PYTHON

Módulo 9 – Módulo matemático

MÓDULO MATEMÁTICO

- Módulo math
  - Permite acceder a las funciones matemáticas
  - https://docs.python.org/3/library/math.html
  - import math
  - Se invocan sobre la clase math
  - Funciones:
    - ceil()→Obtiene el entero más próximo superior.
    - comb()→Proporciona el número de combinaciones de n elementos tomados de m en m (coeficiente binomial)
    - **fabs()→Proporciona el valor absoluto de un elemento**
    - factorial()→Obtiene el factorial de un número
    - floor()→Obtiene el entero más próximo inferior.
    - **fsum()→Obtiene la suma de todos los elementos de un iterable.**

MÓDULO MATEMÁTICO

- Módulo math
  - Funciones (continuación):
    - gcd()→Máximo común divisor de un conjunto de números.
    - **isclose()→Determina si dos valores son próximos dada una tolerancia.**
    - isfinite()→Indica si un número es finito.
    - isinf()→Indica si un número es infinito (positivo o negativo)
    - isnan()→Indica si el parámetro NO es un número (para saber si es la constante nan)
    - isqrt()→Obtiene la raíz cuadrada de un número entero.
    - lcm()→Mínimo común múltiplo.
    - **modf()→Separa la parte entera de la fraccionaria de un número.**
      - **>>> math.modf(24.3)**
      - **(0.3000000000000007, 24.0)#Usar función round para resolver *floatint point error***
    - nextafter(*n*,+-math.inf)→Proporciona el siguiente valor flotante a un número dado en una dirección.
    - perm()→ Calcula el número de permutaciones de n elementos tomados de m en m.

MÓDULO MATEMÁTICO

- Módulo math
  - Funciones (continuación):
    - **prod()→Calcula el producto de todos los elementos de un iterable.**
    - **remainder()→Obtiene el resto de una división.**
    - **trunc()→Trunca un número.**
  - Funciones logarítmicas y exponenciales.
    - exp()→Calcula el resultado de elevar e a x.
    - log()→Logaritmo
    - pow()→Potencia.
    - sqrt()→Raíz cuadrada.
  - Funciones trigonométricas.
    - acos(),asin(),atan(),cos(),dist(),sin(),tan()
  - Funciones de conversión angular.

MÓDULO MATEMÁTICO

- Módulo math
  - Funciones hiperbólicas.
  - Funciones especiales.
  - Constantes:
    - math.pi
    - math.e
    - math.tau
    - math.inf
    - math.nan

MÓDULO MATEMÁTICO

- Módulo random
  - Implementa funciones de generación de números pseudoaleatorios.
  - https://docs.python.org/es/3/library/random.html
  - import random
  - **random.random()→ Genera un número en el rango [0.0 y 1.0).**
  - **random.seed()→Asigna una semilla**
  - random.randbytes()→Genera un número aleatorio de bytes
  - random.randrange()→Genera enteros aleatorios dentro de un rango pudiendo indicar el paso.
  - random.randint()→Genera un entero aleatorio en un intervalo.
  - **random.choice(*seq*)→Devuelve un elemento aleatorio de la secuencia.**
  - **random.choices()→Devuelve una lista de elementos aleatorios.**
  - **random.sample()→Devuelve una lista de elementos únicos aleatorios.**
  - La clase Random → Permite crear un generador de números aleatorios.

MÓDULO MATEMÁTICO - NUMPY

- NumPy →Biblioteca para el manejo de vectores y matrices de grandes dimensiones
- https://numpy.org/
- pip install numpy

MÓDULO MATEMÁTICO - NUMPY

- NumPy
  - Rápido
  - Eficiente
  - Maneja vectores y matrices
  - Realiza cálculos estadísticos
  - Tipos de datos:

    https://numpy.org/doc/stable/user/basics.types.html

MÓDULO MATEMÁTICO - NUMPY

- NumPy
  - import numpy as np
  - np.zeros(*n, dtype=tipo)*→Crea un array con n ceros.
  - np.ones(n, *dtype=tipo*)→Crea un array con n unos.
  - np.full(n,d, *dtype=tipo*)→Crea una array con n elementos con valor d.
  - np.array(*iterador, dtype=tipo*)→Crea un array con los elementos del iterador.
  - np.arange(*rango, dtype=tipo*))→Crea un array con los números dentro de un rango. Rangos:
    - 10
    - 10,20
    - 10,20,2

