PYTHON

Módulo 10 – Módulo de sistema y de la plataforma

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

Generic Operating System Services
(Servicios genéricos del sistema operativo)

https://docs.python.org/es/3/library/allos.html

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulo os:
  - environ→Información del entorno del sistema operativo.
  - getcwd()→Obtención del directorio actual.
  - chdir(path)→Cambio de directorio actual.
  - chroot(path)→Cambia el directorio raíz del proceso actual.(Solo Unix)
  - chmod(path, permisos)→Cambia los permisos de un archivo o directorio.
  - mkdir() y mkdirs()→Creación de directorios.

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulo os:
  - remove()→Eliminación de un archivo.
  - rmdir()→Eliminación de un directorio.
  - removedirs()→Eliminacion recursiva de directorios.
  - rename()→Renombrado de archivos
  - listdir(path)→Obtiene los elementos contenidos en el path.
  - replace()→Reemplaza el nombre de un archivo o directorio.

Módulo de sistema y de la plataforma

- Módulo io:
  - Clase io.IOBASE (clase abstracta):
    - open()→Abre un fichero.
      - fichero = io.open("fichero.txt","w")#Igual que función open
    - close()→Cierra un fichero.
    - closed→Determina si un fichero está cerrado
    - flush()→Vaciado de buffers
    - readable()→Indica si el recurso se puede leer.
    - readline()→Lee un línea.
    - readlines()→Lee todas las líneas y devuelve una lista.
    - seek()→Permite posicionarse en el fichero.
    - writable()→Indica si el fichero se puede escribir.

Módulo de sistema y de la plataforma

- Módulo io:
  - Clase io.RawIOBase (hereda de IOBase, ficheros binarios):
    - read()
    - readAll()
    - readinto()
    - write()
  - Clase io.BufferedIOBase (hereda de IOBase, ficheros binarios, con buffer):
    - read()
    - readinto()
    - write()

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulos datetime (Data Types), calendar (Data Types) y time (Generic Operating System Services).
- Módulo datetime:
  - https://docs.python.org/3/library/datetime.html
    - Clases:
      - datetime.date → Fecha (atributos: year, month y day)
      - datetime.time → Hora (atributos: hour, minute, second, microsecond)
      - datetime.datetime → Fecha y hora
      - **datetime.timedelta**(days=0, seconds=0, microseconds=0, milliseconds=0, minutes=0, hours=0, weeks=0) → Representa un incremento temporal.
      - **datetime.date.today()→Obtiene la fecha actual (objeto date)**
      - **datetime.datetime.now()→Obtiene la fecha y hora actual (objeto datetime)**

Módulo de sistema y de la plataforma

- Módulos datetime, calendar y time
- datetime:
  - Operaciones: + y -
    - >>> offset = datetime.timedelta(days=10)
    - >>> offset
    - datetime.timedelta(days=10)
    - >>> fecha_fin = datetime.date.today() + offset
    - >>> fecha_fin = datetime.date.today() - offset

Módulo de sistema y de la plataforma

- Módulos datetime, calendar y time
- datetime:
  - Número de días transcurridos entre dos fechas
    - >>> nacimiento = datetime.date(1971,9,3)
    - >>> hoy = datetime.date.today()
    - >>> diferencia = hoy - nacimiento
    - >>> print(diferencia.days)

Módulo de sistema y de la plataforma

- Módulos datetime, calendar y time
- datetime:
  - Fijar un periodos:
    - >>> periodo = datetime.timedelta(days=15)
    - >>> fecha_vencimiento = datetime.datetime.now() + periodo
    - >>> "Fecha vencimiento: {}-{}-{}".format(fecha_vencimiento.day, fecha_vencimiento.month, fecha_vencimiento.year)

