuario: Hola

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:d
_Introduce la nueva contraseña del objeto José Luis: Adiós
Introduce el valor de la contraseña actual:No la se
Contraseña incorrecta

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:d
_Introduce la nueva contraseña del objeto José Luis: Adiós
Introduce el valor de la contraseña actual:Hola
Contraseña válida.  Acceso concedido

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
```

```
e) Salir del programa
Selecciona una opción:b
_Introduce el valor de la contraseña actual:Adiós
Contraseña válida.  Acceso concedido

Las propiedades del objeto José Luis son:
=========================================
Nombre de usuario: José Luis
contraseña del usuario: Adiós

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Salir del programa
Selecciona una opción:e
_Programa terminado
```

7. Añade una opción más al menú anterior. Esta opción debe de ejecutar el método que comprueba si el usuario conoce la contraseña actual. Por lo tanto, deberá devolver por pantalla el mensaje generado por el método de comprobación de contraseña ("Contraseña válida. Acceso concedido" o "Contraseña incorrecta")

```
Introduce el nombre del usuario: José
Introduce la contraseña del usuario: Hola

_
 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Ejecutar el módulo de control de contraseña
f) Salir del programa
Selecciona una opción:e
_Introduce el valor de la contraseña actual:Hola
Contraseña válida.  Acceso concedido

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Ejecutar el módulo de control de contraseña
f) Salir del programa
Selecciona una opción:e
_Introduce el valor de la contraseña actual:No la se
Contraseña incorrecta
```

```
 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Ejecutar el módulo de control de contraseña
f) Salir del programa
Selecciona una opción:
```

8. En el ejercicio anterior, hemos conseguido ejecutar el método que comprueba si no conocemos la contraseña del usuario llamándolo desde fuera del código de la clase. Parece que no tiene mucho sentido eso sea así.

Es un método que genera una información necesaria para que el programa funcione de una forma u otra, pero en sí no tiene ningún sentido que muestre por pantalla un mensaje de "Contraseña válida. Acceso concedido" Cuando no queremos entrar a ningún sitio.

Por lo tanto, hemos de realizar un cambio importante en el código de la aplicación para conseguir que no se pueda acceder al método de control de contraseña desde fuera de la clase. ¿Cómo puedes conseguirlo?

```
Introduce el nombre del usuario: José
Introduce la contraseña del usuario: Hola

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Ejecutar el módulo de control de contraseña
f) Salir del programa
Selecciona una opción:d
_Introduce la nueva contraseña del objeto José: Adiós
Introduce el valor de la contraseña actual:Hola
Contraseña válida.  Acceso concedido

 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Ejecutar el módulo de control de contraseña
f) Salir del programa
Selecciona una opción:b
_Introduce el valor de la contraseña actual:Adiós
Contraseña válida.  Acceso concedido

Las propiedades del objeto José son:
==================================
Nombre de usuario: José
contraseña del usuario: Adiós
```

```
 - Menú de configuración de usuario:
  ================================
a) Mostrar el nombre del usuario
b) Mostrar el nombre y la contraseña del usuario
c) Cambiar en nombre del usuario
d) Cambiar la contraseña del usuario
e) Ejecutar el módulo de control de contraseña
f) Salir del programa
Selecciona una opción:e
_Traceback (most recent call last):
      File "D:\Google Drive\eclipse-workspace\19. POO Encapsulado de métodos\src\en_met_08.py", line 68, in <module>
   usuario1.passwordCandidate()
AttributeError: 'Usuario' object has no attribute 'passwordCandidate'
```

21. Herencia

1. Se desea crear un videojuego con los 3 tipos de personajes, perros, gatos y ratones. Todos comparten 2 propiedades que serán la vida (número de 1 a 10) y la fuerza (número de 1 a 5). Crea una clase a la que llamaremos Animal, la clase ha de tener las siguientes características:

   - Propiedades:
   - vida: Todos lo objetos tendrán una vida inicial de 10 puntos.
   - fuerza: Esta propiedad quedará definida a través de un parámetro cuando se cree un objeto de la clase Animal.
   - Métodos:
   - propiedades: Debe mostrar por pantalla un texto en el que se indique el valor de las propiedades vida y fuerza de un objeto de la clase.

   Crea un objeto de la clase Animal, asígnale un valor 5 a su propiedad fuerza y muestra sus propiedades por pantalla utilizando el método propiedades.

