**Módulo de 14:**

**Manejo de fechas y horas datetime, date, time, etc:**

* Datetime
* .year,.month,.day,.hour..minute,.second:
* strftime
* strptime
* Timedelta
* Time
* Date

# **Manejo de fechas y horas datetime, date, time, etc:**

**Manejo fechas y horas**

El Módulo datetime para trabajar con fechas y horas, sobre un formato ISO 8601. (International Organization for Standardization). Esto permite poder enviar y recibir fechas y horarios entre varios programas, lenguajes, bases de datos, paginas web, etc

Datetime genera un objeto datetime con varios métodos y atributos que permiten sacar la diferencia entre dos objetos datetime (timedelta) transformar números o string a datatime o viceversa. Ademas permite manipular el formato de entrada y salida ubicando año,mes,día(estándar ISO) o día,mes,año (estándar español) o mes,día,año (estándar ingles)

|  |
| --- |
| Métodos y atributos:  .datetime()  .now()  .timedelta()  .date()  .fromtimestamp()  .time()  .strptime()  .utcnow()  .today()  .utcfromtimestamp()  .tzinfo()  .MAXYEAR  .strftime()  .MINYEAR  .tzname()  .timezone()  .combine()  .isoformat()  .replace()  .html() |

·

|  |
| --- |
| ISO (Organización Internacional de Normalización) es una organización internacional independiente que desarrolla y promueve estándares en diferentes áreas. En el contexto de la gestión de fechas y horas, la norma ISO 8601 es un estándar internacional que establece un formato para la representación de fechas, horas y combinaciones de ambas.  El formato ISO 8601 se utiliza ampliamente en todo el mundo y tiene varias ventajas. Algunas de las características clave del formato ISO 8601 son:   1. Orden lógico: La fecha se representa en el formato "año-mes-día", lo cual tiene sentido tanto desde una perspectiva lógica como de ordenamiento. 2. Separadores: Se utilizan guiones (-) como separadores entre los componentes de la fecha y dos puntos (:) para separar las partes de la hora. 3. Precisión: El formato permite representar fechas y horas con diferentes niveles de precisión, desde años y meses hasta segundos y fracciones de segundo. 4. Formato extendido: Además del formato básico, el estándar ISO 8601 también define un formato extendido que permite incluir información adicional, como la zona horaria. 5. Consistencia internacional: Al ser un estándar internacional ampliamente adoptado, el formato ISO 8601 facilita la comunicación y el intercambio de información entre diferentes sistemas y países.   Un ejemplo de una fecha y hora en formato ISO 8601 sería "2023-06-22T21:30:45.123456", donde "2023-06-22" representa la fecha en formato "año-mes-día" y "21:30:45.123456" representa la hora en formato "hora:minuto:segundo.fracción de segundo".  El uso del formato ISO 8601 es recomendado en diversas aplicaciones y lenguajes de programación, ya que proporciona una representación clara y consistente de las fechas y horas, evitando ambigüedades y facilitando el procesamiento y la interoperabilidad de la información temporal. |

·

|  |
| --- |
| import datetime  # Obtener la fecha y hora actual  fecha\_hora\_actual = datetime.datetime.now()  #----------------------------------------------------------------------------  # Obtener la fecha actual en formato ISO  fecha\_actual\_iso = fecha\_hora\_actual.date().isoformat()  print(f"Fecha actual (ISO):{ fecha\_actual\_iso }")  #----------------------------------------------------------------------------  # Obtener la hora actual en formato ISO  hora\_actual\_iso = fecha\_hora\_actual.time().isoformat()  print(f"Hora actual (ISO):{ hora\_actual\_iso }")  #----------------------------------------------------------------------------  # Obtener la fecha y hora actual en formato ISO  fecha\_hora\_actual\_iso = fecha\_hora\_actual.isoformat()  print(f"Fecha y hora actual (ISO):{ fecha\_hora\_actual\_iso }") |

·

Fecha actual (ISO):

YYYY año x 4 caracteres,

guion

MM- mes x 2 caracteres,

- guion

DD dia x 2 caracteres

Hora actual (ISO):

