

Занятие 11. Объектно-ориентированное программирование. Базовый синтаксис Python

План занятия

1.	Общие понятия объектно-ориентированного программирования	2
1.1	Понятия объекта и класса.....	2
1.2	Преимущества использования классов в программировании.....	2
1.3	Принципы объектно-ориентированного подхода в программировании.	3
2.	Базовый синтаксис Python.....	4
2.1	Лексемы.....	4
2.2	Зарезервированные (ключевые) слова Python.....	5
2.3	Идентификатор.....	5
3.	Вопросы по теме	6

1. Общие понятия объектно-ориентированного программирования

1.1 Понятия объекта и класса

Объект программы обозначает некоторую сущность той части реального или абстрактного мира (предметной области), которая представляет для нас интерес.

Объекты обладают **свойствами** (атрибутами), **состоянием**, **поведением**, могут обмениваться сообщениями.

Абстракция, описывающая множества объектов одной природы, называется **классом** объекта. Другими словами, класс представляет не объект, а **понятие** объекта. Каждый объект должен принадлежать некоторому классу. Объекты по иному называют **экземплярами** класса.

Объекты могут быть **создаваться** и **удаляться** в любом месте программы.

Состояние объекта характеризуется фиксированными значениями его **свойств** (атрибутов). **Метод объекта** задает способ изменения состояния объекта. Изменение состояния объекта называется **событием**.

Поведение объекта – это последовательность событий от некоторого начального события до некоторого конечного события.

Различают **простые** (атомарные) объекты и **составные** объекты. Простые объекты, в отличие от составных объектов не имеют структуры, т. е. не содержат других объектов.

Каждому классу, объекту, свойству и методу объекта присваивают уникальный **идентификатор** (имя). Идентификаторы, обозначающие классы, объекты, свойства и методы объекта называют **переменными**.

1.2 Преимущества использования классов в программировании

Использование классов предоставляет преимущества абстрактного подхода в программировании.

Полиморфизм (перегрузка): в разных объектах одна и та же операция может выполнять различные функции. Слово «полиморфизм» имеет греческую природу и означает «имеющий многие формы». Простым примером полиморфизма может служить оператор плюс, выполняющийся для различных типов объектов, таких как, числа и строки.

Инкапсуляция: сокрытие внутренних механизмов функционирования объекта. Это второй основной принцип абстракции.

Наследование: для любого класса может быть определен **дочерний** или **производный** класс. Производный класс содержит все свойства базового класса, и, в зависимости от необходимости, в нем добавлены или переопределены новые методы и свойства. Это позволяет избежать написания повторного кода.

Композиция: объект может быть составным и включать в себя другие объекты.

1.3 Принципы объектно-ориентированного подхода в программировании

Объектно-ориентированный подход в программировании подразумевает построение объектной модели предметной области:

1. Описывается проблема с помощью обычного языка с использованием понятий, действий, прилагательных.
2. На основе понятий формулируются классы.
3. На основе действий проектируются методы.
4. На основе прилагательных определяются атрибуты.
5. На выбранном языке программирования реализуются методы и атрибуты.

В некоторых языках программирования используют модификаторы доступа к переменным: public, protected, internal, private.

Модификаторы доступа `public`, `protected`, `internal` или `private` используются для указания одного из следующих объявленных уровней доступности к членам.

Объявленная доступность	Значение
<code>public</code>	Неограниченный доступ.
<code>protected</code>	Доступ ограничен содержащим классом или типами, которые являются производными от содержащего класса.
<code>internal</code>	Доступ ограничен текущей сборкой.
<code>protected internal</code>	Доступ ограничен текущей сборкой или типами, которые являются производными от содержащего класса.
<code>private</code>	Доступ ограничен содержащим классом.

2. Базовый синтаксис Python

2.1 Лексемы

Символы алфавита любого языка программирования образуют лексемы. Если декларация кодировки отсутствует, то, по умолчанию, для построения лексем используются символы кодировки UTF-8. Лексема (token) — это минимальная единица языка, имеющая самостоятельный смысл. Лексемы — формируют базовый словарь языка, понятный компилятору. Если исходный файл не может быть расшифрован, то будет выдано сообщение об ошибке.

Всего существует пять видов лексем:

Ключевые слова (keywords)

Идентификаторы (identifiers)

Литералы (literals)

Операции (operators)

Знаки пунктуации (разделители, punctuators)

2.2 Зарезервированные (ключевые) слова Python

Зарезервированные слова нельзя использовать в качестве имени переменной или любого другого идентификатора. Все **ключевые слова Python** состоят только из букв в нижнем регистре. Получить список ключевых слов в PyCharm возможно командой:

```
import pydoc
print(pydoc.help.keywords)
```

Список зарезервированных слов приведен ниже:

and	elif	if	print
as	else	import	raise
assert	except	in	return
break	exec	is	try
class	finally	lambda	while
continue	for	not	which
def	from	or	yield
del	global	pass	

2.3 Идентификатор

Идентификатор переменной в Python должен начинаться с символа латинского алфавита (от а до Z) или со знака подчеркивания (`_`), после которых может идти произвольное количество букв, знаков подчеркивания и чисел (от 0 до 9).

В **Python** недопустимо использование знаков препинания или специальных символов, таких как `@`, `$` или `%` в качестве идентификаторов. Кроме того, **Python** чувствителен к регистру, то есть `cat` и `Cat` это два разных имени.

В Python существуют следующие правила для идентификаторов:

1. Имена классов начинаются с **Прописной** буквы, все остальные идентификаторы - со **строчной** буквы.
2. Использование знака подчеркивания в качестве первого символа идентификатора означает, что данный идентификатор является `private` (частным - закрытым от использования вне класса).
3. Если идентификатор начинается и заканчивается двумя знаками подчеркивания (например, `__init__`) это означает, что он является специальным именем, определенным внутри языка;
4. Нельзя использовать два подряд символа нижнего подчеркивания в начале и в конце идентификатора;

Примеры.

1. Использование отступов для обозначения блоков циклов

```
begin_fill()
for i in range(5):
    fd(150)
for j in range(5):
    fd(50)
    left(144)
left(144)
end_fill()
```

2. Объединение нескольких физических строк

```
if 1900 < year < 2100 and 1 <= month <= 12 \
and 1 <= day <= 31 and 0 <= hour < 24 \
and 0 <= minute < 60 and 0 <= second < 60: # Успешная проверка даты
    return 1
```

3. Вопросы по теме

Изложите общие понятия объектно-ориентированного программирования

Объясните понятия объекта и класса

Обоснуйте преимущества использования классов в программировании

Перечислите принципы объектно-ориентированного подхода в программировании

Изложите отличительные признаки синтаксиса Python

Дайте определение лексемы

Объясните назначение зарезервированных (ключевых) слов Python

Объясните назначение и правила для идентификаторов