PYTHON EN 150 EJERCICIOS

E001

Crear una variable con el nombre 'miVariable', cargar el valor "Hola" y mostrarlo por pantalla.

E002

Crear la variable 'x' con el valor 1, crear la variable 'y' con el valor 2, crear la variable 'z' con la suma de la variable 'x' e 'y'. Mostrar el valor de la variable 'z' por pantalla

E003

Crear la variable 'cadena' con el valor "Javi". Mostrar por pantalla el texto "Hola" y el contenido de la variable 'cadena'

E004

Crear la variable 'saludo' con el valor "Hola". Crear la variable 'cadena' con el valor "Javi". Mostrar por pantalla la concatenación de la variable 'saludo' con la variable 'cadena'

E005

Pedir por pantalla un nombre con el texto "¿Cuál es tu nombre?" y a continuación mostrar por pantalla "Hola" y el nombre introducido

E006

Pide un nombre por pantalla, a continuación, pide un apellido por pantalla. Muestra a continuación el apellido, una coma y el nombre, todo en la misma línea.

E007

Pide primero un número por pantalla, a continuación, pide otro número y muestra la suma de ambos. Pista: tendrás que utilizar un operador cast

E008

Pide primero un número por pantalla, a continuación, pide otro número y muestra los siguientes resultados utilizado estos datos: la suma, la resta, la multiplicación, la división, la división entera, el resto de la división (el módulo), y el exponente

E009

Pedir por pantalla el alto y el ancho de un rectángulo. Mostrar el área y el perímetro del mismo.

E010

Solicitar por pantalla un número y decir si es par o impar

E011

Solicitar por pantalla una edad y decir si es mayor de edad o no

E012

Pedir un número por pantalla y decir si está entre el 5 y el 10 (ambos inclusive). Utilizar el operador 'and'

E013

Pedir un número por pantalla y decir si está entre el 5 y el 10 (ambos inclusive). NO utilizar el operador 'and'

E014

Pedir al usuario dos números y pintar por pantalla cuál es el mayor

E015

Un usuario tiene que proporcionar la siguiente información:

- Nombre de libro
- ID
- Precio (con decimales)
- Si el envío es gratuito (el usuario pone True o False)

Mostrar la información utilizando tres comillas, print(f'''.....''')

E016

Cambiar el separador del comando print para escribir en la consola "palabras-separadas-por-guiones"

E017

Solicitar un nombre por pantalla y cargarlo en una variable. Mostrar el texto "Nombre: XXXX", utilizando el comando print junto con f y la variable con {}

E018

Imprimir 25 asteriscos con la sentencia print (no escribir los 25 asteriscos dentro del comando)

E019

Pedir por pantalla una edad, y decir si es mayor de edad o menor de edad.

E020

Pedir un número por pantalla y decir si es par o impar.

E021

Pedir un número por pantalla y decir si es positivo, igual a 0 o negativo. Utilizar **if elif y else**.

E022

Pedir un número por pantalla y decir si es par, igual a 0 o impar. Utilizar **if elif y else.**

E023

Pedir un número por pantalla y decir por pantalla “uno” si es 1, “dos” si es 2, “tres” si es 3, o “fuera de rango” si es cualquier otro valor. Utilizar **if elif y else**.

E024

Solicitar una edad y decir si es mayor de edad o menor. Utilizar un **operador ternario**

E025

Solicitar un número y decir si es par o impar. Utilizar un **operador ternario**

E026

Pedir una opción por pantalla y decir por pantalla si es “opción 1” , “opción 2”, “opción 3 o 4”, “ninguna de las anteriores”. Utilizar un **match case**

E027

Solicitar un mes por pantalla y decir a que estación del año pertenece (del mes 1 al 3: invierno, del mes 4 al 6: primavera, del mes 7 al 9: verano y del mes 10 al 12: otoño)

E028

Pedir una nota del 0 al 10 por teclado que puede tener decimales y transformarla en:

- Si está entre 9 y 10: imprimir una A
- Si está entre 8 y menor a 9: imprimir una B
- Si está entre 7 y menor a 8: imprimir una C
- Si está entre 6 y menor a 7: imprimir una D
- Si está entre 0 y menor a 6: imprimir una F
- Cualquier otro valor debe imprimir: Valor incorrecto