17:58:19.553302

HH hora x 4 caracteres,

: dos puntos

MM- minutos x 2 caracteres,

: dos puntos

SS segundo x 2 caracteres

. punto

fracción de

segundo 1/1000000 segundo x 6 caracteres

Fecha y hora actual (ISO):

YYYY año x 4 caracteres,

guion

MM- mes x 2 caracteres,

- guion

DD dia x 2 caracteres

T Caracter T

HH hora x 4 caracteres,

: dos puntos

MM- minutos x 2 caracteres,

: dos puntos

SS segundo x 2 caracteres

. punto

fracción de

segundo 1/1000000 segundo x 6 caracteres

|  |
| --- |
| Fecha actual (ISO): YYYY – MM - DD  Hora actual (ISO): HH : MM : SS . 123456  Fecha y hora actual (ISO): YYYY – MM – DD T HH : MM : SS . 123456 |

Con datetime.datetime.now() obtenea la fecha y hora actual.

El método date() obtendrás solo la fecha

El método time() obtendrás solo el horario.

Fecha específica utilizando la clase datetime.date y pasando los valores de año, mes y día como argumentos.

·

**Date:**

·

|  |
| --- |
| import datetime  year = 1492  month = 10  day = 12  # Colon visualiza América (aunque creía que era india)  fecha\_especifica = datetime.date(year, month, day)  print(" Colon visualiza América:", fecha\_especifica) |

·

|  |
| --- |
| Colon visualiza América: 1492-10-12 |

·

**Datetime:**

|  |
| --- |
| import datetime  year = 1969  month = 8  day = 20  hour = 2  minite = 56  second = 0  fecha\_y\_horario\_especifico = datetime.datetime (year, month, day ,hour,minite,second)  print(f""" El comandante Armstrong fue el primer ser humano que pisó la superficie del satélite terrestre al sur del Mar de la Tranquilidad (Mare Tranquillitatis) el {fecha\_y\_horario\_especifico } """) |

·

|  |
| --- |
| El comandante Armstrong fue el primer ser humano que pisó la superficie del satélite terrestre al sur del Mar de la Tranquilidad (Mare Tranquillitatis) el 1969-08-20 02:56:00 |

·

Obtener componentes individuales de una fecha/hora:

Puedes acceder a los componentes individuales de una fecha/hora, como el año, mes, día, hora, minuto y segundo utilizando los atributos de los objetos datetime.

**.year,.month,.day,.hour..minute,.second:**

|  |
| --- |
| import datetime  year = 1969  month = 8  day = 20  hour = 2  minite = 56  second = 0  hora = datetime.datetime (year, month, day ,hour,minite,second)  print("Fecha desde el primer alunizaje :")  print("Año:", fecha\_hora.year)  print("Mes:", fecha\_hora.month)  print("Día:", fecha\_hora.day)  print("Hora:", fecha\_hora.hour)  print("Minuto:", fecha\_hora.minute)  print("Segundo:", fecha\_hora.second) |

·

|  |
| --- |
| Fecha desde el primer alunizaje :  Año: 1969  Mes: 8  Día: 20  Hora: 2  Minuto: 56  Segundo: 0 |

