Programando con Python

Pregunta 1

Crea una función que reciba los tres coeficientes a, b y c para resolver una ecuación de segundo grado. Muestra la solución por pantalla y ayúdate de la librería math para acceder a la función raíz cuadrada.

Para resolver la ecucaión de segundo grado nos apoyaremos en la formula:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

donde $a \neq 0$. Esto nos devolverá dos posibles soluciones a nuestra ecuación

```
import math
def ecuacion_seg_grado(a, b, c):
    if a != 0:
        x1 = (-b + math.sqrt(b**2 - (4*a*c)))/(2*a)
        x2 = (-b - math.sqrt(b**2 - (4*a*c)))/(2*a)
    else:
        print('El valor de a no puede ser cero')
    return x1, x2

a = float(input('Ingrese el valor de a: '))
b = float(input('Ingrese el valor de b: '))
c = float(input('Ingrese el valor de c: '))
ecuacion_seg_grado(a, b, c )
```

```
Ingrese el valor de a: 1
Ingrese el valor de b: -3
Ingrese el valor de c: 1
(1.434258545910665, 0.2324081207560018)
```

Pregunta 2

Crea una función que lea una frase de teclado y nos diga si es o no un palíndromo (frase que se lee igual de izquierda a derecha o al revés como por ejemplo “La ruta nos aportó otro paso natural”)

```
def es_palindromo(frase):
  frase_sin_espacios = frase.lower().replace(" ", "")

  # Invertir la frase.
  frase_invertida = frase_sin_espacios[::-1]

  # Comparar la frase original con la invertida.
  return frase_sin_espacios == frase_invertida

# Ejemplo de uso
frase = input("Introduce una frase: ")

if es_palindromo(frase):
  print(f"La frase '{frase}' es un palíndromo.")
else:
  print(f"La frase '{frase}' no es un palíndromo.")
```

```
Introduce una frase: Sebastian
La frase 'Sebastian' no es un palíndromo.
```

Pregunta 3

Crea un diccionario que tenga por claves los números del 1 al 10 y como valores sus raíces cuadradas.

```
diccionario = {}
for i in range(1, 11):
    diccionario[i] = {i:math.sqrt(i)}
diccionario
```

```
{1: {1: 1.0},
 2: {2: 1.4142135623730951},
 3: {3: 1.7320508075688772},
 4: {4: 2.0},
```

```
    5: {5: 2.23606797749979},
    6: {6: 2.449489742783178},
    7: {7: 2.6457513110645907},
    8: {8: 2.8284271247461903},
    9: {9: 3.0},
    10: {10: 3.1622776601683795}}
```

Pregunta 4

Crea un diccionario que tenga como claves las letras del alfabeto castellano y como valores los símbolos del código morse (los tienes todos en la Wikipedia). A continuación crea una función que lea una frase del teclado y te la convierta a morse utilizando el diccionario anterior.

```
MORSE_ALFABETO = {
  "a": ".-",
  "b": "-...",
  "c": "-.-.",
  "d": "-..",
  "e": ".",
  "f": "..-.",
  "g": "--.",
  "h": "....",
  "i": "..",
  "j": ".---",
  "k": "-.-",
  "l": ".-..",
  "m": "--",
  "n": "-.",
  "ñ": "--.--",
  "o": "---",
  "p": ".--.",
  "q": "--.-",
  "r": ".-.",
  "s": "...",
  "t": "-",
  "u": "..-",
  "v": "...-",
  "w": ".--",
  "x": "-..-",
  "y": "-.--",
  "z": "--..",
  "á": ".--.-",
  "é": "..-..",
  "í": "..--",
  "ó": "---.",
  "ú": "....-",
}

def convertir_a_morse(frase):
  frase_morse = ""
  for letra in frase.lower():
    if letra in MORSE_ALFABETO:
      frase_morse += MORSE_ALFABETO[letra] + " "
    else:
      frase_morse += letra + " "
  return frase_morse

frase = input("Introduce una frase: ")

frase_morse = convertir_a_morse(frase)

print(f"La frase '{frase}' en código morse es: {frase_morse}")

```

```
La frase 'El jefe maestro escapo' en código morse es: . .-..   .--- . ..-. .   -- .- . ... - .-. ---   . ... -.-. .- .--. ---
```

Pregunta 5

Crea una función que dados dos diccionarios nos diga qué claves están presentes en ambos

```
def claves_en_ambos(diccionario1, diccionario2):
  claves_comunes = set(diccionario1.keys()) & set(diccionario2.keys())
  return claves_comunes

diccionario1 = {"a": 1, "b": 2, "c": 3}
diccionario2 = {"b": 2, "c": 4, "d": 5}

claves_comunes = claves_en_ambos(diccionario1, diccionario2)

print(f"Las claves presentes en ambos diccionarios son: {claves_comunes}")
```

```
Las claves presentes en ambos diccionarios son: {'c', 'b'}
```