E029

Sacar por pantalla los números del 1 al 10. Utilizar un **while**

E030

Pedir un número por pantalla y mostrar con un guion en medio todos los números desde el cero hasta ese valor introducido. Utilizar un **while**

E031

Pedir un número por pantalla y mostrar con un asterisco en medio todos los números desde el número introducido hasta el cero. Utilizar un **while**

E032

Imprimir por pantalla números desde el 0 al 10. Utilizar un **for**

E033

Imprimir por pantalla números desde el 5 al 15. Poner un guion entre los números mostrados. Utilizar un **for**

E034

Imprimir por pantalla números desde el 5 al 15 de dos en dos (5-7-9-11-13-15-). Poner un guion entre los números mostrados. Utilizar un **for**

E035

Pedir un número por pantalla y sacar por pantalla desde ese número hasta el 1. Utilizar un **for**

E036

Pedir una palabra por teclado y sacar por pantalla letra a letra. Utilizar un **for**

E037

Pedir una palabra por teclado y sacar por pantalla letra a letra hasta que aparezca la letra o. Utilizar un **for y break**

E038

Escribir un programa que pida al usuario un número entero positivo y muestre por pantalla la suma de todos los números desde 1 hasta ese número.

E039

Escribir un programa que pida al usuario dos números enteros positivos y muestre por pantalla la tabla de multiplicar del primer número hasta el segundo número.

E040

Escribir un programa que simule un juego de adivinanzas. El programa debe generar un número aleatorio entre 1 y 100 y el usuario debe intentar adivinarlo. El programa debe dar pistas al usuario sobre si su suposición es mayor o menor que el número aleatorio. Al final debe mostrar en cuantos intentos lo has conseguido adivinar.

Generar un número aleatorio del 1 al 100

```
import random
numero_aleatorio = random.randint(1, 100)
```

E041

Escribir un programa que imprima la suma de los números pares del 1 al 100.

E042

Escribir un programa que pida al usuario una cadena de caracteres y luego imprima la cadena al revés.

`len(cadena):` cantidad de caracteres de una cadena

E043

Escribir un programa que imprima una tabla de multiplicar del 1 al 10.

E044

Solicitar un número y escribir una pirámide de números hasta ese número.

Introduce la altura del triángulo: 6

```
     1
    1 2
   1 2 3
  1 2 3 4
 1 2 3 4 5
1 2 3 4 5 6
```

E045

Sacar por pantalla los números de 0 al 9 con un **`for`** y un **`range`**

E046

Pedir por pantalla un mes y si está entre 1 y 12 decir “mes correcto”, en caso contrario, decir “mes incorrecto”. Utilizar **`if`** y **`range`**

E047

Imprimir números divisibles entre 3 que hay en el rango de 0 a 10

E048

Imprimir un rango de números del 2 al 6, incluidos

E049

Crear un rango de 3 al 10 e imprimirlo de 2 en 2 (3,5,7,9)

E050

Crear una lista con los valores 1,2,3,4,5 y mostrar por pantalla la cantidad de elementos que tiene (utilizar la función **`len`**)

E051

Crear una lista con los valores 1,2,3,4,5 y añadir el valor 99 a esta lista. Mostrar la lista por pantalla (no hace falta recorrerla)

E052

Crear una lista con los valores 1,2,3,4,5. Pedir un número por pantalla y si está en la lista decir “Está en la lista” en caso contrario “No está en la lista”

E053

Crear una lista con los valores 1,2,3,4,5. Insertar la palabra “separador” en la posición 2 de la lista y mostrarla por pantalla (no hace falta recorrerla)

E054

Crear una lista con los valores “uno”,”dos”,”tres” y a continuación eliminar el que tiene contenido “dos” a través del método **.remove.** Por último, mostrar el contenido de la lista

E055

Crear una lista con los valores “uno”,”dos”,”tres” y a continuación eliminar el último elemento con el método **.pop** y mostrar por pantalla el elemento eliminado

E056

Crear una lista con los valores “uno”,”dos”,”tres” y a continuación eliminar el elemento con el índice 1. Por último, mostrar el contenido de la lista

E057

Crear una lista con los valores “uno”,”dos”,”tres” y a continuación eliminar toda la lista. Mostrar la lista después para ver que está vacía.