Módulo de sistema y de la plataforma

- Módulos datetime, calendar y time
- Módulo calendar:
  - https://docs.python.org/3/library/calendar.html
  - Proporciona funcionalidad relacionada con calendarios:

```
>>> print(calendar.month(2022,2))
    February 2022
Mo Tu We Th Fr Sa Su
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28
```

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulos datetime, calendar y time
- calendar:

```
>>> print(calendar.calendar(2022))
                                  2022

      January                   February                  March
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
                1  2            1  2  3  4  5  6            1  2  3  4  5  6
 3  4  5  6  7  8  9       7  8  9 10 11 12 13       7  8  9 10 11 12 13
10 11 12 13 14 15 16      14 15 16 17 18 19 20      14 15 16 17 18 19 20
17 18 19 20 21 22 23      21 22 23 24 25 26 27      21 22 23 24 25 26 27
24 25 26 27 28 29 30      28                        28 29 30 31
31

      April                     May                       June
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
             1  2  3                        1                  1  2  3  4  5
```

Módulo de sistema y de la plataforma

- Módulos datetime, calendar y time
- calendar:
  - Clase Calendar.
    - calendario = calendar.Calendar(firstweekday=0) #0→Lunes, 6→Domingo
    - Métodos *iter????*
      - Proporciona iteradores relacionados con días de la semana, días del mes,…
    - Métodos *month????* y *year????*
      - Proporcionan secuencias de días en formato datetime.date o tupla.

Módulo de sistema y de la plataforma

- Módulos datetime, calendar y time
- calendar:
  - Clase TextCalendar.
    - Permite generar calendarios en texto plano.

```
>>> c = calendar.TextCalendar(0)
>>> print(c.formatmonth(2022,2))
    February 2022
Mo Tu We Th Fr Sa Su
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28
```

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulos datetime, calendar y time
- calendar:
  - Clase HTMLCalendar.
    - Permite generar calendarios en HTML.

```
>>> c = calendar.HTMLCalendar(0)
>>> print(c.formatmonth(2022,2))
<table border="0" cellpadding="0" cellspacing="0" class="month">
<tr><th colspan="7" class="month">February 2022</th></tr>
<tr><th class="mon">Mon</th><th class="tue">Tue</th><th
class="wed">Wed</th><th class="thu">Thu</th><th
class="fri">Fri</th><th class="sat">Sat</th><th
class="sun">Sun</th></tr>
<tr><td class="noday"> </td><td class="tue">
```

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulos datetime, calendar y time

- calendar:
  - calendar.monthrange() → Indica en qué día de la semana empieza el mes (0-6) y el número de días.

```
>>> calendar.monthrange(2022, 2)
(1, 28)
```

  - calendar.prmonth(), calendar.prcal() → Muestra por consola calendarios de meses o años.
  - calendar.monthrange() → Genera una tupla con los días del año y mes indicado.

Módulo de sistema y de la plataforma

- Módulos datetime, calendar y time
- calendar:
  - calendar.day_name → Nombre del día de la semana
  - calendar.day_abbr → Nombre del día de la semana abreviado
  - calendar.month_name → Nombre del mes¶
  - calendar.month_abbr→Nombre del mes abreviado
- Constantes
  - calendar.MONDAY
  - calendar.TUESDAY
  - calendar.WEDNESDAY
  - calendar.THURSDAY
  - calendar.FRIDAY
  - calendar.SATURDAY
  - calendar.SUNDAY

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulos datetime (Data Types), calendar (Data Types) y time (Generic Operating System Services).
- Módulo time:
  - https://docs.python.org/3/library/time.html
  - gmtime(0)→Devuelve una estructura con la información referente al momento de inicio del tiempo en la plataforma de ejecución. En Unix y Windows el 1/1/1970 a las 0 horas UTC.
    - time.gmtime(0).tm_year → 1970
  - ctime(*segundos*)→Obtiene la fecha resultante de sumar los segundos enviados como parámetro a la fecha de inicio gmtime.
  - sleep(*segundos*)→Detiene la ejecución del programa.
  - localtime()→Estructura con la fecha actual.