·

|  |  |  |
| --- | --- | --- |
| La siguiente tabla muestra todos los códigos de formato que puede utilizar. | | |
| Directiva | Sentido | Ejemplo |
| %a  %A  %w  %d  %-d  %b  %B  %m  %-m  %y  %-y  %Y  %j  %-j  %U  %W | Nombre abreviado del día de la semana.  Nombre completo del día de la semana.  Día de la semana como un número decimal.  Día del mes como dos números decimales con ceros.  Día del mes como un/dos números decimales.  Nombre del mes abreviado.  Nombre del mes completo.  Mes como dos números decimales con ceros.  Mes como un/dos números decimales.  Año sin siglo como dos números decimales con ceros.  Año sin siglo como un/dos números decimales.  Año con siglo como cuatro números decimales.  Día del año como tres números decimales con ceros.  Día del año como uno a tres números decimales.  Número de semana del año (domingo como primer día de la  semana). Todos los días de un año que preceden al primer domingo se consideran en la semana  Número de semana del año (lunes como primer día de la  semana). Todos los días de un año que preceden al primer lunes se consideran en la semana | sun, mon,…sat  sunday, monday….saturday  0, 1, ..., 6  01, 02, ...,31  1, 2, ..., 31  ene, feb, ...,dic  Enero febrero, ...  01, 02, ..., 12  1, 2, ..., 12  00, 01, ..., 99  0, 1, ..., 99  1999,2000,2025  001, 002, ..., 366  1, 2, ..., 366  0.00, 01, ..., 53  0.00, 01, ..., 53 |
| %H  %-H  %I  %-I  %p  %M  %-M  %S  %-S  %f  %z | Hora como un número decimal con ceros.  Hora (0/23-24)como un/dos números decimales.  Hora (0/12) como un/dos números decimales con ceros.  Hora (reloj de 12 horas) como un número decimal.  AM o PM de la horas.  Minuto como dos números decimales con ceros.  Minuto como un/dos números decimales.  Segundo como dos números decimales con ceros.  Segundo comoun/dos números decimales .  Microsegundo como números decimales, ceros a laizquierda.  Desplazamiento UTC en la forma +HHMM o -HHMM. | 00, 01, ..., 23  0, 1, ..., 23  01, 02, ..., 12  1, 2, ... 12  AM PM  00, 01, ..., 59  0, 1, ..., 59  00, 01, ..., 59  0, 1, ..., 59  000000 – 999999 |
| %Z  %c  %x | Nombre de la zona horaria.  Representación de fecha y hora adecuada de la configuración regional.  Representación de fecha adecuada de la configuración reg.  %XRepresentación de tiempo adecuada de la configuración reg. | Lun 30 sep 07:06:05 2013  09/30/13  07:06:05 |

·

**strftime:**

|  |
| --- |
| Permite que un objeto datetime (fecha/horario) sea cambiado a un objeto string. Nota: se necesita proporcionar el formato (la estructura y orden) de salida de los argumentos en el string |

Tomo el objeto datetime y lo pasamos con **strftime** a string con el formato "%Y-%m-%d %H:%M:%S":

%Y año 4 caracteres numéricos

- separador guion

%m mes 2 caracteres numéricos

- separador guion

%m dia 2 caracteres numéricos

‘ ‘ separador espacio

%H hora 2 caracteres numéricos

: separador dos puntos

%M minuto 2 caracteres numéricos

: separador dos puntos

%S Segundo 2 caracteres numéricos:

|  |
| --- |
| import datetime  fecha\_horario = datetime.datetime.now()  fecha\_string = fecha\_horario.strftime("%Y-%m-%d %H:%M:%S")  print (f"Fecha string formateada: {fecha\_string} \n{type(fecha\_string)=} " ) |

·

|  |
| --- |
| Fecha string formateada: 2060-06-01 01:01:01  type(fecha\_string)=<class 'str'> |

·

|  |
| --- |
| import datetime  # Asigna formato de ejemplo1  formato1 = '%a %d %b %Y %H:%M:%S'  # Asigna formato de ejemplo2  formato2 = '%d-%m-%y %I:%m %p'  hoy = datetime.date.today() # Asigna fecha-hora  # Muestra fecha-hora según ISO 8601  print("Fecha en formato ISO 8601:", hoy)  # Aplica formato ejemplo1  cadena1 = hoy.strftime(formato1)  # Aplica formato ejemplo2  cadena2 = hoy.strftime(formato2)  # Muestra fecha-hora según ejemplo1  print("Formato1: '%a %d %b %Y %H:%M:%S'", cadena1)  # Muestra fecha-hora según ejemplo2  print("Formato2: '%d-%m-%y %I:%m %p'", cadena2) |

·

|  |
| --- |
| Fecha en formato ISO 8601: 2060-06-01  Formato1: '%a %d %b %Y %H:%M:%S' Fri 01 Jun 2023 01:01:01  Formato2: '%d-%m-%y %I:%m %p' 01-06-50 01:01 AM |

·

**strptime:**

|  |
| --- |
| Permite que un objeto string sea cambiado a un objeto datetime (fecha/horario). Nota: se necesita proporcionar el formato (la estructura y orden) de los argumentos en el string para analizar cadena de caracteres y si es posible convertirla en un datetime |