E058

Crear una lista con los valores “uno”,”dos”,”tres” y mostrar la longitud de la lista por pantalla

E059

Crear una lista con los valores “uno”,”dos”,”tres” y cambiar el elemento 1 por “hola”. A continuación, mostrar la lista por pantalla

E060

Crear una lista con los valores “uno”,”dos”,”tres” y mostrar todos los elementos a través de un **for**

E061

Crear una lista con los valores “uno”,”dos”,”tres”, ordenarla y mostrar todos los elementos a través de un **for**

E062

Crear una lista con los valores “uno”,”dos”,”tres”, crear otra lista con los datos de la primera ordenada y mostrar todos los elementos de la ordenada a través de un **for**

E063

Crear una lista con los valores “uno”,”dos”,”tres”, ordenarla de forma inversa y mostrar todos los elementos a través de un **for**

E064

Crear una lista con los valores “uno”,”dos”,”tres”, crear otra lista con los datos de la primera ordenada de forma inversa y mostrar todos los elementos de la ordenada a través de un **for**

E065

Tenemos la siguiente lista: nombres = [“Carlos", “Lara", "María José", “María", “Ana"]. Sacar por pantalla el nombre más largo y el más corto.

E066

Solicitar por pantalla un conjunto de números hasta que se introduzca un 0 y mostrarlo por pantalla en orden descendente. Utilizar listas

E067

Tenemos la siguiente lista: nombres = [“Carlos", “Lara", "María José", “María", “Ana"]. Solicitar por pantalla un nombre y si existe en la lista, borrarlo. A continuación, volver a mostrar la lista.

E068

Dada la siguiente tupla, crear una lista que sólo incluya los números menores a 5

tupla = (13, 1, 8, 3, 2, 5, 8)

E069

Crear un set con los valores 10,20,30,40,20,10 y mostrarlo por pantalla

E070

Crear un set con los valores 10,20,30,40 y eliminar el elemento 30 y a continuación mostrar el set por pantalla

E071

Crear un set con los valores 10,20,30,40 y mostrar la cantidad de elementos que tiene por pantalla

E072

Crear un set con los valores 10,20,30,40, ordenarlos en orden invertido y mostrar el set por pantalla

E073

Tenemos el set s1= {1,2,3} y el set s2 = {3,4,5}.

- Mostrar la unión de los dos
- Mostrar la intersección ellos
- Mostrar la diferencia de ambos

E074

Solicitar nombres por pantalla hasta que el usuario pulse la tecla INTRO y mostrarla sin valores repetidos. Utilizar un set

E075

Crear un diccionario con los valores “María” : 14, “Carlos”: 23 , “Lara”: 18 y mostrarla por pantalla

E076

Crear un diccionario con los valores “María” : 14, “Carlos”: 23 , “Lara”: 18 y mostrar el valor de “Carlos”

E077

Crear un diccionario con los valores “María” : 14, “Carlos”: 23 , “Lara”: 18 y quitar el valor de “Carlos” y mostrar el diccionario por pantalla

E078

Crear un diccionario con los valores “María” : 14, “Carlos”: 23 , “Lara”: 18 y mostrar con un `for` todas las claves (María,Carlos,Lara)

E079

Crear un diccionario con los valores “María” : 14, “Carlos”: 23 , “Lara”: 18 y mostrar con un `for` todas las claves con su valor al lado.

María 14
Carlos 23
Lara 18

E080

Crear un diccionario con los valores “María” : 14, “Carlos”: 23 , “Lara”: 18 y mostrar la cantidad de elementos que tiene (3)

E081

Crear un diccionario con los valores “María” : 14, “Carlos”: 23 , “Lara”: 18 y mostrar el diccionario ordenado por las claves a través de un for

Carlos
Lara
María

E082

Solicitar palabras por pantalla hasta que se deje pulse INTRO sin ninguna palabra y mostrar utilizando un diccionario, la frecuencia de aparición de esas palabras.

