**Тест Даукша Денис. Самостоятельная №3:**

1. Что такое функция и как ее объявить?  
   **В Python функция объявляется с использованием ключевого слова def, за которым следует имя функции и скобки, в которых указываются аргументы функции.**
2. Как создать функцию?

**Чтобы создать функцию в Python, используйте ключевое слово "def", за которым следует имя функции и круглые скобки с параметрами функции (если они есть). Затем, после двоеточия, вы можете написать тело функции с помощью отступа.**

1. Как показать что созданная функция возвращает результат?

**В Python, чтобы показать, что функция возвращает результат, вы можете использовать ключевое слово `return` с выражением, которое представляет возвращаемое значение.**

1. В чем разница между параметрами и аргументами функции?

**Параметры - это имена, указанные в определении функции. Аргументы - это значения, передаваемые в функцию при вызове.**

1. Как рекурсивные функции работают в Python?

**Когда рекурсивная функция вызывается, она решает некоторую базовую задачу и вызывает саму себя для решения подзадачи. Рекурсия продолжается, пока не достигнется условие окончания**

1. Что такое декораторы функций и как они используются? ( привести свой пример декоратора)  
   **Декораторы функций в Python - это специальные функции, которые позволяют изменять поведение других функций без изменения их собственного кода. Они оборачивают функцию вокруг другой функции, добавляя определенное поведение до и/или после вызова оригинальной функции.**

**Пример декоратора:**

**def print\_decorator(func):**

**def wrapper():**

**print("Перед выполнением функции")**

**func()**

**print("После выполнения функции")**

**return wrapper**

**@print\_decorator**

**def hello():**

**print("Привет, мир!")**

1. Как расшифровывается ООП?  
   **Объекто-Ориентированное Программирование**
2. Что относится к основным принципам(парадигмам) ООП?

**Наследование  
Полиморфизм  
Инкапсуляция  
Абстракция**

1. Что будет результатом программы и почему?

**None  
  
В классе Test объявлен статический атрибут test со значением None. Результат по ссылке с путём Test.test является значение None**

1. Какой принцип ООП описывает следующее предложение? Этот принцип является способностью использовать общий интерфейс для нескольких форм (типов данных).

**Полиморфизм**

1. Какой из перечисленных вариантов является верным объявлением private поля?

* private field = 0
* field = 0
* \_field = 0
* **\_\_field = 0**

1. Как создать конструктор класса?

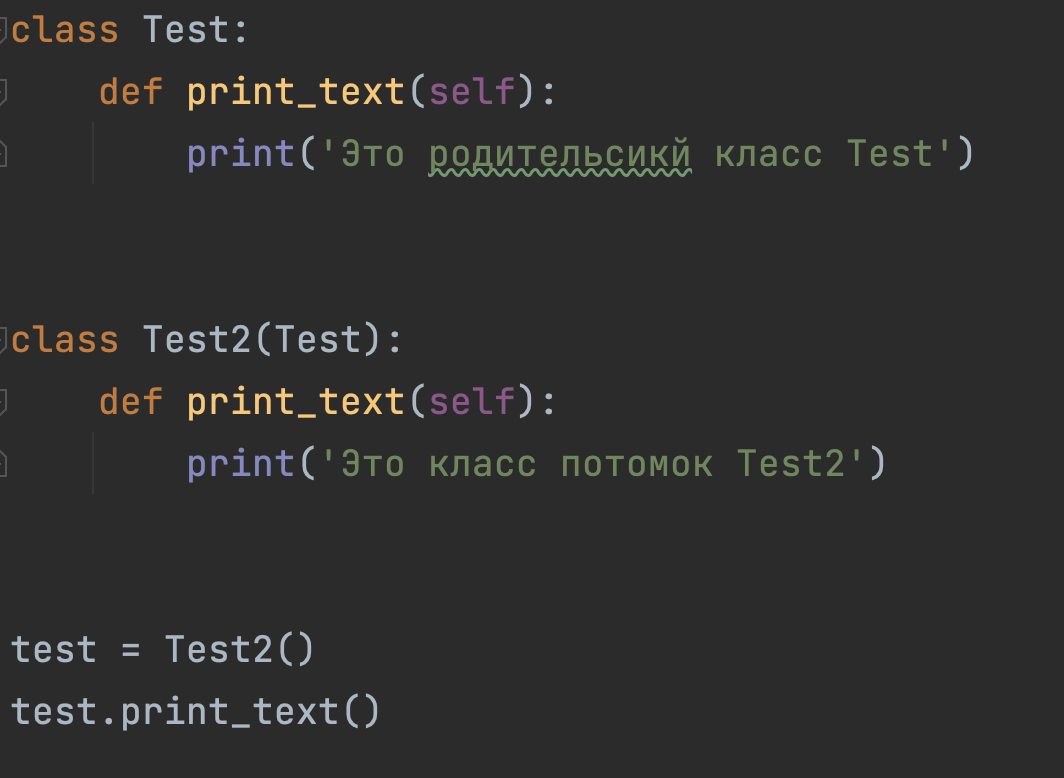
**Чтобы создать конструктор класса в Python, необходимо определить метод `\_\_init\_\_()` внутри класса.**

**Конструктор используется для инициализации объекта класса при его создании. Вы можете добавить любую необходимую логику в метод `\_\_init\_\_()` для инициализации состояния объекта класса.**

1. Как много конструкторов в классе может иметь Python?

