IT CAREER HUB

Python

Файлы JSON и модуль datetime

Преподаватель

Портрет

Имя Фамилия

Текущая должность

Количество лет опыта

Какой у Вас опыт - ключевые кейсы

Самые яркие проекты

Дополнительная информация по вашему усмотрению

Корпоративный e-mail

Социальные сети (по желанию)

Важно

- Камера должна быть включена на протяжении всего занятия
- В течение занятия вопросы задавать в чате или когда преподаватель спрашивает, есть ли у Вас вопросы
- Вести себя уважительно и этично по отношению к остальным участникам занятия
- Организационные вопросы по обучению решаются с кураторами, а не на тематических занятиях
- Во время занятия будут интерактивные задания, будьте готовы включить камеру или демонстрацию экрана по просьбе преподавателя

Повторение

- [ ] Генератор
- [ ] Функция с `yield`
- [ ] Ключевое слово `yield`
- [ ] Генераторные функции с параметрами
- [ ] Использование генератора в `for`
- [ ] Бесконечные генераторы
- [ ] Метод `close`
- [ ] Конструкция `yield from`

План занятия

- JSON
- Модуль `json`
- Сравнение типов Python и JSON
- Форматирование JSON
- `JSONDecodeError`
- Модуль `datetime`

ICH IT CAREER HUB

ОСНОВНОЙ БЛОК

IT CAREER HUB

JSON

Полезно знать

ICH
IT CAREER HUB

JSON (JavaScript Object Notation)

Это текстовый формат для хранения и передачи данных. Он представляет данные в удобном для чтения виде и используется для обмена информацией между системами

Пример JSON-объекта

```
{
    "name": "Alice",
    "age": 25,
    "is_student": false,
    "courses": ["Math", "Physics"]
}
```

Особенности JSON

- Структура представляет собой объекты и массивы
- Поддерживает числа, строки, булевы значения, массивы и объекты
- Все строки в JSON записываются только в двойных кавычках
- Данные хранятся в текстовом виде

JSON применяется для передачи или хранения структурированных данных

- API (обмен данными между системами)
- Базы данных
- Конфигурационные файлы
- Фронтенд и бэкенд

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

Модуль json

Полезно знать

IT CAREER HUB

Модуль json

Этот модуль позволяет сериализовать и десериализовать данные в формате JSON

Полезно знать

ICH
IT CAREER HUB

Сериализация

Это процесс преобразования данных в формат JSON, чтобы их можно было хранить или передавать

Функции для сериализации JSON

```
json.dumps(obj)
```

```
json.dump(obj, file)
```

Полезно знать

IT CAREER HUB

Функция json.dumps

Эта функция преобразует Python-объект в JSON-строку, чтобы передать или сохранить данные в текстовом формате

Пример использования json.dumps

```python
import json

data = {"name": "Alice", "age": 25, "is_student": False}

json_string = json.dumps(data)  # Преобразование в JSON-строку
print(type(json_string))
print(json_string)
```

Полезно знать

IT CAREER HUB

Функция json.dump

Эта функция записывает Python-объект в файл в формате JSON

Пример использования json.dump

```python
import json

data = {"name": "Alice", "age": 25, "is_student": False}

with open("data.json", "w") as file:
    json.dump(data, file)  # Запись JSON в файл
```

Применение

```
json.dumps(obj)
```

Преобразование объекта Python в JSON-строку для передачи по сети или хранения в базе данных

```
json.dump(obj, file)
```

Запись объекта Python в JSON-файл, например для сохранения конфигураций

Полезно знать

ICH
IT CAREER HUB

Десериализация

Это процесс обратного преобразования JSON в объект Python, чтобы с ним можно было работать в программе

Функции для десериализации JSON

```
json.loads(json_string)
```

```
json.load(file)
```

Полезно знать

IT CAREER HUB

Функция json.loads

Эта функция преобразует JSON-строку в объект Python, чтобы с ним можно было работать в коде

Пример использования json.loads

```
import json

json_object = '{"name": "Alice", "age": 25, "is_student": false}'
json_objects = '[{"name": "Alice", "age": 25, "is_student": false},
{"name": "Bob", "age": 20, "is_student": true}]'

data_dict = json.loads(json_object)    # Преобразование JSON-строки в
Python-объект
print(type(data_dict))
print(data_dict)

data_list = json.loads(json_objects)    # Преобразование JSON-строки в
Python-объект
print(type(data_list))
print(data_list)
```

Полезно знать

IT CAREER HUB

Функция json.load

Эта функция загружает данные из JSON-файла в Python-объект

Пример использования json.load

```
import json

with open("data.json", "r") as file:
    data = json.load(file)  # Загрузка JSON из файла

print(type(data))
print(data)
```

Применение

`json.loads()`

Если JSON приходит **в виде строки**, например, из API

`json.dump(obj, file)`

Если JSON **хранится в файле** и его нужно загрузить в Python

IT CAREER HUB

ВОПРОСЫ

ICH IT CAREER HUB

Сравнение типов Python и JSON

Сравнение типов Python и JSON

Тип в Python	Тип в JSON	Пример Python	Пример JSON
dict	object	{"name": "Alice"}	{"name": "Alice"}
list	array	["apple", "banana"]	["apple", "banana"]
tuple	array	("apple", "banana")	["apple", "banana"]
str	string	"Hello"	"Hello"
int	number	42	42
float	number	3.14	3.14
bool	boolean	True, False	true, false
NoneType	null	None	null