Pregunta 6

Crea una función que dado un número N nos diga si es primo o no (tiene que ir dividiendo por todos los números x comprendidos entre 2 y el número N − 1 y ver si la división de N x tiene resto cero o no).

```
def es_primo(numero):
  if numero <= 1:
    return False

  for i in range(2, numero):
    if numero % i == 0:
      return False

  return True

numero = int(input("Introduce un número: "))

if es_primo(numero):
  print(f"El número {numero} es primo.")
else:
  print(f"El número {numero} no es primo.")
```

```
El número 113 es primo.
```

Pregunta 7

Investiga la documentación de la clase string y crea un método que lea una frase del teclado y escriba la primera letra de cada palabra en Mayúscula

```
frase = input('Ingresa una frase: ').strip()
print(frase)
print()
print(frase.title())
```

```
cortana esta manipulando al jefe maestro

Cortana Esta Manipulando Al Jefe Maestro
```

Pregunta 8

Crea una función que calcule el máximo común divisor de dos números introducidos por el usuario por teclado.

```
def mcd(a, b):
  while b:
    a, b = b, a % b
  return a

a = int(input("Introduce el primer número: "))
b = int(input("Introduce el segundo número: "))

mcd_ab = mcd(a, b)

print(f"El máximo común divisor de {a} y {b} es: {mcd_ab}")
```

```
El máximo común divisor de 12 y 18 es: 6
```

Pregunta 9

Investiga el Cifrado del César y crea una función que lo reproduzca en Python. Cada letra del mensaje original se desplaza tres posiciones en el alfabeto estándar. La A se convierte en la D, la B se convierte en la E, la C se convierte en la F. . . y cuando se acaba el alfabeto se le vuelve a dar la vuelta: la X se convierte en la A, la Y en la B y la X en la C. Los números no sufren ninguna modificación

El cifrado del César es un método de cifrado simple que consiste en sustituir cada letra del mensaje original por otra que se encuentra a un número fijo de posiciones en el alfabeto. En este caso, el desplazamiento es de 3 posiciones.

```
def cifrar_cesar(mensaje):
  alfabeto = "abcdefghijklmnñopqrstuvwxyzáéíóú"
  mensaje_cifrado = ""

  for letra in mensaje:
    if letra in alfabeto:
      posicion = alfabeto.find(letra)
      nueva_posicion = (posicion + 3) % len(alfabeto)
      nueva_letra = alfabeto[nueva_posicion]
      mensaje_cifrado += nueva_letra
    else:
      mensaje_cifrado += letra

  return mensaje_cifrado

mensaje = "Hola mundo!"
mensaje_cifrado = cifrar_cesar(mensaje)

print(f"Mensaje original: {mensaje}")
print(f"Mensaje cifrado: {mensaje_cifrado}")
```

```
Mensaje original: Hola mundo!
Mensaje cifrado: Hrñd oxpgr!
```

Pregunta 10

Dado una lista de nombres de persona, escribe una función que los ordene de tres formas diferentes:

- De forma alfabética
- De forma alfabética invertida
- De nombre más corto al más largo.

```
def ordenar_alfabetico(nombres):
    return sorted(nombres)

def ordenar_alfabetico_invertido(nombres):
    return sorted(nombres, reverse=True)

def ordenar_por_longitud(nombres):
    return sorted(nombres, key=len)

nombres = ["Ana", "Juan", "María", "Pedro", "Laura","Sebastian"]
nombres_ordenados_alfabetico = ordenar_alfabetico(nombres)
nombres_ordenados_alfabetico_invertido = ordenar_alfabetico_invertido(nombres)
nombres_ordenados_por_longitud = ordenar_por_longitud(nombres)

print(f"Nombres ordenados alfabéticamente: {nombres_ordenados_alfabetico}")
print(f"Nombres ordenados alfabéticamente invertida: {nombres_ordenados_alfabetico_invertido}")
print(f"Nombres ordenados por longitud: {nombres_ordenados_por_longitud}")
```

```
Nombres ordenados alfabéticamente: ['Ana', 'Juan', 'Laura', 'María', 'Pedro', 'Sebastian']
Nombres ordenados alfabéticamente invertida: ['Sebastian', 'Pedro', 'María', 'Laura', 'Juan', 'Ana']
Nombres ordenados por longitud: ['Ana', 'Juan', 'María', 'Pedro', 'Laura', 'Sebastian']
```