**В классе может быть только один конструктор.**

1. Что будет результатом данной программы



**Это класс потомок Test2**

1. Какой параметр обязательно принимает в себя метод экземпляра?

* тип объекта
* **сам экземпляр класса**
* название класса
* таких аргументов нет

1. Как использовать инкапсуляцию для защиты данных в классах?

**В Python инкапсуляция - это механизм, который позволяет ограничить доступ к атрибутам и методам класса извне. Он используется для защиты данных и предоставления контроля над операциями с ними.**

**Для использования инкапсуляции в Python используются особые методы и соглашения по именованию. Существуют два уровня инкапсуляции: слабая инкапсуляция и сильная инкапсуляция.**

**Слабая инкапсуляция:**

**protected => \_(name)**

**Сильная инкапсуляция:**

**private => \_\_(name)**

1. Какая разница между абстрактными классами и интерфейсами в Python?

**Разница между абстрактными классами и интерфейсами в Python заключается в том, что абстрактные классы могут содержать и конкретные методы и свойства, в то время как интерфейсы определяют только абстрактные методы без конкретной реализации.**

1. Как показать в классе что метод является абстрактым?

**В классе можно показать, что метод является абстрактным, используя модуль abc (Abstract Base Classes) из стандартной библиотеки Python. В данном модуле есть класс ABCMeta, который является метаклассом для создания абстрактных классов. Использовать декоратор @abstractmethod**

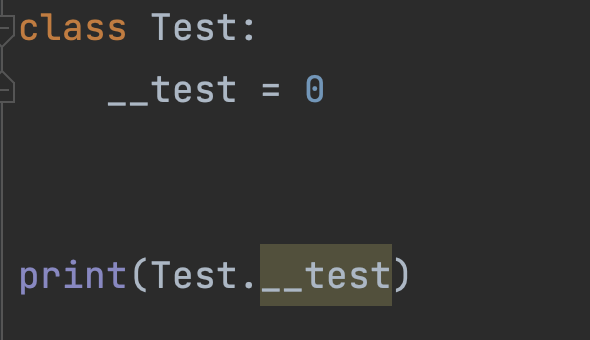
**class MyClass(metaclass=ABCMeta):**

**@abstractmethod**

**def my\_abstract\_method(self):**

**pass**

1. Что будет результатом программы?



**При попытке обратиться к этому полю с помощью `Test.\_\_test` возникнет ошибка, так как поле \_\_test является приватным**

1. Что делает финализатор класса?

**Финализатор класса в Python - это метод с названием `\_\_del\_\_`. Он выполняется автоматически при удалении экземпляра класса сборщиком мусора. Финализатор используется для освобождения ресурсов, таких как файлы или сетевые соединения, перед тем, как объект будет удален из памяти.**

1. Распишите работу магических методов: new, call

**\_\_new\_\_ вызывается до метода "\_\_init\_\_" и отвечает за создание нового экземпляра класса. Он принимает класс в качестве первого аргумента и может принимать дополнительные аргументы для инициализации.**

**\_\_call\_\_ позволяет вызывать экземпляры класса как функции. Этот метод позволяет определить, что должно происходить при вызове экземпляра класса, подобно вызову функции.**

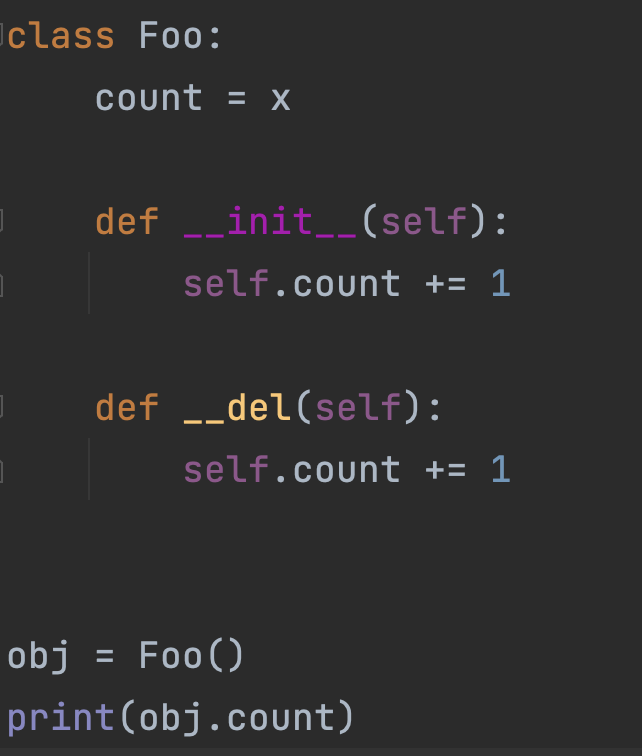
1. Если в классе определены два метода с одинаковыми именами и разными списками параметров, что будет результатом?

**Если в классе определены два метода с одинаковыми именами и разными списками параметров, то результатом будет переопределение первого метода вторым методом. При вызове этого метода будет выполняться код из второго метода, а не из первого.**

1. Значением поля класса по умолчанию может являться?

* **значение переменной**
* **константа**
* **результат вызова функции #Если функция — не метод класса**
* возможность указания значений полей по умолчанию в Python не предусмотрена

1. Укажите результат выполнения скрипта и почему:



**NameError  
  
Атрибут count объявлен неправильно. Переменная х не определена. \_\_del — метод, а не диструктор, т.к. правильное объявление - \_\_del\_\_**

25. Как указать в программе что класс A наследуется от B?

**Ответ: Class A(B):**