Особенности

- Кортежи (tuple) преобразуются в массив (array), `set` и `frozenset` не поддерживаются в JSON
- Булевы значения (`True`, `False`) в JSON записываются **с маленькой буквы** (`true`, `false`)
- Отсутствие значения (`None`) в JSON записывается как `null`

Пример: запись в файл объекта, содержащего все типы данных

```python
import json

data = {
    "dict_example": {"key": "value"},
    "list_example": ["apple", "banana"],
    "tuple_example": ("apple", "banana"),
    "string_example": "Hello",
    "int_example": 42,
    "float_example": 3.14,
    "bool_example_true": True,
    "bool_example_false": False,
    "none_example": None
}

# Запись в файл data.json
with open("data.json", "w", encoding="utf-8") as file:
    json.dump(data, file)
```

Пример: чтение обратно в Python

```
import json

# Чтение данных обратно в Python
with open("data.json", "r", encoding="utf-8") as file:
    loaded_data = json.load(file)

# Вывод загруженных данных
print(type(loaded_data))
print(loaded_data)
```

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

Форматирование JSON

Важно

По умолчанию `json.dumps()` и `json.dump()` создают **однострочный JSON**, что делает его менее читабельным.

Параметры для форматирования JSON

`indent`

`ensure_ascii`

`sort_keys`

Полезно знать

IT CAREER HUB

Параметр indent

Этот параметр используется для добавления отступов в JSON, чтобы сделать его более читаемым

Пример использования indent

```
import json

data = {"name": "Alice", "age": 25, "is_student": False,
        "courses": {"math": "A", "physics": "B" }}

json_string = json.dumps(data)  # Без отступа
print(json_string)

json_string = json.dumps(data, indent=4)  # С отступом
print(json_string)
```

Полезно знать

IT CAREER HUB

Параметр ensure_ascii

Этот параметр определяет, как JSON будет хранить юникод-символы

Пример использования ensure_ascii

```
import json

data = {"город": "Берлин", "страна": "Германия"}

json_string = json.dumps(data)  # По умолчанию (ensure_ascii=True)
print(json_string)

json_string = json.dumps(data, ensure_ascii=False)  # Отключаем ASCII-кодировку
print(json_string)
```

Полезно знать

ICH
IT CAREER HUB

Параметр sort_keys

Этот параметр используется для сортировки ключей в JSON-объекте по алфавиту

Пример использования sort_keys

```
import json

data = {"name": "Alice", "age": 25, "is_student": False, "city":
"London"}

json_string = json.dumps(data, indent=4)  # Без сортировки ключей
print(json_string)

json_string = json.dumps(data, indent=4, sort_keys=True)  # Сортировка
ключей
print(json_string)
```

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

JSONDecodeError

Полезно знать

JSONDecodeError

Эта ошибка возникает, если строка JSON имеет неверный формат и не может быть разобрана с помощью `json.loads()` или `json.load()`. Она указывает, что JSON-данные повреждены, содержат синтаксические ошибки или не соответствуют ожиданиям

Пример возникновения JSONDecodeError

```
import json

invalid_json = '{"name": "Alice", "age": 25, "is_student": false,'   # Ошибка: нет закрывающей скобки

data = json.loads(invalid_json)  # Загрузка некорректного JSON
```

Важно

Используйте `try-except**` при загрузке JSON, чтобы избежать ошибки

Пример использования try-except**

```
import json

invalid_json = '{"name": "Alice", "age": 25, "is_student": false,'   # Ошибка: нет закрывающей скобки

try:
      data = json.loads(invalid_json)   # Попытка загрузки некорректного JSON
except json.JSONDecodeError as e:
    print(f"Ошибка декодирования JSON: {e}")
```

Причины JSONDecodeError

Причина	Пример
Пропущенные кавычки или запятые	`{"name": "Alice", "age": 25, "is_student": false,}  # Лишняя запятая`
Использование одинарных кавычек вместо двойных	`{'name': 'Alice'}  # Неверный формат`
Неполные или повреждённые данные	`{"name": "Alice", "age": 25  # Нет закрывающей скобки`

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

ЗАДАНИЯ

Выберите верный вариант ответа

Какой результат выдаст следующий код?

```
import json

data = {
    "city": "Paris",
    "temperature": 22,
    "is_rainy": False
}

json_string = json.dumps(data)
loaded_data = json.loads(json_string)

print(type(json_string))
print(type(loaded_data))
```

a. Будет выведено: <class 'dict'>, <class 'str'>
b. Будет выведено: <class 'str'>, <class 'dict'>
c. Типы неизвестны, зависит от содержимого data

Выберите верный вариант ответа

Какой результат выдаст следующий код?

```
import json

data = {
    "city": "Paris",
    "temperature": 22,
    "is_rainy": False
}

json_string = json.dumps(data)
loaded_data = json.loads(json_string)

print(type(json_string))
print(type(loaded_data))
```

a. Будет выведено: <class 'dict'>, <class 'str'>
b. Будет выведено: <class 'str'>, <class 'dict'>
c. Типы неизвестны, зависит от содержимого data