```
Propiedades del Animal:
Vida: 10
Fuerza: 5
```

2. La propiedad fuerza no podrá tomar en ningún caso un valor mayor de 5. Modifica el programa anterior de tal forma que el valor de la propiedad fuerza se solicite a través de un input. A continuación, modifica el constructor de los objetos Animal de tal forma que el objeto solamente se cree cuando el valor introducido para la propiedad fuerza sea un número menor o igual que cinco. Este control se tiene que realizar no a través del input si no a través del método __init__ del constructor.

```
Introduce el valor de 'fuerza' del objeto a crear:4
Propiedades del Animal:
Vida: 10
Fuerza: 4.0
```

```
Introduce el valor de 'fuerza' del objeto a crear:6
Traceback (most recent call last):
  File "D:\Google Drive\eclipse-workspace\20. Herencia\src\herencia_02.py", line 13, in <module>
    perro.propiedades()
  File "D:\Google Drive\eclipse-workspace\20. Herencia\src\herencia_02.py", line 9, in propiedades
    print("Vida:",self.vida,"\nFuerza:",self.fuerza)
AttributeError: 'Animal' object has no attribute 'vida'
```

3. Vamos a crear ahora tres tipos de personajes para nuestro juego. Partiremos del ejercicio anterior pero antes de nada elimina todo el código que no pertenezca a la definición de la clase Animal.

   Cada uno de estos tipos quedará definido por una nueva clase. Todas las nuevas clases de personajes heredarán las propiedades fuerza y vida de la superclase Animal. Cada una de las nuevas clases tendrá unas propiedades particulares:

- Clase Perros:
  - Propiedad: size (número 3)
- Clase Gatos:
  - Propiedad: hambre (número 2)
- Clase Ratones:
  - Propiedad: color (“blanco”)

Crea un objeto de la clase Perros y muestra sus propiedades por pantalla sus propiedades utilizando el método propiedades.

Ten en cuenta que vas a utilizar el constructor de la superclase Animal para crear el objeto Perros, por lo tanto, deberás pasar como parámetro el valor de la propiedad fuerza.

```
Introduce el valor de 'fuerza' del objeto a crear:5
Propiedades del Animal:
Vida: 10
Fuerza: 5.0
```

4. Observa que el programa anterior, aunque funciona correctamente, no tiene mucho sentido ya que no estamos mostrando la propiedad tamaño del objeto de clase Perros. Crea dentro de la clase Perros un método que sobrescriba el método propiedades de la superclase y muestre las propiedades del objeto de la clase perros (puede sonar un poco extraño, se explica como hacerlo en el punto “Sobrescribir un método heredado” de tus apuntes).

```
Introduce el valor de 'fuerza' del objeto a crear:5
Propiedades del Perro:
Vida: 10
Fuerza: 5.0
Tamaño: 3
```

5. Añade un objeto tipo Gatos y otro tipo Ratones. Muestra sus propiedades por pantalla.

```
Introduce el valor de 'fuerza' del primer objeto a crear:4
Propiedades del Perro:
Vida: 10
Fuerza: 4.0
Tamaño: 3
Introduce el valor de 'fuerza' del segundo objeto a crear:3
Propiedades del Gato:
Vida: 10
Fuerza: 3.0
Hambre: 2
Introduce el valor de 'fuerza' del tercer objeto a crear:1
Propiedades del Ratón:
Vida: 10
Fuerza: 1.0
Color: blanco
```

6. Vamos a complicar un poco las cosas modificando como creamos los objetos tipo Perros. Hasta ahora para crear un objeto de la clase Perros utilizamos una orden del tipo miperro=Perros(valor) donde valor es un número que indica el valor de la fuerza del objeto creado. Sin embargo, el objeto Perros tiene otra propiedad, size, a la que siempre le asignamos un valor 3.

Modifica el código del ejercicio anterior para que al crear un objeto Perros, utilicemos dos parámetros, el primero será la fuerza y el segundo el tamaño (size). El truco será utilizar un constructor para los objetos tipo Perros que llame al constructor de los objetos tipo Animal. Intenta resolver esta situación.