MÓDULO MATEMÁTICO - NUMPY

- NumPy
  - np.zeros((*filas*,*columnas*),*dtype=tipo*)→Matriz
  - np.ones((*filas*,*columnas*),*dtype=tipo*)→Matriz
  - np.full((*filas,columnas,d*),*dtype=tipo*)→Matriz
  - np.linspace(*inicio,fin,n*)→Devuelve n números entre inicio y fin espaciados homogénamente.

```
>>> np.linspace(0,10,21)
array([ 0. ,  0.5,  1. ,  1.5,  2. ,  2.5,  3. ,  3.5,  4. ,  4.5,  5. ,
        5.5,  6. ,  6.5,  7. ,  7.5,  8. ,  8.5,  9. ,  9.5, 10. ])
```

MÓDULO MATEMÁTICO - NUMPY

- NumPy
  - np.random.random(dimensión)
  - np.random.normal(media, desviación, dimensión)
  - np.random.randint(lim_inferior, lim_superior, dimensión)
  - np.eye(dimensión)→Matriz identidad
  - np.empty(dimensión)→Matriz sin inicializar con valores de memoria.

MÓDULO MATEMÁTICO - NUMPY

- Atributos de los arrays:
  - ndim→Número de dimensiones
  - shape→tamaño de cada dimensión (figura)
  - size→Número de elementos
  - itemsize→Tamaño en bytes de item
  - nbytes→Tamaño en bytes de la estructura

```
>>> x
array([[0.07096621, 0.03174951, 0.55799243],
       [0.58363417, 0.35460255, 0.82353577],
       [0.68698763, 0.78995433, 0.99254654]])
>>> x.ndim
2
>>> x.shape
(3, 3)
>>> x.size
9
>>> x.itemsize
8
>>> x.nbytes
72
>>>
```

MÓDULO MATEMÁTICO - NUMPY

- Acceso a los elementos individuales:
  - A través de índices, dependiendo de las dimensiones:
    - array[i][j][z]

```
>>> x
array([[9, 3, 7],
       [8, 8, 1],
       [5, 4, 8]])
>>> x[1,1]
8
>>> x[1,1]=15
>>> x
array([[ 9,  3,  7],
       [ 8, 15,  1],
       [ 5,  4,  8]])
```

MÓDULO MATEMÁTICO - NUMPY

- Acceso a los elementos individuales:
  - Admite *slicing*:

```
>>> x
array([0.42140097, 0.1333651 , 0.66494606, 0.15654996, 0.13042769,
       0.78527268, 0.29186879, 0.05090091, 0.51401955, 0.85277874])
>>> x[1:3]
array([0.1333651 , 0.66494606])
>>> x[::-1]
array([0.85277874, 0.51401955, 0.05090091, 0.29186879, 0.78527268,
       0.13042769, 0.15654996, 0.66494606, 0.1333651 , 0.42140097])
>>> x[::2]
array([0.42140097, 0.66494606, 0.13042769, 0.29186879, 0.51401955])
```

MÓDULO MATEMÁTICO - NUMPY

- Acceso a los elementos individuales:
  - Admite *slicing*:

```
>>> x
array([[7, 9, 5],
       [3, 8, 7],
       [2, 6, 4]])
>>> x[::-1,::-1]
array([[4, 6, 2],
       [7, 8, 3],
       [5, 9, 7]])
>>> x[1,:]
array([3, 8, 7])
>>> x[:,1]
array([9, 8, 6])
>>> x[0:2,0:2]
array([[7, 9],
       [3, 8]])
```

MÓDULO MATEMÁTICO - NUMPY

- Acceso a los elementos individuales:
  - El *slicing* genera referencias (no copias):

```
>>> x
array([[7, 9, 5],
       [3, 8, 7],
       [2, 6, 4]])
>>> y = x[0:2,0:2]
>>> y
array([[7, 9],
       [3, 8]])
>>> y[0,0]=15
>>> y
array([[15,  9],
       [ 3,  8]])
>>> x
array([[15,  9,  5],
       [ 3,  8,  7],
       [ 2,  6,  4]])
```