Найдите ошибку и исправьте код

```
import json

data = {"name": "Alice", "age": 25}
with open("user.json", "w", encoding="utf-8") as f:
    json.dumps(data, f)
```

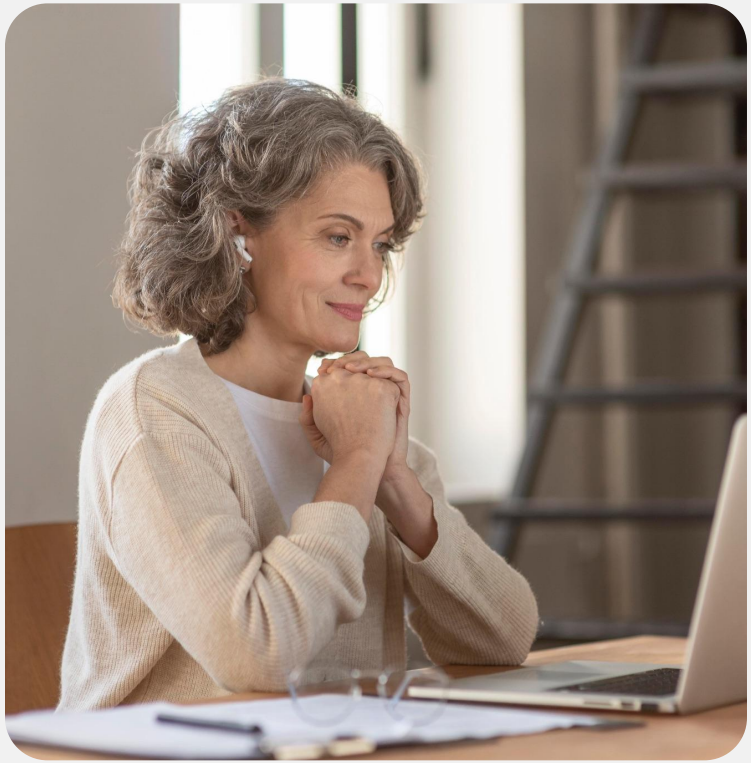
Найдите ошибку и исправьте код

Решение:

Используется неправильный метод преобразования `json.dumps`.

Исправленный код:

```
import json

data = {"name": "Alice", "age": 25}
with open("user.json", "w", encoding="utf-8") as f:
    json.dump(data, f)
```

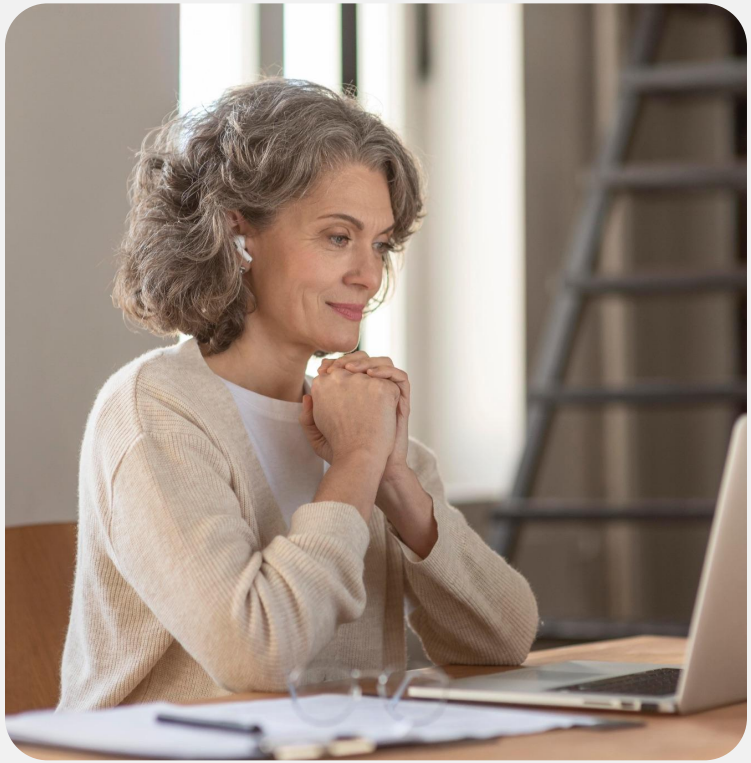


Выберите верные варианты ответа

Что произойдёт при десериализации этой строки?

```
import json
json.loads("{'x': 1, 'y': 2}")
```

a. Будет создан словарь
b. Будет создана строка
c. Произойдет запись в файл
d. Возникнет `JSONDecodeError`