```
Introduce el valor de 'fuerza' del objeto a crear:5
Introduce el valor del 'tamaño' del objeto a crear:1
Propiedades del Perro:
Vida: 10
Fuerza: 5.0
Tamaño: 1.0
```

Para poder realizar este ejercicio debes entender un bloque de teoría que no se encuentra en los apuntes y que explicaremos a continuación.

7. Toma el código del ejercicio anterior e intenta imprimir por pantalla el nombre del objeto que has creado:

```
fuerza=float(input("Introduce el valor de 'fuerza' del objeto a crear:"))
size=float(input("Introduce el valor del 'tamaño' del objeto a crear:"))
perro=Perros(fuerza,size)
perro.propiedades()
print(perro)
```

Mostrará como resultado:

```
Introduce el valor de 'fuerza' del objeto a crear:5
Introduce el valor del 'tamaño' del objeto a crear:2
Propiedades del Perro:
Vida: 10
Fuerza: 5.0
Tamaño: 2.0
<__main__.Perros object at 0x000001C1A54D7580>
```

Generando un mensaje referido al objeto, pero no una información relevante.

Python permite crear un método especial de clase con el nombre __str__ al que podemos añadir el código que nos interese. Lo que hace especial a este método, es que se ejecutará cuando invoquemos el nombre el objeto al que hace referencia.

8. Modifica el ejercicio anterior para que cuando escribamos la orden print(perro) se muestre un mensaje en el que se indique el valor de la vida, fueras y tamaño del objeto en una única línea.

```
fuerza=float(input("Introduce el valor de 'fuerza' del objeto a crear:"))
size=float(input("Introduce el valor del 'tamaño' del objeto a crear:"))
perro=Perros(fuerza,size)
perro.propiedades()
print(perro)
```

Muestre:

```
Introduce el valor de 'fuerza' del objeto a crear:4
Introduce el valor del 'tamaño' del objeto a crear:3
Propiedades del Perro:
Vida: 10
Fuerza: 4.0
Tamaño: 3.0
Este perro tiene 4.0 puntos de fuerza, 10 puntos de vida y un tamaño de 3.0
```

Realizar un programa que incorpore la clase Calculadora. La clase debe de incorporar 4 métodos para realizar las 4 operaciones básicas (sumar, restar, multiplicar y dividir) con dos números.

```
# creamos la clase
class Calculadora:
    def __init__(self,valor1,valor2):
        self.valor1=valor1
        self.valor2=valor2
        print("Los números son ",valor1," ",valor2)
    def sumar(self):
        return self.valor1+self.valor2
    def restar(self):
        return self.valor1-self.valor2
    def multiplicar(self):
        return self.valor1*self.valor2
    def dividir(self):
        return self.valor1/self.valor2
```

```
# bloque principal
x=int(input("Número 1 "))
y=int(input("Número 2 "))
calcular=Calculadora(x,y)
print(calcular.sumar())
print(calcular.restar())
print(calcular.multiplicar())
print(calcular.dividir())
```

1. El tipo range()

1. Escribe un programa que solicite un número de tipo entero. A continuación, debe escribir la lista de números enteros consecutivos desde el 0 al valor seleccionado.

```
Escribe un número entero: 9
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
Escribe un número entero: -5
[0, -1, -2, -3, -4, -5]
```

```
Escribe un número entero: 0
[0]
```

2. Escribe un programa que pida un número entero mayor que cero y Escribe varias listas de números consecutivos, como indican los ejemplos siguientes:

```
Escribe un número entero mayor que 0: 0
¡Le he pedido un número entero mayor que 0!
```

```
Escribe un número entero mayor que 0: 5
```

```
[0, 1, 2, 3, 4, 5]
[5, 4, 3, 2, 1, 0]
[1, 2, 3, 4]
[4, 3, 2, 1]
[0, 1, 2, 3, 4, 5, 4, 3, 2, 1, 0]
```

```
Escribe un número entero mayor que 0: 1
[0, 1]
[1, 0]
[]
[]
[0, 1, 0]
```

3. Escribe un programa que pida dos números enteros (el segundo mayor que el primero) y Escribe varias listas de números consecutivos, como indican los siguientes ejemplos:

