

## 7 модуль. Задания

### Глава 7.1. `async with` и асинхронные контекстные менеджеры

1. Какие методы должен определять объект, чтобы его можно было использовать в блоке `async with`?

- `__aenter__` и `__aexit__`
- `__start__` и `__end__`
- `__enter__` и `__exit__`
- `__async__` и `__await__`

Правильный ответ: `__aenter__` и `__aexit__`

2. Что происходит при вызове `__aenter__` в асинхронном контекстном менеджере?

- Подготавливаются ресурсы для использования
- Запускается асинхронная функция
- Освобождаются ресурсы
- Завершается использование ресурсов

Правильный ответ: Подготавливаются ресурсы для использования

3. Что происходит при вызове `__aexit__` в асинхронном контекстном менеджере?

- Освобождаются ресурсы и корректно завершается их использование
- Запускается асинхронная функция
- Подготавливаются ресурсы для использования
- Запускается процесс аренды ресурсов

Правильный ответ: Освобождаются ресурсы и корректно завершается их использование

4. В каких случаях может быть полезно использование `async with`?

- Все вышеперечисленное
- При работе с асинхронными сетевыми соединениями
- При асинхронном взаимодействии с файлами
- При использовании асинхронных примитивов синхронизации

Правильный ответ: Все вышеперечисленное

5. Что может произойти при неправильном использовании `async with`?

- Улучшение производительности
- Уменьшение количества ошибок
- Неправильное управление ресурсами и возможные блокировки
- Увеличение скорости выполнения

Правильный ответ: Неправильное управление ресурсами и возможные блокировки

### Глава 7.2. `async for` и асинхронные итераторы

1. Что такое `async for`?

- Метод для выполнения цикла `for` в обратном порядке
- Метод для асинхронного выполнения цикла `for`
- Метод для синхронного выполнения цикла `for`
- Метод для выполнения цикла `for` без итерации

Правильный ответ: Метод для асинхронного выполнения цикла `for`

2. Какой основной принцип работы `async for`?

- Итерация с блокировкой событийного цикла
- Блокировка выполнения программы
- Итерация с использованием многопоточности
- Итерация без блокировки событийного цикла

Правильный ответ: Итерация без блокировки событийного цикла

3. В каких сценариях используется `async for`?

- Все вышеперечисленное
- Параллельная обработка задач
- Работа с асинхронными API
- Обработка асинхронных потоков данных

Правильный ответ: Все вышеперечисленное

4. Что необходимо, чтобы объект можно было использовать в `async for`?

- Поддержка синхронного итератора
- Поддержка асинхронного итератора (`__aiter__` и `__anext__`)
- Поддержка многопоточности
- Экземпляр любого класса

Правильный ответ: Поддержка асинхронного итератора (`__aiter__` и `__anext__`)