El futuro digital es de todos

MinTIC

Módulos

Rogerio Orlando Beltrán Castro

Universidad Industrial de Santander

MisiónTIC2022

1. Módulo os

- El módulo os nos permite acceder a funcionalidades dependientes del Sistema Operativo. Sobre todo, aquellas que nos refieren información sobre el entorno del mismo y nos permiten manipular la estructura de directorios

Archivos y directorios

El módulo os nos provee de varios métodos para trabajar de forma portable con las funcionalidades del sistema operativo. Veremos a continuación, los métodos más destacados de este módulo.

Archivos y directorios

Descripción	Método
Saber si se puede acceder a un archivo o directorio	`os.access(path, modo_de_acceso)`
Conocer el directorio actual	`os.getcwd()`
Cambiar de directorio de trabajo	`os.chdir(nuevo_path)`
Cambiar al directorio de trabajo raíz	`os.chroot()`
Cambiar los permisos de un archivo o directorio	`os.chmod(path, permisos)`
Cambiar el propietario de un archivo o directorio	`os.chown(path, permisos)`
Crear un directorio	`os.mkdir(path[, modo])`
Crear directorios recursivamente	`os.mkdirs(path[, modo])`
Eliminar un archivo	`os.remove(path)`
Eliminar un directorio	`os.rmdir(path)`
Eliminar directorios recursivamente	`os.removedirs(path)`
Renombrar un archivo	`os.rename(actual, nuevo)`
Crear un enlace simbólico	`os.symlink(path, nombre_destino)`

El módulo os y las variables de entorno

El módulo os también nos provee de un diccionario con las variables de entorno relativas al sistema. Se trata del diccionario environ:

```
import os
for variable, valor in os.environ.items():
    print(variable, valor)
```

os.path

El módulo os también nos provee del submódulo path (os.path) el cual nos permite acceder a ciertas funcionalidades relacionadas con los nombres de las rutas de archivos y directorios. Entre ellas, las más destacadas se describen en la siguiente tabla:

Descripción	Método
Ruta absoluta	`os.path.abspath(path)`
Directorio base	`os.path.basename(path)`
Saber si un directorio existe	`os.path.exists(path)`
Conocer último acceso a un directorio	`os.path.getatime(path)`
Conocer tamaño del directorio	`os.path.getsize(path)`
Saber si una ruta es absoluta	`os.path.isabs(path)`
Saber si una ruta es un archivo	`os.path.isfile(path)`
Saber si una ruta es un directorio	`os.path.isdir(path)`
Saber si una ruta es un enlace simbólico	`os.path.islink(path)`
Saber si una ruta es un punto de montaje	`os.path.ismount(path)`

https://docs.python.org/3/library/os.path.html

2. Módulo sys

El módulo sys es el encargado de proveer variables y funcionalidades, directamente relacionadas con el intérprete

https://docs.python.org/3/library/sys.html

Variables del módulo sys

Variable	Descripción
`sys.argv`	Retorna una lista con todos los argumentos pasados por línea de comandos. Al ejecutar `python modulo.py arg1 arg2`, retornará una lista: `['modulo.py', 'arg1', 'arg2']`
`sys.executable`	Retorna el path absoluto del binario ejecutable del intérprete de Python
`sys.maxint`	Retorna el número positivo entero mayor, soportado por Python
`sys.platform`	Retorna la plataforma sobre la cuál se está ejecutando el intérprete
`sys.version`	Retorna el número de versión de Python con información adicional

Métodos del módulo sys

Método	Descripción
`sys.exit()`	Forzar la salida del intérprete
`sys.getdefaultencoding()`	Retorna la codificación de caracteres por defecto
`sys.getfilesystemencoding()`	Retorna la codificación de caracteres que se utiliza para convertir los nombres de archivos unicode en nombres de archivos del sistema
`sys.getsizeof(object[, default])`	Retorna el tamaño del objeto pasado como parámetro. El segundo argumento (opcional) es retornado cuando el objeto no devuelve nada.

3. Módulo subprocess

El módulo subprocess es aquel que nos permite trabajar de forma directa con órdenes del sistema operativo.

Advertencia: El módulo subprocess** se presenta en este capítulo solo con fines educativos, mostrando ejemplos básicos y sencillos. Por lo tanto, se recomienda tener mucho cuidado en el uso de este módulo, desaconsejando su uso para órdenes que puedan comprometer el sistema**.

https://docs.python.org/3/library/subprocess.html

math- Funciones matemáticas

Este módulo proporciona acceso a las funciones matemáticas definidas por el estándar C.