MÓDULO MATEMÁTICO - NUMPY

- Acceso a los elementos individuales:
  - Creación de copias.
    - Función copy()

```
>>> x
array([[15,  9,  5],
       [ 3,  8,  7],
       [ 2,  6,  4]])
>>> y = x[0:2,0:2].copy()
>>> y
array([[15,  9],
       [ 3,  8]])
>>> y[0,0]=18
>>> y
array([[18,  9],
       [ 3,  8]])
>>> x
array([[15,  9,  5],
       [ 3,  8,  7],
       [ 2,  6,  4]])
```

MÓDULO MATEMÁTICO - NUMPY

- NumPy
  - Otras funciones:
    - np.reshape()→Redimensionamiento.
    - np.concatenate()→Concatenación.
    - np.split()→Disgregación.
    - np.vsplit()→Disgregación.
    - np.hsplit()→Disgregación.

MÓDULO MATEMÁTICO - NUMPY

- NumPy
  - UFuncs→Funciones universales:
    - Incorporan operaciones sobre las matrices y vectores.
    - https://numpy.org/doc/stable/reference/ufuncs.html
      - Operadores
      - Valores absolutos
        - np.abs
        - np.absolute
      - Funciones trigonométricas
      - Funciones de agregación

MÓDULO MATEMÁTICO - NUMPY

- NumPy
  - UFuncs→Funciones universales:
    - Redirección de salida:
      - Ejemplo:

```
>>> x
array([[15,  9,  5],
       [ 3,  8,  7],
       [ 2,  6,  4]])
>>> y = np.empty([3,3])
>>> np.multiply(x, 5, out=y)
array([[75., 45., 25.],
       [15., 40., 35.],
       [10., 30., 20.]])
```

MÓDULO MATEMÁTICO - NUMPY

- NumPy
  - Operadores y funciones:
    - Aplicados array + número → aplican a cada elemento del array.
    - Aplicados a dos arrays → aplican entre elementos de los arrays.
      - Exigen equivalencia en longitud.

Operator	Equivalent ufunc	Description
+	np.add	Addition (e.g., 1 + 1 = 2)
-	np.subtract	Subtraction (e.g., 3 - 2 = 1)
-	np.negative	Unary negation (e.g., -2)
*	np.multiply	Multiplication (e.g., 2 * 3 = 6)
/	np.divide	Division (e.g., 3 / 2 = 1.5)
//	np.floor_divide	Floor division (e.g., 3 // 2 = 1)
**	np.power	Exponentiation (e.g., 2 ** 3 = 8)
%	np.mod	Modulus/remainder (e.g., 9 % 4 = 1)

https://www.oreilly.com/library/view/python-data-science/9781491912126/

MÓDULO MATEMÁTICO - NUMPY

- NumPy
  - Funciones de agregación:

Table 2-3. Aggregation functions available in NumPy

Function Name	NaN-safe Version	Description
`np.sum`	`np.nansum`	Compute sum of elements
`np.prod`	`np.nanprod`	Compute product of elements
`np.mean`	`np.nanmean`	Compute median of elements
`np.std`	`np.nanstd`	Compute standard deviation
`np.var`	`np.nanvar`	Compute variance
`np.min`	`np.nanmin`	Find minimum value
`np.max`	`np.nanmax`	Find maximum value
`np.argmin`	`np.nanargmin`	Find index of minimum value
`np.argmax`	`np.nanargmax`	Find index of maximum value
`np.median`	`np.nanmedian`	Compute median of elements
`np.percentile`	`np.nanpercentile`	Compute rank-based statistics of elements
`np.any`	N/A	Evaluate whether any elements are true
`np.all`	N/A	Evaluate whether all elements are true

https://www.oreilly.com/library/view/python-data-science/9781491912126/

MÓDULO MATEMÁTICO - NUMPY

- NumPy
  - Funciones de ordenación:
    - np.sort(matriz)→Ordena contenidos.
    - np.argsort(matriz)→Ordena índices.
    - np.sort(matriz, x=0)→Ordena matriz por columnas.
    - np.sort(matriz, x=1)→Ordena matriz por filas.