·

La sintaxis de strptime() para convertir una cadena a objeto datetime es la siguiente:

|  |
| --- |
| from datetime import datetime  fecha\_string = "2050-06-01 12:30:56"  formato = "%Y-%m-%d %H:%M:%S"  objeto\_datetime = datetime.strptime(fecha\_string, formato)  print (f"{objeto\_datetime} %Y-%m-%d %H:%M:%S,\n {type(objeto\_datetime)=}") |

·

|  |
| --- |
| Para convertir una cadena a objeto datetime  2050-06-01 12:30:56 %Y-%m-%d %H:%M:%S,  type(objeto\_datetime)=<class 'datetime.datetime'> |

El string (fecha\_string) que representa la fecha y hora, y formato es otro string que especifica el formato de la fecha\_string (fecha de entrada).

|  |
| --- |
| El Módulo datetime ofrece muchas más funcionalidades, como comparar fechas, obtener la diferencia entre dos fechas, trabajar con zonas horarias, etc. |

**Timedelta:**

Para sumar o restar días, semanas, meses o años con objetos de clase datetime.

|  |
| --- |
| import datetime  fecha\_actual = datetime.date.today()  un\_dia = datetime.timedelta(days=1)  fecha\_mañana = fecha\_actual + un\_dia  fecha\_pasada = fecha\_actual - un\_dia  print("Fecha actual:", fecha\_actual)  print("Fecha mañana:", fecha\_mañana)  print("Fecha pasada:", fecha\_pasada) |

·

|  |
| --- |
| Fecha actual: 2050-01-01  Fecha mañana: 2050-01-02  Fecha pasada: 2049-12-31 |

·

|  |
| --- |
| import datetime  year = 1969  month = 8  day = 20  fecha\_hora = datetime.date (year, month, day )  fecha\_actual = datetime.date.today()  delta=(fecha\_actual-fecha\_hora)  print("Fecha desde el primer alunizaje a hoy :", delta) |

··

|  |
| --- |
| Fecha desde el primer alunizaje a hoy : 999999 days, 0:00:00 |

|  |
| --- |
| from datetime import timedelta, date  print (f"{(date.today() - timedelta(10))=}") |

··

|  |
| --- |
| (date.today() - timedelta(10))=datetime.date(2050,06, 20) |

|  |
| --- |
| from datetime import date  def days\_diff(start, end):  return (end - start).days  print(f"{date(2000, 1, 1)} - {date(2020, 10, 28)} = {days\_diff(date(2020, 1, 1), date(2020, 10, 28))} dias") |

·

|  |
| --- |
| 2000-01-01 - 2020-10-28 = 301 dias |

·

|  |
| --- |
| from datetime import timedelta  t1 = timedelta(weeks = 2, days = 5, hours = 1, seconds = 33)  t2 = timedelta(days = 4, hours = 11, minutes = 4, seconds = 54)  days\_diff = t1 - t2  print(f"{days\_diff =}") |

·

|  |
| --- |
| days\_diff =datetime.timedelta(days=14, seconds=50139) |

·

|  |
| --- |
| from datetime import timedelta  t1 = timedelta(seconds = 33)  t2 = timedelta(seconds = 54)  t3 = t1 - t2  print(f"{days\_diff =}")  print(f"{abs(days\_diff) =}") |

·

|  |
| --- |
| days\_diff =datetime.timedelta(days=-1, seconds=86379)  abs(days\_diff) =datetime.timedelta(seconds=21) |

