



Python має два основних модулі для роботи з датою та часом:

Модуль time

Використовується для роботи з часом у секундах від епохи (Unix time).

Представляє час у вигляді `struct_time`.

Забезпечує низькорівневі функції для вимірювання часу.

Дозволяє призупиняти виконання програми (`sleep`).

Модуль datetime

Високорівневий модуль для роботи з датами та часом.

Підтримує роботу з часовими поясами.

Дозволяє форматування, обчислення та маніпуляцію датами.

Використовується у більшості програм для роботи з датами.

Отримання поточного часу

```
import time  
print(time.time()) # Кількість секунд від 1 січня 1970 року
```

Перетворення у struct_time

```
print(time.localtime()) # Локальний час  
print(time.gmtime()) # Час у UTC
```

Форматування часу

```
print(time.strftime('%Y-%m-%d %H:%M:%S', time.localtime()))
```

Призупинення виконання

```
time.sleep(2) # Зупиняє виконання програми на 2 секунди
```

Отримання поточного часу

```
from datetime import datetime  
print(datetime.now()) # Локальний час  
print(datetime.utcnow()) # UTC-час
```

Форматування дати

```
dt = datetime.now()  
print(dt.strftime('%Y-%m-%d %H:%M:%S')) # Форматування у рядок
```

Робота з timedelta

```
from datetime import timedelta  
now = datetime.now()  
future_date = now + timedelta(days=5)  
print(future_date) # Дата через 5 днів
```

Перетворення рядка у дату

```
date_str = "2023-12-31 23:59:59"  
date_format = "%Y-%m-%d %H:%M:%S"  
date_obj = datetime.strptime(date_str, date_format)  
print(date_obj)
```

Додавання часового поясу до дати

```
from datetime import datetime, timezone, timedelta  
utc_time = datetime.now(timezone.utc)  
print(utc_time)
```

Конвертація у локальний час


```
local_tz = timezone(timedelta(hours=2)) # UTC+2  
local_time = utc_time.astimezone(local_tz)  
print(local_time)
```

Додавання днів, годин та хвилин

```
from datetime import datetime, timedelta  
now = datetime.now()  
future = now + timedelta(days=3, hours=5, minutes=30)  
print(future)
```

Віднімання часу

```
past = now - timedelta(weeks=2)  
print(past)
```



```
from datetime import datetime
timestamp = 1700000000
converted = datetime.fromtimestamp(timestamp)
print(converted)
ts = now.timestamp()
print(ts)
```

Zulu Time (Z)

Інша назва: Z-time, Zulu time.

Відповідає UTC+0.

Використовується у військовій, авіаційній та метеорологічній сфері.

Позначається літерою "Z" наприкінці часу.

Формат: HH:MM:SSZ або YYYY-MM-DDTHH:MM:SSZ.

Приклад: 2025-02-17T14:30:00Z.

UTC (Coordinated Universal Time) – стандартний всесвітній час.

Він є базою для всіх часових поясів (UTC±X).

У ньому немає літеральної позначки Z, але числове значення таке ж.

Наприклад, 2025-02-17 14:30:00 UTC = 2025-02-17T14:30:00Z.




ISO 8601 – це міжнародний стандарт форматування дати та часу, який може містити:

Час у UTC (2025-02-17T14:30:00Z).

Час зі зміщенням (2025-02-17T16:30:00+02:00 для Києва).

Локальний час без зазначення часового поясу (2025-02-17T14:30:00 – може бути неочевидним, якщо не вказано зсув).



```
from datetime import datetime, timezone
# Поточний час у UTC
now_utc = datetime.now(timezone.utc)
# Zulu Time (UTC + 'Z' позначка)
zulu_time = now_utc.strftime("%Y-%m-%dT%H:%M:%SZ")
# UTC Time (явний формат UTC)
utc_time = now_utc.strftime("%Y-%m-%d %H:%M:%S UTC")
# ISO 8601 (з можливістю вказати зміщення, але тут UTC)
iso_8601_utc = now_utc.isoformat()
# За замовчуванням буде 'Z' в кінці
# Вивід результатів
print(f"Zulu Time: {zulu_time}")
print(f"UTC Time: {utc_time}")
print(f"ISO 8601 UTC: {iso_8601_utc}")
```




```
from datetime import date
```

```
today = date.today()
```

```
print(today) # 2025-02-17
```

```
custom_date = date(2025, 5, 20) print(custom_date) # 2025-05-20
```



```
from datetime import time  
t = time(14, 30, 15) # 14:30:15  
print(t) # 14:30:15
```



```
timestamp = datetime.now().timestamp()  
print(timestamp)
```



```
from datetime import timezone
```

```
iso_utc = datetime.now(timezone.utc).isoformat()  
print(iso_utc) # 2025-02-17T14:30:00.123456+00:00
```

```
utc_now = datetime.now(pytz.utc)  
kyiv_tz = pytz.timezone("Europe/Kiev")
```