E083

Escribir la función suma que reciba dos parámetros y devuelva la suma de ambos. Probar la función con este código

```
print (suma(3,4))
```

E084

Crear la función **nombreCompleto** que reciba dos parámetros (nombre y apellido) y los muestre por pantalla. Probar la función con este código

```
nombreCompleto("Javier","Gómez")
nombreCompleto("Carlos","Gil")
```

E085

Crear la función divide que reciba dos parámetros y devuelva el resultado de la división de ambos parámetros

```
print (divide(3,2))
print (divide(12,3))
```

E086

Crear la función divide que reciba dos parámetros y devuelva el resultado de la división de ambos parámetros. Si recibe un solo parámetro, se divide este entre 1

```
print (divide(3))
print (divide(12,3))
```

E087

Crear la función multiplica que reciba dos parámetros y devuelva el resultado de la multiplicación de ambos parámetros. Si recibe un solo parámetro, debe devolver el cuadrado de este.

```
print (multiplica(3))
print (multiplica(12,3))
```

E088

Crear la función saludo que si recibe un parámetro con un nombre saca por pantalla "Hola" y el nombre, y si no se pasa ningún parámetro saca por pantalla "Hola Mundo"

```
saluda("Javi")
saluda("")
```

1.

E089

Definir una función `maxi()` que tome como argumento dos números y devuelva el mayor de ellos. (Es cierto que python tiene una función max() incorporada, pero hacerla nosotros mismos es un muy buen ejercicio.

```
print(max(10,10))
maxi(10,10)
maxi(10,11)
maxi(30,10)
```

E090

Definir una función `max_de_tres(),` que tome tres números como argumentos y devuelva el mayor de ellos.

```
max_de_tres(10,20,34)
```

E091

Definir la función **`largo_cadena()`** que calcule la longitud de una lista o una cadena dada. (Es cierto que python tiene la función `len()` incorporada, pero escribirla por nosotros mismos resulta un muy buen ejercicio.

```
print(largo_cadena("periquito"))
```

E092

Escribir la función `es_vocal()` que tome un carácter y devuelva True si es una vocal, de lo contrario devuelve False.

```
print (es_vocal("a"))
print (es_vocal("E"))
print (es_vocal("m"))
```

E093

Escribir una función **`sum()`** y una función **`multip()`** que sumen y multipliquen respectivamente todos los números de una lista. Crear la lista por teclado hasta que se introduzca 0.

E094

Definir una función `inversa()` que calcule la inversión de una cadena. Por ejemplo, la cadena "estoy probando" debería devolver la cadena "odnaborp yotse"

```
print(inversa("cosa"))
```

E095

Definir una función `listar()` que reciba una cantidad de parámetros variable y los muestre por pantalla.

```
listar(1,2,3)
listar("uno","dos","tres","cuatro")
```

E096

Crear una función para sumar los valores recibidos de tipo numérico, utilizando argumentos variables `*args` como parámetro de la función y retornar como resultado la suma de todos los valores pasados como argumentos.

```
print(sumar_valores(3, 5, 9, 4, 6))
```

E097

Crear una función para multiplicar los valores recibidos de tipo numérico, utilizando argumentos variables `*args` como parámetro de la función y regresar como resultado la multiplicación de todos los valores pasados como argumentos.

```
print(multiplicar_valores(3,5,15,3))
```

E098

Crear la función `listarTerminos()` que recibe como parámetro un diccionario y tiene que mostrar la `key` y el `valor` por pantalla.

```
print(listarTerminos(uno=1,dos=2,tres=3))
print(listarTerminos(nombre="Javi",edad=32,pelo="rubio",ojos="azules")
)
```

E099

Crear una función `mostarElementos()` y pasar como parámetro una lista. La función tiene que mostrar los elementos de la lista.

```
mostrarElementos([1,2,3,4])
mostrarElementos(["uno","dos","tres"])
```

E100

Crear una función en Python que reciba una lista de palabras y devuelva un diccionario con la cantidad de veces que aparece la palabra en la lista. La carga de la lista de palabras se hace por teclado hasta que se pulsa la tecla INTRO

E101

Crear una función que reciba un texto y devuelva un diccionario con la frecuencia de aparición de las palabras que tiene el texto.

```
# Ejemplo de uso
texto = "Este es un ejemplo de un texto para contar la frecuencia de
las palabras."
frecuencias = contarPalabras(texto)
# Mostrar la frecuencia de las palabras
for palabra, frecuencia in frecuencias.items():
  print(f"Palabra: {palabra}, Fec: {frecuencia}")
```