Estas funciones no se pueden utilizar con números complejos; utilice las funciones del mismo nombre del cmathmódulo si necesita soporte para números complejos. La distinción entre funciones que admiten números complejos y aquellas que no lo hacen se hace ya que la mayoría de los usuarios no quieren aprender tantas matemáticas como se requiere para comprender números complejos. Recibir una excepción en lugar de un resultado complejo permite la detección más temprana del número complejo inesperado utilizado como parámetro, de modo que el programador pueda determinar cómo y por qué se generó en primer lugar.

Este módulo proporciona las siguientes funciones. Excepto cuando se indique explícitamente lo contrario, todos los valores devueltos son flotantes.

Funciones de representación y teoría de números

- math.ceil(x)

Devuelve el techo de x , el número entero más pequeño mayor o igual que x . Si x no es un flotante, delega en x.__ceil__(), que debería devolver un Integralvalor.

- math.comb(n , k)

Devuelve el número de formas de elegir k elementos de n elementos sin repetición y sin orden.

Evalúa hasta cuando y evalúa a cero cuando .n! / (k! * (n - k)!)k <= nk > n

También se llama coeficiente binomial porque es equivalente al coeficiente del k-ésimo término en la expansión polinomial de la expresión .(1 + x) ** n

Aumenta TypeErrorsi alguno de los argumentos no son números enteros. Aumenta ValueErrorsi alguno de los argumentos es negativo.

- math.copysign(x , y)

Devuelve un flotador con la magnitud (valor absoluto) de x pero el signo de y . En plataformas que admiten ceros firmados, devuelve -1.0 .copysign(1.0, -0.0)

Funciones de representación y teoría de números

- math.fabs(x)

Devuelve el valor absoluto de x .

- math.factorial(x)

Devuelve x factorial como un número entero. Aumenta ValueErrorsi x no es integral o es negativo.

En desuso desde la versión 3.9: la aceptación de flotantes con valores integrales (como 5.0) está en desuso.

- math.floor(x)

Devuelve el piso de x , el entero más grande menor o igual que x . Si x no es un flotante, delega en x.__floor__(), que debería devolver un Integralvalor.

Funciones de representación y teoría de números

- math.fmod(x , y)

Retorno , según lo definido por la biblioteca de la plataforma C. Tenga en cuenta que es posible que la expresión de Python no devuelva el mismo resultado. La intención del estándar C es que sea exactamente (matemáticamente; con precisión infinita) igual a para algún número entero n tal que el resultado tenga el mismo signo que x y una magnitud menor que . Python devuelve un resultado con el signo de y en su lugar, y puede que no sea exactamente computable para argumentos flotantes. Por ejemplo, es , pero el resultado de Python es , que no se puede representar exactamente como un flotante, y redondea a lo sorprendente . Por esta razón, la funciónfmod(x, y)x % yfmod(x, y)x - n*yabs(y)x % yfmod(-1e-100, 1e100)-1e-100-1e-100 % 1e1001e100-1e-1001e100fmod()generalmente se prefiere cuando se trabaja con flotantes, mientras que se prefiere Python cuando se trabaja con enteros.x % y

- math.frexp(x)

Devuelve la mantisa y el exponente de x como el par . m es un flotante ye es un número entero tal que exactamente. Si x es cero, devuelve , de lo contrario . Se utiliza para "separar" la representación interna de un flotador de forma portátil.(m, e)x == m * 2**e(0.0, 0)0.5 <= abs(m) < 1

- math.fsum(iterable)

Devuelve una suma precisa de valores en coma flotante en el iterable.

Funciones de representación y teoría de números

- math.gcd(* enteros)

Devuelve el máximo común divisor de los argumentos enteros especificados. Si alguno de los argumentos es distinto de cero, el valor devuelto es el entero positivo más grande que es divisor de todos los argumentos. Si todos los argumentos son cero, el valor devuelto es 0. gcd()sin argumentos devuelve 0.

- math.isclose(a , b , * , rel_tol = 1e-09 , abs_tol = 0.0)

Retorno Truesi los valores a y b son cerca uno del otro y Falsede lo contrario.

Si dos valores se consideran cercanos o no, se determina de acuerdo con las tolerancias absolutas y relativas dadas.

rel_tol es la relativa tolerancia - es la diferencia máxima permitida entre una y b , en relación con el mayor valor absoluto de una o b . Por ejemplo, para establecer una tolerancia del 5%, pase rel_tol=0.05. La tolerancia predeterminada es 1e-09, que asegura que los dos valores sean iguales dentro de aproximadamente 9 dígitos decimales. rel_tol debe ser mayor que cero.

abs_tol es la tolerancia absoluta mínima, útil para comparaciones cercanas a cero. abs_tol debe ser al menos cero.

Si no se producen errores, el resultado será: .abs(a-b) <= max(rel_tol * max(abs(a), abs(b)), abs_tol)

Los valores especiales de IEEE 754 de NaN, infy -infse manejarán de acuerdo con las reglas de IEEE. En concreto, NaNno se considera cercano a ningún otro valor, incluido NaN. infy -infsolo se consideran cercanos a ellos mismos.