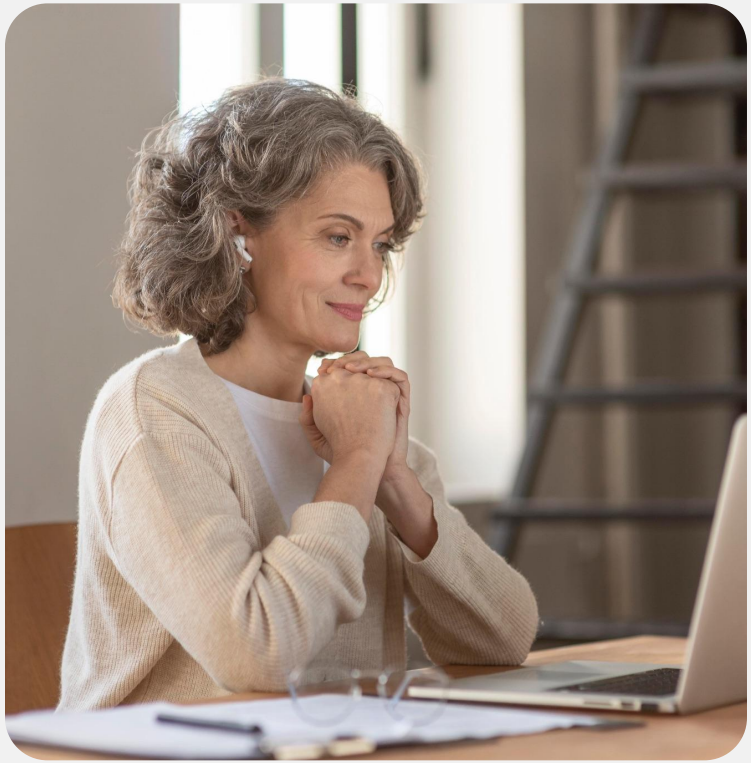


Выберите верные варианты ответа

Что произойдёт при десериализации этой строки?

```
import json
json.loads("{'x': 1, 'y': 2}")
```

a. Будет создан словарь
b. Будет создана строка
c. Произойдет запись в файл
d. Возникнет `JSONDecodeError`



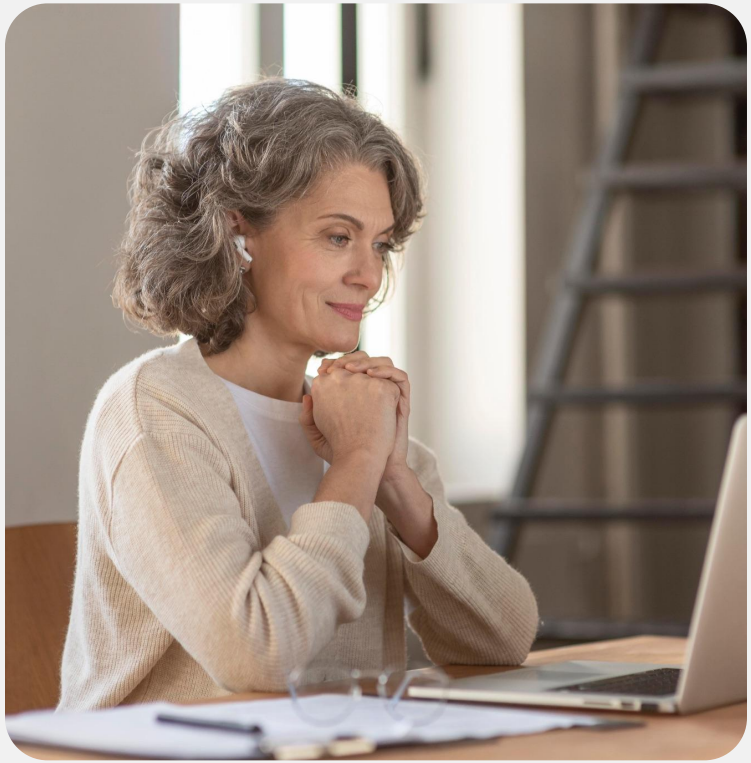
Выберите верные варианты ответа

Почему этот код вызывает ошибку?

```
import json

data = {"values": {1, 2, 3}}
json.dumps(data)
```

a. Ошибка в ключах словаря
b. JSON не поддерживает множества
c. Нельзя сериализовать числа
d. Кортежи не поддерживаются

a.



Выберите верные варианты ответа

Почему этот код вызывает ошибку?

```
import json

data = {"values": {1, 2, 3}}
json.dumps(data)
```

a. Ошибка в ключах словаря
b. JSON не поддерживает множества
c. Нельзя сериализовать числа
d. Кортежи не поддерживаются

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

Модуль datetime

Полезно знать

ICH
IT CAREER HUB

Модуль datetime

Этот модуль предоставляет инструменты для работы с датами, временем и их форматированием.
Он позволяет получать текущее время, вычислять разницу между датами, конвертировать форматы и работать с часовыми поясами.

Полезно знать

IT CAREER HUB

Метод datetime.now()

Этот метод модуля `datetime` используется для получения текущей даты и времени

Пример получения текущей даты и времени

```
from datetime import datetime

now = datetime.now()  # Получаем текущую дату и время
print(type(now))  # Объект datetime
print(now)
```

Особенности

- Возвращает объект `datetime` с текущей датой и временем
- Позволяет извлекать отдельные компоненты даты

Зачем это нужно

- Фиксация времени событий
- Отметка времени при выполнении операций
- Создание временных меток

Пример получения компонентов даты

```
from datetime import datetime

now = datetime.now()

print("Год:", now.year)
print("Месяц:", now.month)
print("День:", now.day)
print("Часы:", now.hour)
print("Минуты:", now.minute)
print("Секунды:", now.second)
```

Полезно знать

IT CAREER HUB

Метод strftime()