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulos datetime (Data Types), calendar (Data Types) y time (Generic Operating System Services).
- Módulo time:
  - mktime*(estructura_de_fecha)*→a partir de una estructura de fecha calcula los segundos transcurridos desde la fecha inicial.

```
>>> ahora = time.localtime()
>>> time.mktime(ahora)
1644487565.0
```

  - strftime()→Permite dar formato a las fechas.

```
>>> ahora = time.localtime()
>>> time.strftime("%d/%b/%Y",ahora)
'10/Feb/2022'
```

  - strptime()→Convierte string a estructura de fecha.

```
>>> fecha = "Tue, 10 Feb 2022 08:15:28"
>>> estructura = time.strptime(fecha, "%a, %d %b %Y %H:%M:%S")
>>> print(estructura)
time.struct_time(tm_year=2022, tm_mon=2, tm_mday=10, tm_hour=8, tm_min=15, tm_sec=28, tm_wday=1,
tm_yday=41, tm_isdst=-1)
```

https://docs.python.org/3/library/datetime.html#strftime-strptime-behavior

Módulo de sistema y de la plataforma

- Módulos datetime (Data Types), calendar (Data Types) y time (Generic Operating System Services).

- Módulo time:
  - clase time.struct_time.

Index	Attribute	Values
0	`tm_year`	(for example, 1993)
1	`tm_mon`	range [1, 12]
2	`tm_mday`	range [1, 31]
3	`tm_hour`	range [0, 23]
4	`tm_min`	range [0, 59]
5	`tm_sec`	range [0, 61]; see **(2)** in `strftime()` description
6	`tm_wday`	range [0, 6], Monday is 0
7	`tm_yday`	range [1, 366]
8	`tm_isdst`	0, 1 or -1; see below
N/A	`tm_zone`	abbreviation of timezone name
N/A	`tm_gmtoff`	offset east of UTC in seconds

Módulo de sistema y de la plataforma

- Logs:
  - Módulo logging
  - Configuración:
    - Establecer nivel de filtrado de logs:
      - logging.getLogger().setLevel(logging.*level*)
      - *level:*

Nivel	Valor numérico
CRITICAL	50
ERROR	40
WARNING	30
INFO	20
DEBUG	10
NOTSET	0

https://docs.python.org/es/3/library/logging.html

Módulo de sistema y de la plataforma

- Logs:
  - Módulo logging
  - Uso:

```
logging.debug("Depuración")
logging.info("Información")
logging.warning("Advertencia")
logging.error("Error")
logging.critical("Error crítico")
```

```
logging.exception("Excepción")→Nivel ERROR. Sólo para excepciones.
```

Módulo de sistema y de la plataforma

- Logs:
  - Módulo logging
  - Uso Exception:

```
try:
    b=5
    c=0
    a = b/c
except ZeroDivisionError as error:
    logging.exception(error)
```

ERROR:root:division by zero

Traceback (most recent call last):

File "h:_ProyectosPython\borrar.py", line 15, in funcion

a = b/c

ZeroDivisionError: division by zero

Módulo de sistema y de la plataforma

- Logs:
  - Módulo logging
  - Configuración salida a fichero:

```
logging.basicConfig(filename="fichero_salida.log",
    filemode='a',
    format='%(asctime)s,%(msecs)d %(name)s %(levelname)s %(message)s',
    datefmt='%H:%M:%S',
    level=logging.DEBUG)
```

  - Atributos:
    - https://docs.python.org/3/library/logging.html#logrecord-attributes

Módulo de sistema y de la plataforma

- Módulo getpass
  - getpass(prompt)→Permite introducir por terminal una contraseña y que esta esté oculta.
  - getuser()→Proporciona el usuario del sistema.