Funciones de representación y teoría de números

- math.isfinite(x)

Devuelve Truesi x no es infinito ni NaN, y de lo Falsecontrario. (Tenga en cuenta que 0.0 se considera finito).

- math.isinf(x)

Devuelve Truesi x es un infinito positivo o negativo, y de lo Falsecontrario.

- math.isnan(x)

Devuelve Truesi x es un NaN (no un número) y de lo Falsecontrario.

- math.isqrt(n)

Devuelve la raíz cuadrada entera del entero no negativo n . Este es el piso de la raíz cuadrada exacta de n , o equivalentemente el mayor entero a tal que $a^2 \leq n$.

Para algunas aplicaciones, puede ser más conveniente tener el menor número entero a tal que $n \leq a^2$, o en otras palabras, el techo de la raíz cuadrada exacta de n . Para n positivo , esto se puede calcular usando .a = 1 + isqrt(n - 1)

Funciones de potencia y logarítmicas

math.exp(x)

Devuelve e elevado a la potencia x , donde e = 2.718281… es la base de los logaritmos naturales. Suele ser más preciso que o .math.e ** xpow(math.e, x)

math.expm1(x)

Devuelve e elevado a la potencia x , menos 1. Aquí e es la base de los logaritmos naturales. Para pequeños flotadores x , la resta en puede resultar en una pérdida significativa de precisión ; la función proporciona una forma de calcular esta cantidad con total precisión:exp(x) - 1expm1()

Funciones de potencia y logarítmicas

- math.log(x [, base])

Con un argumento, devuelve el logaritmo natural de x (a la base e).

Con dos argumentos, devuelve el logaritmo de x a la base dada , calculada como log(x)/log(base).

- math.log1p(x)

Devuelve el logaritmo natural de 1 + x (base e). El resultado se calcula de forma precisa para x cerca de cero.

- math.log2(x)

Devuelve el logaritmo en base 2 de x . Suele ser más preciso que .log(x, 2)

Funciones de potencia y logarítmicas

- math.log10(x)

Devuelve el logaritmo de x en base 10 . Suele ser más preciso que .log(x, 10)

- math.pow(x , y)

Vuelve x elevado al poder y. Los casos excepcionales siguen el Anexo 'F' de la norma C99 en la medida de lo posible. En particular, y siempre regresa , incluso cuando es un cero o un NaN. Si ambos y son finitos, es negativo y no es un número entero, entonces no está definido y aumenta .pow(1.0, x)pow(x, 0.0)1.0xxyxypow(x, y)ValueError. A diferencia del **operador integrado , math.pow()convierte ambos argumentos a tipo float. Utilice **o la pow()función incorporada para calcular potencias enteras exactas.

- math.sqrt(x)

Devuelve la raíz cuadrada de x .

Funciones trigonométricas

- math.acos(x)

Devuelve el arco coseno de x , en radianes. El resultado está entre 0y pi.

- math.asin(x)

Devuelve el arco seno de x , en radianes. El resultado está entre -pi/2y pi/2.

- math.atan(x)

Devuelve el arco tangente de x , en radianes. El resultado está entre -pi/2y pi/2.

Conversión angular

- math.degrees(x)

Convierta el ángulo x de radianes a grados.

- math.radians(x)

Convierta el ángulo x de grados a radianes.

Funciones hiperbólicas

Las funciones hiperbólicas son análogas a las funciones trigonométricas que se basan en hipérbolas en lugar de círculos.

- math.acosh(x)

Devuelve el coseno hiperbólico inverso de x .

- math.asinh(x)

Devuelve el seno hiperbólico inverso de x .

- math.atanh(x)

Devuelve la tangente hiperbólica inversa de x .

Funciones especiales

- math.erfc(x)

Devuelve la función de error complementaria en x . La función de error complementario se define como . Se usa para valores grandes de x donde una resta de uno causaría una pérdida de significancia .1.0 - erf(x)

- math.gamma(x)

Devuelve la función Gamma en x .

Constantes

- math.pi

La constante matemática π = 3,141592…, hasta la precisión disponible.

- math.e

La constante matemática e = 2.718281…, hasta la precisión disponible.

- math.tau

La constante matemática τ = 6.283185…, hasta la precisión disponible. Tau es una constante circular igual a 2 π , la relación entre la circunferencia de un círculo y su radio. Para obtener más información sobre Tau, vea el video de Vi Hart Pi is (still) Wrong , y comience a celebrar el día de Tau comiendo el doble de pastel.

- math.inf

Un infinito positivo de coma flotante. (Para infinito negativo, utilice -math.inf.) Equivalente a la salida de float('inf').

- math.nan

Un valor de punto flotante “no es un número” (NaN). Equivalente a la salida de float('nan').

