JSON и Python: перенос данных без потери смысла

1. Что возвращает json.loads('[1,2]') в Python?

1. Список [1,2]
2. Множество {1,2}
3. Строку '[1,2]'
4. Кортеж (1,2)

Правильный ответ: Список [1,2]

2. Какое JSON-значение соответствует Python None при стандартном кодировании?

1. nil
2. null
3. undefined
4. None

Правильный ответ: null

3. Как записывается истина в JSON?

1. true
2. yes
3. True
4. TRUE

Правильный ответ: true

4. Какие кавычки требует JSON для строковых ключей?

1. Ключи вообще не заключаются в кавычки
2. Любые без ограничений
3. Двойные
4. Только одинарные

Правильный ответ: Двойные

5. Допустима ли завершающая запятая в стандартном JSON-массиве [1,2,]?

1. Только при UTF-8
2. Да, всегда
3. Нет
4. Только у чисел

Правильный ответ: Нет

6. Какой метод превращает Python-объект в JSON-строку?

1. json.dumps
2. json.loads
3. str.strip
4. json.load

Правильный ответ: json.dumps

7. Что возвращает json.loads('{"n":3}')['n']?

1. Строку '3'
2. None
3. Число 3
4. Список [3]

Правильный ответ: Число 3

8. Можно ли стандартным json.dumps напрямую сериализовать set без дополнительного обработчика?

1. Да, всегда как null
2. Да, всегда как объект
3. Нет, возникает TypeError
4. Да, всегда как массив

Правильный ответ: Нет, возникает TypeError

9. Что делает ensure_ascii=False в json.dumps со строками кириллицы?

1. Удаляет все неанглийские буквы
2. Позволяет оставить символы кириллицы в JSON-строке
3. Превращает числа в текст
4. Шифрует строку

Правильный ответ: Позволяет оставить символы кириллицы в JSON-строке

10. Почему успешный json.loads ещё не гарантирует корректность заявки?

1. Все JSON-объекты имеют одинаковые поля
2. JSON автоматически проверяет возраст и согласие
3. Парсер подтверждает достоверность человека
4. Нужно проверить схему, типы и обязательные поля

Правильный ответ: Нужно проверить схему, типы и обязательные поля