Этот метод используется для преобразования даты в строку в нужном формате

Пример

```
from datetime import datetime

now = datetime.now()
# Преобразование в строку
print(str(now))
# Преобразование в строку указанного формата
formatted_date = now.strftime("%d.%m.%Y %H:%M:%S")
print(type(formatted_date))
print(formatted_date)
```

Популярные коды форматов strftime

Код	Описание	Пример
%d	День (01-31)	28
%m	Месяц (01-12)	02
%Y	Год (4 цифры)	2025
%y	Год (2 цифры)	25
%H	Часы (00-23)	14
%M	Минуты (00-59)	30
%S	Секунды (00-59)	15
%A	Полное название дня	Friday
%B	Полное название месяца	February

Примеры форматирования

```
from datetime import datetime

now = datetime.now()
print(now.strftime("%Y-%m-%d"))       # 2025-02-28 (ISO формат)
print(now.strftime("%d/%m/%Y"))       # 28/02/2025 (европейский формат)
print(now.strftime("%I:%M %p"))       # 02:30 PM (12-часовой формат)
print(now.strftime("%A, %B %d, %Y"))  # Friday, February 28, 2025
```

Полезно знать

ICH
IT CAREER HUB

Метод strptime()

Этот метод используется для преобразования строки в объект `datetime`.
Это необходимо, когда дата хранится в виде текста (например, в файле) и её нужно использовать для вычислений или фильтрации

Пример

```
from datetime import datetime

date_string = "28|02|2025 14-30-15"  # Дата в виде строки
date_obj   =   datetime.strptime(date_string,   "%d|%m|%Y   %H-%M-%S")     #
Указываем форматы и те же разделители

print(type(date_obj))
print(date_obj)
```

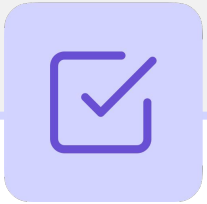
Сравнение дат

Python позволяет **сравнивать даты** так же, как числа, используя операторы сравнения (>, <, ==, !=, >=, <=). Объекты `datetime` можно сравнивать напрямую, так как они содержат **и дату, и время**.

Пример: разбор даты из строки и сравнение с текущей датой**

```
from datetime import datetime

now = datetime.now()
deadline = datetime.strptime("01.12.2025", "%d.%m.%Y")

if now > deadline:
    print("Срок истёк!")
else:
    print("До дедлайна ещё есть время.")
```

Разница между датами



Python позволяет **вычислять разницу между датами** с помощью **вычитания объектов** `datetime`, которое возвращает объект `timedelta`.

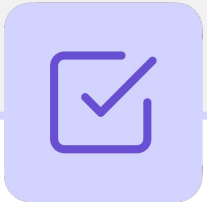
Пример: Разница между датами

```
from datetime import datetime

date1 = datetime(2025, 2, 28)
date2 = datetime(2025, 3, 5)

difference = date2 - date1  # Разница между датами
print(type(difference))
print(difference)
print(difference.days)
```

Пример: Разница с учётом времени

```
from datetime import datetime

dt1 = datetime(2025, 2, 28, 14, 30)
dt2 = datetime(2025, 3, 2, 10, 0)

difference = dt2 - dt1
print(difference)
print(difference.total_seconds())
```

Полезно знать



Объект `timedelta` можно использовать для **изменения дат**, добавляя или вычитая временные интервалы.

Пример: Добавление и вычитание времени

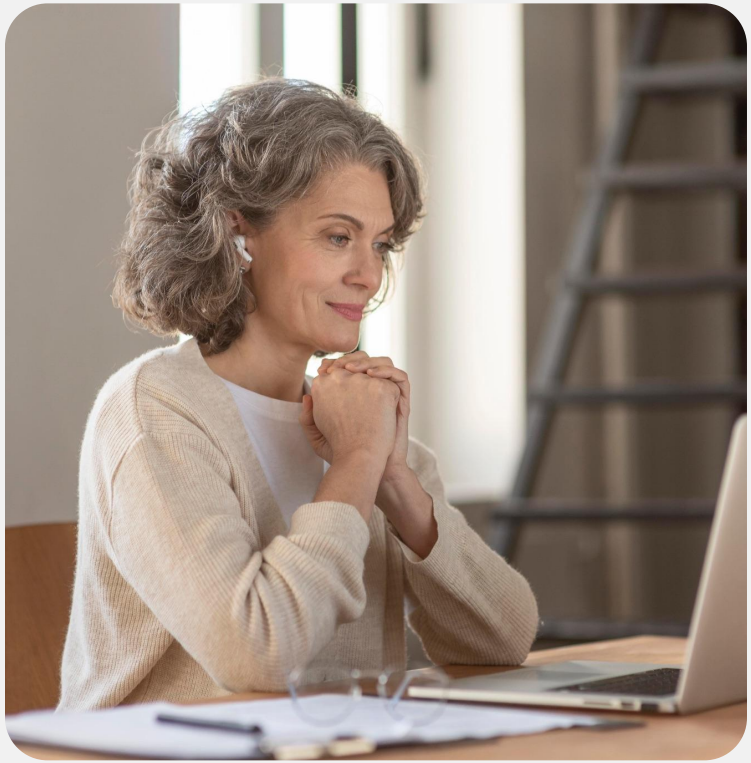
```python
from datetime import datetime, timedelta

# Дата начала задачи
start_date = datetime(2025, 2, 28)

# Дедлайн через 2 недели
deadline = start_date + timedelta(weeks=2)
print("Дедлайн:", deadline.strftime("%d.%m.%Y"))

# Проверка, прошёл ли дедлайн
today = datetime(2025, 3, 15)  # Текущая дата

if deadline > today:
    print("Дедлайн пропущен!")
else:
    print("Ещё есть время для выполнения задачи.")
```