E102

Crear dos funciones, una que introduzca una lista de números hasta que se pulse INTRO y otra que devuelva la media de los números de la lista

```
# Obtener la lista de números del usuario
numeros = leer_numeros()
# Calcular el promedio de la lista
```

```
promedio = calcular_promedio(numeros)
# Mostrar el promedio
if promedio is None:
  print("No se ha introducido ningún número.")
else:
  print(f"El promedio de la lista es: {promedio}")
```

E103

Definir una función `es_palindromo()` que reconoce palíndromos (es decir, palabras que tienen el mismo aspecto escritas invertidas), ejemplo: `es_palindromo ("radar")` tendría que devolver True.

```
es_palindromo("radar")
es_palindromo("coche")
```

E104

Definir una función `procedimiento()` que tome una lista de números enteros e imprima un histograma en la pantalla. Ejemplo: `procedimiento([4, 9, 7])` debería imprimir lo siguiente:

```
****

*********

*******
```

E105

Crear la clase Persona con los atributos nombre y edad. Cuando se crea el objeto se carga con la información “Vacío”. Probar con el siguiente código.

```
p=Persona()
print(p.nombre)
print(p.edad)
p.nombre="Javi"
p.edad=56
print(p.nombre)
print(p.edad)
```

E106

Crear la clase Persona con los atributos nombre y edad. En el constructor, admitir estos dos valores.

```
p=Persona("Javi",56)
print(p.nombre)
print(p.edad)
```

E107

Crear la clase Persona con los atributos nombre y edad. Crear un método mostrarDetalle() que imprima los valores del objeto

```
p=Persona("Javi",56)
p.mostrarDetalle()
```

Nombre: Javi - Edad: 56

E108

Crear la clase Persona con los atributos nombre y edad. Cuando se imprima el objeto se debe ver su contenido.

```
p=Persona("Javi",56)
print(p)
```

E109

Crear una clase Aritmética con dos propiedades (operandoA y operandoB) y definir cuatro métodos para operar (sumar, restar, multiplicar, dividir) que retornan el resultado

```
aritmetica1 = Aritmetica(5, 3)
print(f'Suma: {aritmetica1.sumar()}')
print(f'Resta: {aritmetica1.restar()}')
print(f'Multiplicación: {aritmetica1.multiplicar()}')
print(f'División: {aritmetica1.dividir():.2f}')
```

Suma: 8
Resta: 2
Multiplicación: 15
División: 1.67

E110

Crear una clase denominada Rectángulo que se inicializa con la longitud de la base y de la altura y tendrá un método que se denomina calcular_area() y retorna el área. La base y la altura se pide por teclado

Proporciona la base: 4
Proporciona la altura: 6
Área rectángulo: 24

E111

Crear una clase denominada Cubo que se inicializa con ancho, alto y profundo y tendrá un método que se denomina calcular_volumen() y retorna el volumen del cubo. El alto, ancho y profundidad se pide por teclado.

Proporciona el ancho: 3
Proporciona el alto: 4
Proprociona lo profundo: 5
Volúmen cubo: 60

E112

Crear la clase Persona con los atributos nombre y edad. Crear la clase Empleado que hereda de persona con el atributo sueldo.

```
empleado1 = Empleado('Juan', 28, 5000)
print(empleado1.nombre)
print(empleado1.edad)
print(empleado1.sueldo)
```

E113

Crear la clase Persona con los atributos nombre y edad y con el método __str__ para mostrar su contenido. Crear la clase Empleado con el método __str__ que muestre su contenido utilizando el __str__ de su padre

```
p=Persona('Javi',56)
print (p)
e = Empleado('Juan', 28, 5000)
print (e)
```

Nombre: Javi Edad: 56
Nombre: Juan Edad: 28 Nombre: Juan Edad: 28

E114

Definir una clase padre llamada Vehículo y dos clases hijas llamadas Coche y Bicicleta, las cuales heredan de la clase Padre Vehículo.