```
Escribe un número entero: 8
Escribe otro número entero mayor que 8: 5
¡Te he pedido un número mayor que 8!
```

```
Escribe un número entero: 5
Escribe otro número entero mayor que 5: 11
[5, 6, 7, 8, 9, 10, 11]
[10, 9, 8, 7, 6, 5]
[6, 7, 8, 9, 10, 11, 12]
[10, 9, 8, 7, 6]
[5, 6, 7, 8, 9, 10, 11, 10, 9, 8, 7, 6, 5]
```

```
Escribe un número entero: -2
Escribe otro número entero mayor que -2: 3
[-2, -1, 0, 1, 2, 3]
[2, 1, 0, -1, -2]
[-1, 0, 1, 2, 3, 4]
[2, 1, 0, -1]
[-2, -1, 0, 1, 2, 3, 2, 1, 0, -1, -2]
```

4. Escribe un programa que pida dos números enteros y Escribe la lista de números consecutivos de uno a otro, en orden creciente o decreciente.

```
Escribe el número entero inicial: 3
Escribe el número entero final: 12
[3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
```

```
Escribe el número entero inicial: 16
Escribe el número entero final: 5
[16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5]
```

```
Escribe el número entero inicial: 5
Escribe el número entero final: 5
[5]
```

5. Escribe un programa que pida dos números enteros y Escribe la lista de números consecutivos que hay entre ellos, de menor a mayor.

```
Escribe un número entero: 4
Escribe otro número entero: 10
[5, 6, 7, 8, 9]
```

```
Escribe un número entero: 10
Escribe otro número entero: 4
[5, 6, 7, 8, 9]
```

```
Escribe un número entero: 4
Escribe otro número entero: -3
[-2, -1, 0, 1, 2, 3]
```

```
Escribe un número entero: 5
Escribe otro número entero: 6
[]
```

```
Escribe un número entero: 5
Escribe otro número entero: 5
[]
```

6. Escribe un programa que pida dos números enteros m y n y Escribe una lista de n números consecutivos a partir de m.

```
Escribe el número entero inicial: 4
Escribe cuántos valores quiere: -5
¡La cantidad de valores no puede ser negativa!
```

```
Escribe el número entero inicial: 4
Escribe cuántos valores quiere: 10
[4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
```

```
Escribe el número entero inicial: 4
Escribe cuántos valores quiere: 0
[]
```

7. Escribe un programa que pida dos números enteros y Escribe la lista de números pares que hay entre ellos (incluidos ellos mismos si son pares).

```
Escribe el número entero inicial: 6
Escribe el número entero final: 4
¡El número final debe ser mayor que el inicial!
```

```
Escribe el número entero inicial: 3
Escribe el número entero final: 12
[4, 6, 8, 10, 12]
```

```
Escribe el número entero inicial: 6
Escribe el número entero final: 15
[6, 8, 10, 12, 14]
```

```
Escribe el número entero inicial: 8
Escribe el número entero final: 8
[8]
```

8. Escribe un programa que pida tres números enteros y Escribe la lista de números múltiplos del tercero que hay entre los dos primeros (incluidos ellos mismos si son múltiplos del número indicado).

```
Escribe el número entero inicial: 5
Escribe el número entero final: 4
¡El número final debe ser mayor que el inicial!
```

```
Escribe el número entero inicial: 9
Escribe el número entero final: 37
¿De qué número quiere los múltiplos?: 0
¡Los múltiplos deben ser de un número entero mayor que cero!
```

```
Escribe el número entero inicial: 9
Escribe el número entero final: 37
¿De qué número quiere los múltiplos?: 5
Entre 9 y 37 hay 6 múltiplos de 5:
[10, 15, 20, 25, 30, 35]
```

```
Escribe el número entero inicial: 9
Escribe el número entero final: 18
¿De qué número quiere los múltiplos?: 3
Entre 9 y 18 hay 4 múltiplos de 3 :
[9, 12, 15, 18]
```

```
Escribe el número entero inicial: 7
Escribe el número entero final: 10
¿De qué número quiere los múltiplos?: 25
Entre 7 y 10 hay 0 múltiplos de 25 :
[]
```