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

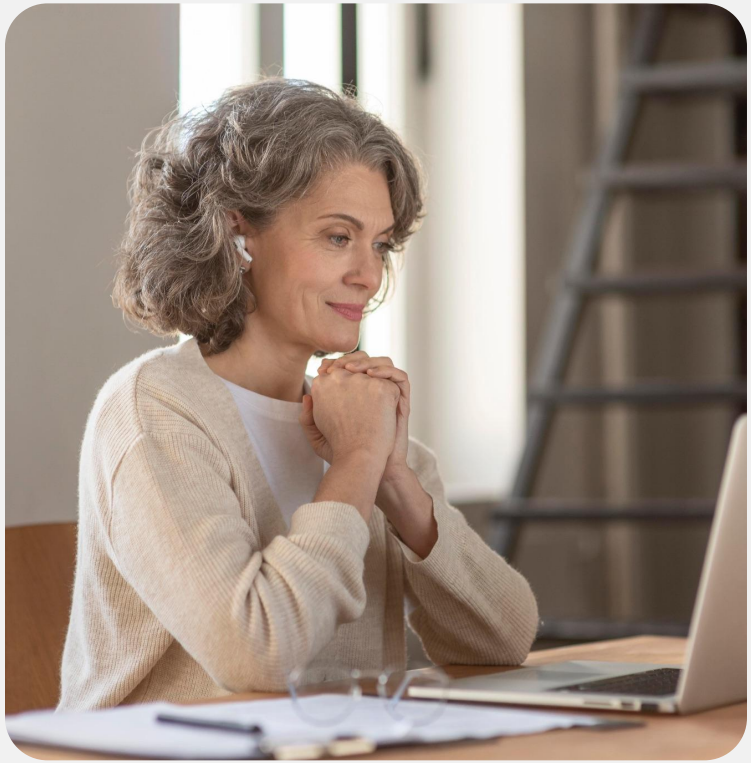
ЗАДАНИЯ



Выберите верный вариант ответа

Какой тип вернёт функция datetime.now()?

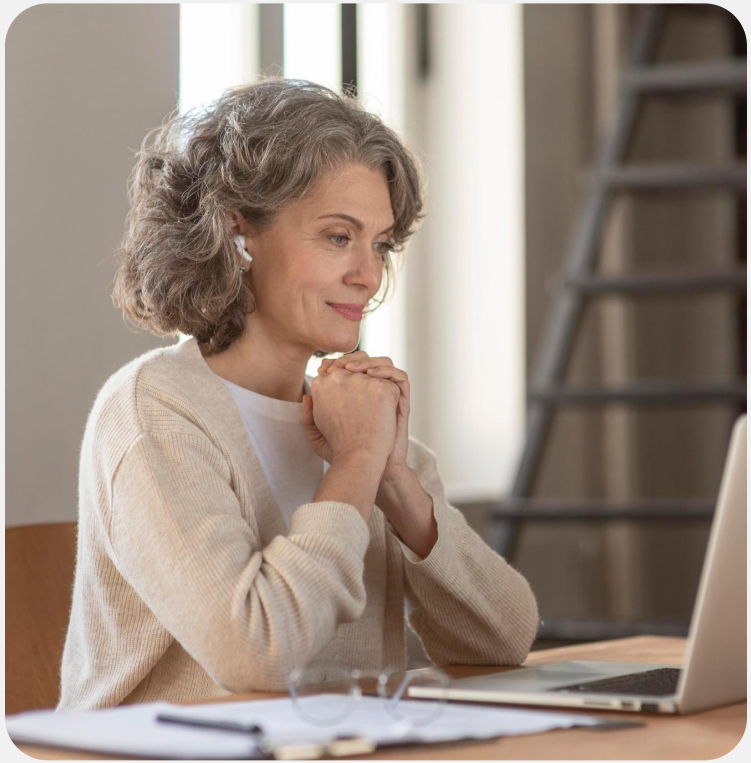
a. date
b. datetime
c. str
d. time



Выберите верный вариант ответа

Какой тип вернёт функция datetime.now()?

