Основні поняття ООП: класи, об'єкти, атрибути, методи

GENIUS space

Основні поняття ООП:
класи, об'єкти, атрибути, методи

1. Класи (Classes):

- Класи є шаблонами або формами, за допомогою яких визначаються об'єкти.
- Вони описують структуру та поведінку об'єктів, які належать до цього класу.
- Класи містять атрибути (змінні) та методи (функції), які можна використовувати для створення та взаємодії з об’єктами.

```python
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def greet(self):
        return f"Hello, my name is {self.name} and I am {self.age} years old."
```

2. Об'єкти (Objects):

- Об'єкти - це конкретні екземпляри класів, створені на основі класу.
- Вони мають власні атрибути та можуть викликати методи, визначені у відповідному класі.
- Об'єкти реалізують концепцію інкапсуляції, яка дозволяє зберігати дані та функції, які їх опрацьовують, разом у одному об’єкті.

```python
person1 = Person("Alice", 30)
person2 = Person("Bob", 25)
```

GENIUS space

3. Атрибути (Attributes):

- Атрибути - це змінні, які визначені у класі та призначені для зберігання даних, пов'язаних з об'єктами класу.
- Вони визначають стан об'єкта та представляють його характеристики.
- Атрибути можуть бути публічними, приватними або захищеними, залежно від їх видимості.

```
print(person1.name)  # Виведе: Alice
print(person2.age)   # Виведе: 25
```

4. Методи (Methods):

- Методи - це функції, які визначені у класі та призначені для виконання операцій над об'єктами цього класу.
- Вони реалізують поведінку об'єктів класу та використовують атрибути для виконання різних операцій.
- Методи можуть бути публічними (доступними ззовні), приватними (доступними лише в межах класу) або захищеними (доступними для підкласів).

```
print(person1.greet())  # Виведе: Hello, my name is Alice and I am 30 years old.
```

Ініціалізація об'єктів через конструктор

Ініціалізація об'єктів через конструктор - це процес надання початкових значень атрибутам об'єкта під час його створення. У багатьох об'єктно-орієнтованих мовах програмування, таких як Python, це зазвичай реалізовано за допомогою спеціального методу, відомого як конструктор класу.
У Python цей метод має ім'я **__init__**.

```python
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

person1 = Person("Alice", 30)
person2 = Person("Bob", 25)

print(person1.name)  # Виведе: Alice
print(person1.age)   # Виведе: 30

print(person2.name)  # Виведе: Bob
print(person2.age)   # Виведе: 25
```

У цьому прикладі конструктор **__init__** класу **Person** приймає два аргументи: **name** і **age**. Після створення об'єкта класу **Person**, конструктор встановлює значення цих атрибутів за допомогою переданих аргументів.

При створенні об'єктів **person1** і **person2** ми передаємо їхні імена і вік у конструктор, і ці значення зберігаються в атрибутах **name** і **age** відповідно.

Конструктори дозволяють задати початковий стан об'єктів та ініціалізувати їх атрибути, щоб об'єкти були готові до використання при їхньому створенні.