Ejercicio 1

Realiza un programa que devuelva el área de un círculo a partir de un radio:

Área de un círculo = pi*r*r

Solución Ejercicio 1

```
import math
radio=int(input('Ingrese el radio del circulo:'))
area=math.pi*math.pow(radio,2)
print('El área del circulo es:',area)
```

Ejercicio 2

Elaborar un algoritmo que permita calcular el número de micro discos 3.5 necesarios para hacer una copia de seguridad, de la información almacenada en un disco cuya capacidad se conoce. Hay que considerar que el disco duro está lleno de información, además expresado en gigabyte. Un micro disco tiene 1.44 megabyte y un gigabyte es igual a 1,024 megabyte.

Solución Ejercicio 2

```
import math #librería necesaria para usar funciones Matemáticas
#en este caso math.ceil(), que redondea un numero al Entero superior
#Decoración: Nombre del Algoritmo
print("-------------------------------------------------------")
print("Ejercicio5: NÚMERO DE MICRO DISCOS 3.5 NECESARIOS")
print("-------------------------------------------------------")
#Entradas
print("Ingrese GB: ")
GB = float( input())
#Proceso
MG = GB*1024
MD = MG/1.44
#Salida
print("\nSALIDA: ")
print("-------------------------------------------------------")
print(MD)
#En caso de Decimal Aproximar al siguiente entero
#Ya que la parte decimal debe ser almacenada en otro DISCO 3.5
print("Numero de Discos necesarios: ", math.ceil(MD))
```

Ejercicio 3

Si se conoce la longitud de dos de los lados de un triángulo (b y c) y el ángulo entre ellos (alfa), expresado en grados sexagesimales, la longitud del tercer lado (a) se calcula por la fórmula:

$$a^2 = b^2 + c^2 - 2bc*\cos(alfa)$$

Solución Ejercicio 3

```
import math #Necearia para fórmulas matemáticas
#Decoración: Nombre del Algoritmo
print("------------------------------------------------------")
print("Complemento7: CALCULAR EL TERCER LADO DEL TRIANGULO")
print("------------------------------------------------------")
#Constantes
PI = 3.1416
#Entradas
print("Ingrese lados del triángulo:")
b = float( input("Lado b: "))
c = float( input("Lado c: "))
print("Ingrese el ángulo en grados sexagesimales:")
alfa = float( input())
#Proceso
#fórmula para calcular lado 'a' con alfa transformado
a = ( b**2 + c**2 - 2*b*c * math.cos( alfa*PI/180 ) )**0.50
#Salida
print("\nSALIDA: ")
print("------------------------------------------------------")
print("El lado a es:", a)
```

Ejercicio 4

Realizar un programa que solicite un rango para generar 10 números de los cuales se debe identificar cual es el mayor y el menor.

Solución Ejercicio 4

```
import random
rango1=int(input('Ingrese el primer dato del rango:'))
rango2=int(input('Ingrese el segundo dato del rango:'))
mayor=0
menor=0
for i in range(10):
    numero=random.randint(rango1,rango2)
    print('El número generado es:',numero)
    if i==1:
        mayor=numero
        menor=numero
    elif numero>mayor:
        mayor=numero
    elif numero<menor:
        menor=numero
print('El número mayor es:',mayor,' y el número menor es:',menor)
```

Ejercicio 5

Crea un programa, que pida los coeficientes a, b y c de una ecuación de segundo grado y calcule la solución

$$a\,x^2 + b\,x + c = 0$$

La formula matemática que resuelve esta ecuación es la siguiente:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}, \text{es decir, hay dos soluciones } x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \text{ y } x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

Ten en cuenta los siguientes casos especiales en la resolución:

- Si a = 0
- Si $b^2 - 4ac < 0$ las raíces son imaginarias, pero se puede mostrar el resultado separando la parte real de la imaginaria., o bien decir que no tiene resultados reales.

Solución Ejercicio

```
from math import sqrt
A = int(input("Ingrese el coeficiente de la variable cuadrática\n"))
B = int(input("Ingrese el coeficiente de la variable lineal\n"))
C = int(input("Ingrese el término independiente\n"))
x1= 0
x2= 0
if A!=0:
    if ((B**2)-4*A*C) < 0:
        print("La solución de la ecuación es con números complejos")
    else:
        x1 = (-B+sqrt(B**2-(4*A*C)))/(2*A)
        x2 = (-B-sqrt(B**2-(4*A*C)))/(2*A)
        print("Las soluciones de la ecuación son:")
        print(x1)
        print(x2)
else:
    if B!=0:
        x=-C/B
        print('Solución es: ',x)
    else:
        if B!=0:
            print('La ecuación no tiene solución')
        else:
            print('La ecuación tiene solución infinitas soluciones')
```