La clase padre debe tener los siguientes atributos y métodos

Vehiculo (Clase Padre):
-Atributos (color, ruedas)
-Métodos (__init__() y __str__)
Coche (Clase Hija de Vehículo) (Además de los atributos y métodos heredados de Vehículo):
-Atributos (velocidad (km/hr))
-Métodos (__init__() y __str__())

Bicicleta (Clase Hija de Vehículo) (Además de los atributos y métodos heredados de Vehículo):
-Atributos (tipo (urbana/montaña/etc)
-Métodos (__init__() y __str__())

```
# Creamos un objeto de la clase Vehiculo
vehiculo = Vehiculo('Rojo', 4)
print(vehiculo)
# Creamos un objeto de la clase hija Coche
coche = Coche('Azul', 4, 150)
print(coche)
# Creamos un objeto de la clase hija Bicicleta
bicicleta = Bicicleta('Blanca', 2, 'Urbano')
print(bicicleta)
```

Color: Rojo, Ruedas: 4
Color: Azul, Ruedas: 4, Velocidad (km/hr): 150
Color: Blanca, Ruedas: 2, Tipo: Urbano

E115

Definir una clase Color con el atributo color y el método __str__. Definir unaclase Figura con el atributo lado y el método __str__. Definir la clase Cuadrado con el atributo textura y que herede de Color y Figura. Debe tener el método __str__

```
c=Color("verde")
print (c)

f=Figura(2)
print (f)

c=Cuadrado("azul",3,"listo")
print (c)
```

Color: verde
Lado: 2
Color: azul Lado: 3 Textura: listo

E116

Crear la clase abstracta Figura con los métodos **`area()`** y **`perimetro().`** Definir la clase Cuadrado que hereda de Figura.

```
c=Cuadrado(5)
print(c.area())
print(c.perimetro())
```

E117

Crear la clase Persona con los atributos nombre y edad. Tener una variable `contador` que cuente la cantidad de Personas creadas

```
a=Persona("Javi",56)
print (Persona.contador)
a=Persona("Carlos",23)
print (Persona.contador)
```

E118

Crear la clase Persona con los atributos nombre y edad. Tener una variable `contador` que cuente la cantidad de Personas creadas. Crear un método estático **`numero()`** que muestre la cantidad de personas.

```
a=Persona("Javi",56)
Persona.numero()
a=Persona("Carlos",23)
Persona.numero()
```

Cantidad de personas: 1
Cantidad de personas: 2

E119

Crear la clase Persona con los atributos id, nombre y edad. El nombre y la edad se reciben por parámetro al crear la clase y el atributo id es una secuencia que tendrá el valor 1 para el primer objeto, 2 para el segundo...

```
persona1 = Persona('Javi', 28)
print(persona1)
persona2 = Persona('Lara', 30)
print(persona2)
persona3 = Persona('Carlos', 25)
print(persona3)
Persona.generar_siguiente_valor()
persona4 = Persona('María', 35)
print(persona4)
print(f'Valor contador personas: {Persona.contador_personas}')
```

Persona [1 Javi 28]
Persona [2 Lara 30]
Persona [3 Carlos 25]
Persona [5 María 35]
Valor contador personas: 5

E120

Crear la clase Vendedor con los atributos nombre y ventas. Sobrecargar el operador + para sumar dos vendedores y su resultado sea como nombre “Total” y las ventas, la suma de los dos.

```
a=Vendedor("Javi",23)
b=Vendedor("María",55)
c=a+b
print(a)
print(b)
print(c)
```

Nombre: Javi Ventas: 23
Nombre: María Ventas: 55
Nombre: TOTAL Ventas: 78

E121

Queremos definir una clase abstracta llamada Forma que contenga un método abstracto llamado **`calcular_area()`** que deberá ser implementado por las clases concretas que heredan de ella, como **`Rectangulo`** y **`Circulo`**. Además, queremos definir un método común llamado **`obtener_nombre()`** que devuelva el nombre de la forma.

```
# Crear instancias de las clases
mi_rectangulo = Rectangulo(5, 10)
mi_circulo = Circulo(3)

# Imprimir nombre y área de las formas
print(mi_rectangulo.obtener_nombre())  # Salida: Rectangulo
print(mi_rectangulo.calcular_area())   # Salida: 50
print(mi_circulo.obtener_nombre())     # Salida: Circulo
print(mi_circulo.calcular_area())      # Salida: 28.26
```

E122

Pedir un número por pantalla y no dejar de pedirlo hasta que no se introduzca un número correcto.

Introduce valor: s
Valor no numérico. Vuelve a intentarlo
Introduce valor: 2

E123

Pedir dos números correctos por pantalla y sacar el resultado de su división. Controlar todos los errores.