Módulo de sistema y de la plataforma

Acceso a archivos y directorios

https://docs.python.org/es/3/library/filesys.html

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulo os y os.path
  - https://docs.python.org/3/library/os.html
  - os.getlogin()→Usuario del sistema
  - os.getcwd()→Obtiene el directorio actual
  - os.mkdir("Nueva Carpeta") →Crea una carpeta
  - os.makedirs("Principal/Secundaria") → Crea una estructura de carpetas
  - os.rmdir("carpeta")→Borra una carpeta (debe estar vacía)
  - os.rename("antiguo.txt","nuevo.txt")
  - os.renames("antiguo.txt","nuevo.txt")

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulo os y os.path
  - os.removedirs("uno/otro/ultimo")→Borra desde la carpeta indicada hacia arriba hasta encontrar una carpeta con contenido
  - os.listdir(*path*)→Proporciona una lista con el contenido de una carpeta.
  - os.listdir(os.getcwd())→Proporciona una lista con el contenido de la carpeta actual.

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulo os y os.path
  - https://docs.python.org/3/library/os.path.html
  - os.path.exists(("d:/showMsg.txt")→Determina si existe un path.
  - os.path.getsize("d:/showMsg.txt")→Obtiene el tamaño en bytes de un path
  - os.path.isfile("d:/showMsg.txt")→Indica si un path es un fichero.
  - os.path.isdir(“d:/”)→Indica si un path es un directorio.

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulo shutil
  - import shutil
  - Proporciona funciones de alto nivel para manejo de ficheros.
    - shutil.rmtree("_borrar")→Borra un directorio y **TODO** su contenido.
    - shutil.copyfile()→Copia un fichero
    - shutil.copy()→Copia un fichero o carpeta
    - shutil.copy2()→Igual que copy, intentando copiar metadatos.
    - shutil.copytree()→Copia un directorio y su contenido.
    - shutil.move()→Mueve un archivo o directorio con su contenido.

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Ejemplo: recorrer de forma recursiva un path:

```
import os
def get_contenido(ruta):
    try:
        contenido = os.listdir(ruta)
        for c in contenido:
            c=ruta+"/"+c
            if os.path.isdir(c):
                get_contenido(c)
            else:
                print(c)
    except PermissionError:
        pass
get_contenido("D:/psp")
```

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

Servicios en tiempo de ejecución de Python

https://docs.python.org/es/3.10/library/python.html

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

- Módulo sys
  - sys.exit(codigo)→Salida del programa.
  - sys.getwindowsversion()→Obtención de información del sistema operativo.
  - sys.version → Versión del intérprete de Python.

MÓDULO DE SISTEMA Y DE LA PLATAFORMA

Técnicas y otros módulos

Módulo de sistema y de la plataforma

- Toma de tiempos:
- módulo time:
  - time.time() →Tiempo actual.

  - start = time.time()
  - …proceso…
  - end = time.time()
  - print(f"Tiempo empleado: {end – start}")

MÓDULO DE DEBUGGING AND PROFILING

- Toma de tiempos

- módulo timeit:
  - timeit(setup, stmt, number)→Calcula el tiempo que tarda en ejecutarse una sentencia.
    - setup→Por defecto pass, código que se necesita ejecutar antes de realizar el cálculo.
    - stmt→Sentencia.
    - number→número de veces que se tiene que ejecutar el código.

```
import timeit

setup_code = "lista = list(range(0,100))"

code = """
total = 0
for i in lista:
    total = total + i
"""

print(timeit.timeit(stmt=code, setup=setup_code, number=10000))
```

MÓDULO CONCURRENT EXECUTION

- Módulo subprocess
  - Permite la creación de nuevos procesos.
    - run()→Permite ejecutar un proceso. Devuelve un objeto subprocess.CompletedProcess.
      - args
      - returncode
      - stdout (con *capture_output=True*)
      - stderr

```
cp = subprocess.run(“programa.exe", capture_output=True, text=True)
cp = subprocess.run("python programa.py", capture_output=True, text=True)
```