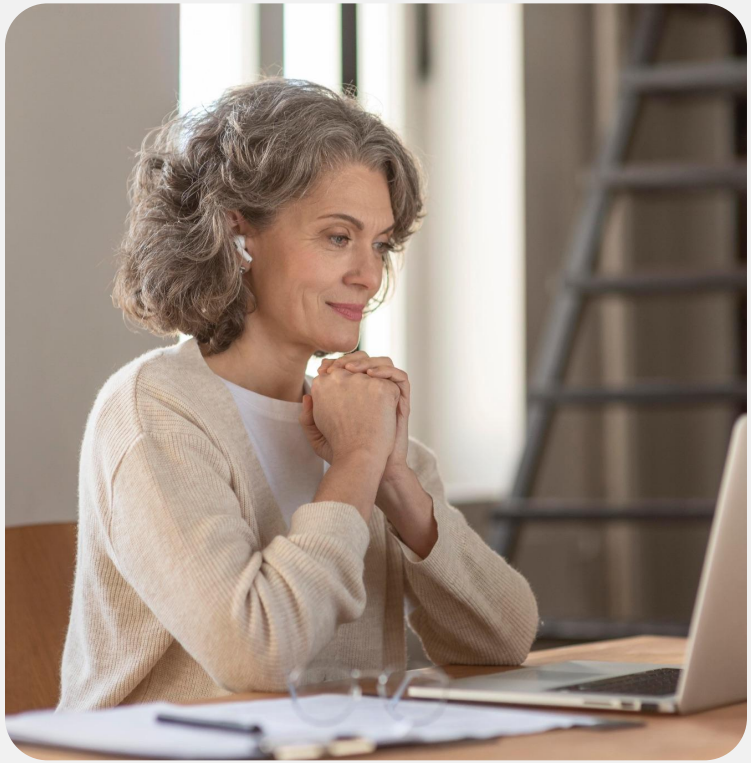
a. date
b. datetime
c. str
d. time



Выберите верный вариант ответа

Какой формат соответствует строке "01|12|2025 14-30-00"?

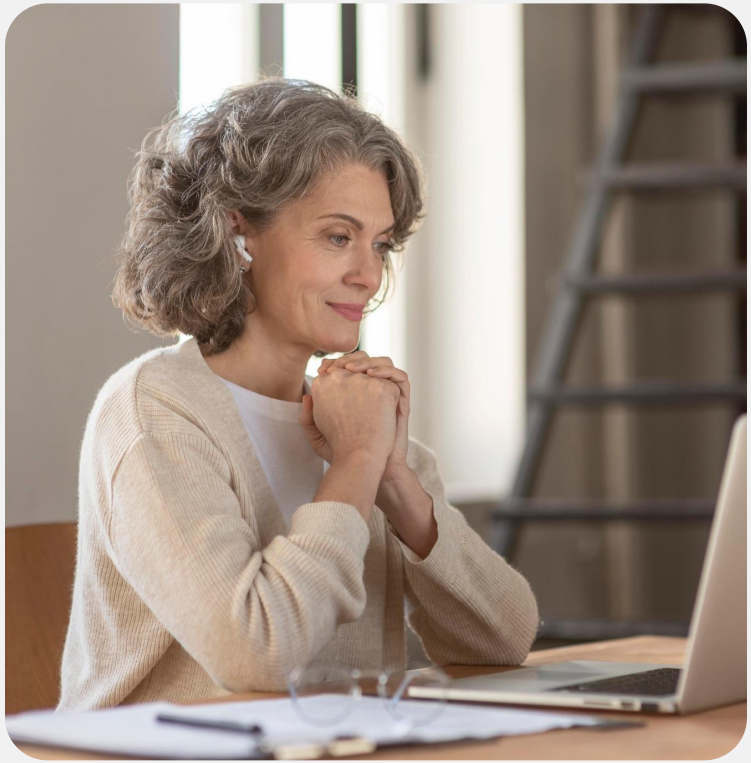
a. "%dd|%mm|%YYYY %HH-%MM-%SS"
b. "%dd|%mm|%yyyy %hh-%mm-%ss"
c. "%d|%m|%Y %H-%M-%S"
d. "%d|%m|%Y %h-%m-%s"



Выберите верный вариант ответа

Какой формат соответствует строке "01|12|2025 14-30-00"?

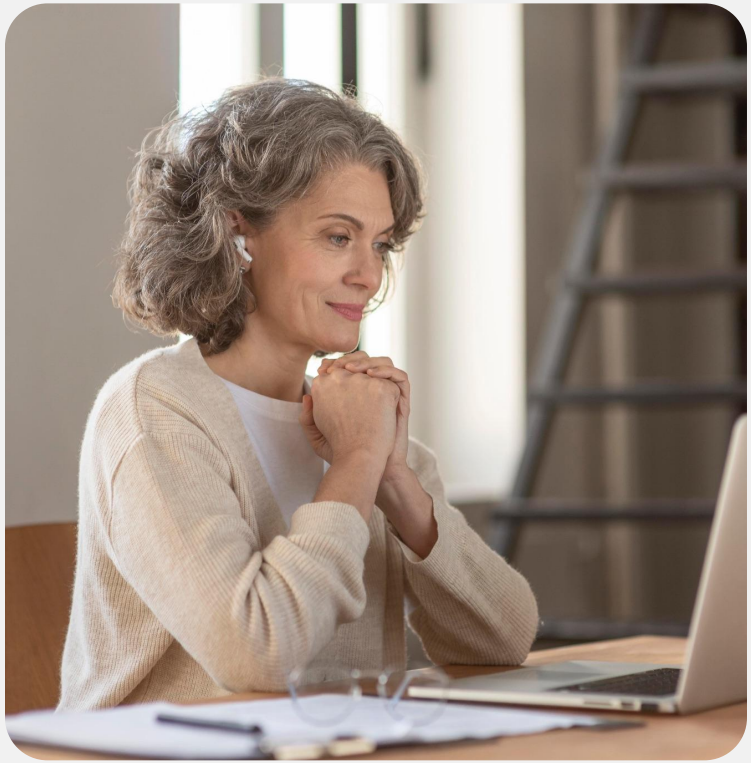
a. "%dd|%mm|%YYYY %HH-%MM-%SS"
b. "%dd|%mm|%yyyy %hh-%mm-%ss"
c. "%d|%m|%Y %H-%M-%S"
d. "%d|%m|%Y %h-%m-%s"



