Практическое занятие № 16

Тема: составление программ с использованием ООП.

Цели практического занятия: закрепить усвоенные знания, понятия ,алгоритмы, основные принципы составления программ, приобрести навыки составление программ с ООП в IDE Py CharmCommunity.

Постановка задачи 1: Создайте класс "Человек" с атрибутами "имя", "возраст" и "пол". Напишите метод, который выводит информацию о человеке в формате "Имя: имя, Возраст: возраст, Пол: пол".

Код:

```
#Создайте класс "Человек" с атрибутами "имя", "возраст" и "пол".
Напишите метод,
#который выводит информацию о человеке в формате "Имя: имя, Возраст:
возраст,
#Пол: пол".
class Person:
    """Class to represent a person with attributes and a method to
display information."""

    def __init__(self, name, age, gender):
        """Initialize the person's attributes."""
        self.name = name
        self.age = age
        self.gender = gender

    def print_person_info(self):
        """Display the person's information in a formatted way."""
        print(f"Имя: {self.name}")
        print(f"Возраст: {self.age}")
        print(f"Пол: {self.gender}")


# Create a Person object
person1 = Person("Иван Петров", 35, "мужчина")

# Display the person's information
person1.print_person_info()
```

Постановка задачи 2: Создание базового класса "Животное" и его наследование для создания классов "Собака" и "Кошка". В классе "Животное" будут общие методы, такие как "дышать" и "питаться", а классы-наследники будут иметь свои уникальные методы и свойства, такие как "гавкать" и "мурлыкать".

Код:

```
class Animal:
    """Base class for all animals."""

    def __init__(self, name, species):
        self.name = name
        self.species = species

    def breathe(self):
        print(f"{self.name} is breathing.")

    def eat(self):
        print(f"{self.name} is eating.")


class Dog(Animal):
    """Class representing a dog."""

    def __init__(self, name, breed):
        super().__init__(name, "dog")
        self.breed = breed

    def bark(self):
        print(f"{self.name} says Woof!")


class Cat(Animal):
    """Class representing a cat."""

    def __init__(self, name, breed):
        super().__init__(name, "cat")
        self.breed = breed

    def purr(self):
        print(f"{self.name} is purring.")


# Create an Animal object
animal1 = Animal("Rex", "dinosaur")

# Create a Dog object
dog1 = Dog("Max", "Labrador Retriever")
```

```
# Create a Cat object
cat1 = Cat("Whiskers", "Siamese")

# Call methods on each object
animal1.breathe()
animal1.eat()

dog1.bark()
dog1.breathe()  # Inherited method from Animal

cat1.purr()
cat1.eat()  # Inherited method from Animal
```

Постановка задачи 3: Для задачи из блока 1 создать две функции, save_def и load_def, которые позволяют сохранять информацию из экземпляров класса (3 шт.) в файл и загружать ее обратно. Использовать модуль pickle для сериализации и десериализации объектов Python в бинарном формате

Код:

```
import pickle

class Person:
    """Class to represent a person with attributes and a method to
display information."""

    def __init__(self, name, age, gender):
        """Initialize the person's attributes."""
        self.name = name
        self.age = age
        self.gender = gender

    def print_person_info(self):
        """Display the person's information in a formatted way."""
        print(f"Имя: {self.name}")
        print(f"Возраст: {self.age}")
        print(f"Пол: {self.gender}")


def save_def(persons, filename):
    """Saves a list of Person objects to a file using pickle."""
    with open(filename, "wb") as file:
        pickle.dump(persons, file)
```

```python
def load_def(filename):
    """Loads a list of Person objects from a file using pickle."""
    with open(filename, "rb") as file:
        loaded_persons = pickle.load(file)
    return loaded_persons


# Create Person objects
person1 = Person("Иван Петров", 35, "мужчина")
person2 = Person("Мария Иванова", 28, "женщина")
person3 = Person("Сергей Кузнецов", 42, "мужчина")

persons_list = [person1, person2, person3]

# Save the list of Person objects to a file
save_def(persons_list, "persons.pickle")

# Load the list of Person objects from the file
loaded_persons = load_def("persons.pickle")

# Print the information of the loaded Person objects
for person in loaded_persons:
    person.print_person_info()
```

Вывод: я закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ с ООП в IDE PyCharm Community.