·

**Time:**

|  |  |
| --- | --- |
|  | Función con descripción |
| time.altzone  time.asctime  ([tupletime])  time.clock( )  time.ctime ([segs])  time.gmtime  ([segs])  time.localtime  ([segs])  time.mktime  (tupletime)  time.sleep (segundos)  time.strftime  (fmt[,tupletime])  time.strptime  (str,fmt=  '%a %b %d  %H:%M:%S %Y')  time.tiempo( )  time.tzset() | El desplazamiento de la zona horaria DST local, en segundos al oeste de UTC, si se define uno. Esto es negativo si la zona horaria DST local está al este de UTC (como en Europa occidental, incluido el Reino Unido). Solo use esto si la luz del día es distinta de cero.   Acepta una tupla de tiempo y devuelve una cadena legible de 24 caracteres como 'Tue Dec 11 18:07:14 2008'.  Devuelve el tiempo de CPU actual como un número de segundos de coma flotante. Para medir los costos computacionales de diferentes enfoques, el valor de time.clock es más útil que el de time.time().   Como asctime(localtime(secs)) y sin argumentos es como asctime()   Acepta un instante expresado en segundos desde la época y devuelve una tupla de tiempo t con la hora UTC. Nota: t.tm\_isdst siempre es 0  Acepta un instante expresado en segundos desde la época y devuelve una tupla de tiempo t con la hora local (t.tm\_isdst es 0 o 1, dependiendo de si DST se aplica a segundos instantáneos por reglas locales).  Acepta un instante expresado como una tupla de tiempo en hora local y devuelve un valor de coma flotante con el instante expresado en segundos desde la época.  Suspende el hilo de llamada durante segundos segundos.  Acepta un instante expresado como una tupla de tiempo en la hora local y devuelve una cadena que representa el instante especificado por la cadena fmt.  Analiza str de acuerdo con la cadena de formato fmt y devuelve el instante en formato de tupla de tiempo.  Devuelve el instante de tiempo actual, un número de segundos en punto flotante desde la época.  Restablece las reglas de conversión de tiempo utilizadas por las rutinas de la biblioteca. La variable de entorno TZ especifica cómo se hace esto.  0 <= horas < 24 0 <= minutos < 60 0 <= segundos < 60 0 <= microsegundos < 1000000 |

·

|  |
| --- |
| from datetime import time  mintime = time.min  print("Min Time supported", mintime)  maxtime = time.max  print("Max Time supported", maxtime) |

·

|  |
| --- |
| Min Time supported 00:00:00  Max Time supported 23:59:59.999999 |

·

|  |
| --- |
| from datetime import time  hora = time(23,59,59,9999)  print("Hora:", hora.hour)  print("Minutos:", hora.minute)  print("Segundos:", hora.second)  print("Microsegundos:", hora.microsecond) |

·

|  |
| --- |
| Hora: 23  Minutos: 59  Segundos: 59  Microsegundos: 9999 |

·

|  |
| --- |
| from datetime import time  hora = time(23,59,59,9999)  # convierto la hora de data a string  string = hora.isoformat()  print("formato texto iso", string)  print(f"{type(string)=}")  print("hora Original:", hora)  hora = hora.replace(hour = 13, second = 12)  print("New Time:", hora)  print ("ver archivo strftime y strptime")  Ftime = hora.strftime("%I:%M %p")  print("Formatted time", Ftime) |

·

|  |
| --- |
| formato texto iso 23:59:59.009999  type(string)=<class 'str'>  hora Original: 23:59:59.009999  New Time: 13:59:12.009999  ver archivo strftime y strptime  Formatted time 01:59 PM |

·

**Date:**

|  |
| --- |
| from datetime import date  mindate = date.min  print("Min fecha suportado", mindate)  maxdate = date.max  print("Max fecha suportado", maxdate) |

·

|  |
| --- |
| Min Time supported 00:00:00  Max Time supported 23:59:59.999999 |

·

|  |
| --- |
| from datetime import date  fecha = date(2025, 12, 31)  print("Año:", fecha.year)  print("Mes:", fecha.month)  print("Día:", fecha.day) |

·

|  |
| --- |
| Año: 2025  Mes: 12  Día: 31 |

·

|  |
| --- |
| from datetime import date  hoy = date.today()  print("hoy", hoy)  # convierto la fecha de data a string  string = date.isoformat(hoy)  print("formato texto iso", string)  print(f"{type(string)=}")  print("dia semana usando weekday():", hoy.weekday())  print("dia semana usando isoweekday():", hoy.isoweekday())  # proleptic Gregorian ordinal  print("proleptic Gregorian ordinal:", hoy.toordinal())  # date from the ordinal  print("Date desde ordinal", date.fromordinal(720521)) |

·

|  |
| --- |
| hoy 2023-06-23  formato texto iso 2023-06-23  type(string)=<class 'str'>  dia semana usando weekday(): 4  dia semana usando isoweekday(): 5  proleptic Gregorian ordinal: 738694  Date desde ordinal 1973-09-21 |

·