Выберите верный вариант ответа

Что делает strftime()?

a. Преобразует строку в дату
b. Сравнивает две даты
c. Возвращает разницу между датами
d. Преобразует дату в строку



Выберите верный вариант ответа

Что делает strftime()?

a. Преобразует строку в дату
b. Сравнивает две даты
c. Возвращает разницу между датами
d. Преобразует дату в строку

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

ПРАКТИЧЕСКАЯ РАБОТА

Поиск низких оценок за период

Реализуйте программу, которая должна:

- Прочитать данные из файла `grades.json`.
- Реализовать функцию `filter_low_scores()`, которая:
  - Принимает минимальный проходной балл (`threshold`) и диапазон дат (`start_date, end_date`) в формате дд-мм-гггг.
  - Возвращает все оценки **ниже порога**, полученные **в заданный период**.
  - Сохраняет отфильтрованные записи в файл `filtered_low_scores.json`.

Поиск низких оценок за период

Данные:

```
 {"name": "Bob", "subject": "Science", "grade": 86, "date": "06-09-2025"},
  {"name": "Diana", "subject": "Science", "grade": 85, "date": "31-01-2025"},
  {"name": "Bob", "subject": "Literature", "grade": 60, "date": "19-07-2025"},
  {"name": "Charlie", "subject": "Literature", "grade": 78, "date": "05-08-2025"},
  {"name": "Ethan", "subject": "Literature", "grade": 69, "date": "08-04-2025"},
  {"name": "Charlie", "subject": "Science", "grade": 63, "date": "24-10-2025"},
  {"name": "Ethan", "subject": "Math", "grade": 80, "date": "30-01-2025"},
  {"name": "Alice", "subject": "Physics", "grade": 90, "date": "15-09-2025"},
  {"name": "Ethan", "subject": "Science", "grade": 63, "date": "18-09-2025"},
  ...
]
```

Пример вызова:

```
filter_low_scores(70, "01-01-2025", "31-03-2025")
```

Поиск низких оценок за период

Пример вызова:

```
filter_low_scores(70, "01-01-2025", "31-03-2025")
```

Пример вывода (filtered_low_scores.json):

```
[
    {"name": "Ethan", "subject": "History", "grade": 66, "date": "10-03-2025"},
    {"name": "Bob", "subject": "Literature", "grade": 68, "date": "22-01-2025"},
    {"name": "Ethan", "subject": "History", "grade": 62, "date": "25-02-2025"}
]
```

IT CAREER HUB

ДОМАШНЕЕ ЗАДАНИЕ

Домашнее задание

Анализ курсов студентов

Реализуйте программу, которая должна:

1. Прочитать файл `student_courses.json`, содержащий:
   a. Имя,
   b. дату рождения (`birth_date`) в формате дд.мм.гггг,
   c. дату поступления (`enrollment_date`) в том же формате,
   d. список курсов.
2. Вычислить:
   a. Общее количество студентов.
   b. Средний возраст на момент поступления.
   c. Количество студентов на каждом курсе.
3. Сохранить отчёт в JSON-файл `student_courses_report.json`.

Домашнее задание

Анализ курсов студентов

Данные:


```
[
  {"name": "Diana Williams", "birth_date": "12.06.1983", "enrollment_date": "29.04.2023", "courses": ["Physics",
"Chemistry"]},
  {"name": "Tina Miller", "birth_date": "06.07.2004", "enrollment_date": "18.04.2020", "courses": ["Biology", "Business"]},
  {"name": "Kevin Miller", "birth_date": "20.12.2004", "enrollment_date": "16.12.2020", "courses": ["Linguistics", "Math",
"History"]},
  {"name": "Fiona Brown", "birth_date": "05.07.1999", "enrollment_date": "02.09.2022", "courses": ["Art", "Philosophy"]},
  {"name": "Charlie Davis", "birth_date": "17.07.1998", "enrollment_date": "17.05.2023", "courses": ["Chemistry", "Physics",
"Business"]},
  {"name": "Diana Jones", "birth_date": "24.12.1980", "enrollment_date": "26.11.2021", "courses": ["Economics",
"Linguistics"]},
  {"name": "Alice Johnson", "birth_date": "22.09.1981", "enrollment_date": "23.12.2020", "courses": ["Chemistry", "Economics",
"Math"]},
  {"name": "Ian Lopez", "birth_date": "23.11.2001", "enrollment_date": "07.05.2020", "courses": ["Philosophy", "Art",
"Physics"]},
  {"name": "Kevin Davis", "birth_date": "30.01.1997", "enrollment_date": "20.03.2021", "courses": ["Math", "Economics"]},
  ...
]
```

Домашнее задание

Анализ курсов студентов

Пример вызова:

```
filter_low_scores(70, "01-01-2025", "31-03-2025")
```

Заключение