Introduce valor A: s
ERROR: Valor no numérico. Vuelve a intentarlo
Introduce valor A: 1
Introduce valor B: 0
ERROR: El divisor es 0
Introduce valor A: 4
Introduce valor B: 2
División: 2.0

E124

Crear un error personalizado (miError). Se debe lanzar este error cuando el valor numérico introducido sea menor de 0

Introduce valor mayor de 0: -2
ERROR: debe mayor de 0 1001
Introduce valor mayor de 0:

E125

Solicitar dos números por pantalla y dar el resultado de su división. Los números no deben de ser iguales. Controlar los errores con try de que deben ser números, la división entre cero y que los números no deben ser iguales. Repetir la solicitud de datos hasta que todo esté correcto.

E126

Crear el archivo a.txt y grabar dentro el texto “archivo”

E127

Mostrar por pantalla el contenido del archivo a.txt

E128

Crear el archivo b.txt y grabar dentro el texto “El niño jugó con su camión”

E129

Mostrar por pantalla el contenido del archivo b.txt

E130

Crear el archivo c.txt con `with` grabando en tres líneas diferentes “Lin 1” “Lin 2” “Lin 3”

E131

Añadir al archivo c.txt con `with` en tres líneas diferentes “Lin 4” “Lin 5” “Lin 6”

E132

Leer al archivo c.txt con `with`. Si el archivo no existe, el programa debe decir “Archivo no encontrado”

E133

Leer los tres primeros caracteres del archivo a.txt con `with`. Si el archivo no existe, el programa debe decir “Archivo no encontrado”

E134

Leer la primera línea del archivo c.txt con `with`. Si el archivo no existe, el programa debe decir “Archivo no encontrado”

E135

Leer el archivo c.txt con `with e introduce su contenido en una lista en donde cada línea sea un elemento de la lista. Muestra la lista`. Si el archivo no existe, el programa debe decir “Archivo no encontrado”

E136

Pedir palabras por pantalla hasta que se pulsa la tecla INTRO. Grabar cada palabra en una línea en un archivo denominado “palabras.txt”

E137

Decir cuantas palabras están grabadas en el archivo “palabras.txt”. Si el archivo no existe, el programa debe decir “Archivo no encontrado”

E138

Decir cual es la palabra mas larga y su longitud del archivo “palabras.txt”. Si el archivo no existe, el programa debe decir “Archivo no encontrado”

E139

Decir cuantas vocales están grabadas en el archivo “palabras.txt”. Si el archivo no existe, el programa debe decir “Archivo no encontrado”

E140

Mostrar el contenido del archivo “palabras.txt”. Si el archivo no existe, el programa debe decir “Archivo no encontrado”

E141

Pedir números por pantalla hasta que se introduce un 0. Grabar cada número en una línea en un archivo denominado “numeros.txt”. Si se escribe algo que no sea un número, sacar el mensaje “Solo se admiten números” y volver a pedirlo

E142

Pedir números por pantalla hasta que se introduce un 0. **Añadir** cada número en una línea del archivo denominado “numeros.txt”. Si se escribe algo que no sea un número, sacar el mensaje “Solo se admiten números” y volver a pedirlo.

E143

Mostrar el contenido del archivo “numeros.txt”. Si el archivo no existe, el programa debe decir “Archivo no encontrado”.

E144

Mostrar la suma de todos los números del archivo “numeros.txt”. Si el archivo no existe, el programa debe decir “Archivo no encontrado”.

E145

Mostrar la media de todos los números del archivo “numeros.txt”. Si el archivo no existe, el programa debe decir “Archivo no encontrado”.

E146

Reescribir el archivo “números.txt” quitando todos los números repetidos. Si el archivo no existe, el programa debe decir “Archivo no encontrado”.

E147

Reescribir el archivo “números.txt” ordenando los números de menor a mayor. Si el archivo no existe, el programa debe decir “Archivo no encontrado”.

E148

Pedir por pantalla una provincia y su cantidad de habitantes y grabar un archivo llamado “poblacion.txt” que por cada línea tenga la provincia y cantidad de habitantes separado por coma. Dejar de pedir provincias cuando se deje la provincia en blanco

E149

Leer el archivo “poblacion.txt” y decir que provincia tiene más habitantes

E150

Leer el archivo “poblacion.txt” y reescribirlo de manera que, si hay dos provincias iguales, sumar los habitantes de estos registros y grabar la suma.

https://github.com/jagode67/Ejercicios-Python