Студент группы ИС-22 Васин Д.Е.

Практическое занятие №16

Тема: составление программ с использованием ООП.

Цель: закрепить усвоенные знания, понятия, алгоритмы,основные принципы составления программ, приобрести навыки составления программ с ООП в IDE PyCharm Community.

Постановка задачи №1: Создайте класс «Счетчик», который имеет атрибут текущего значения и методы для инкремента и декремента значения

Тип алгоритма: линейный

Текст программы:

```
"""
Создайте класс «Счетчик», который имеет атрибут текущего значения и методы для
инкремента и декремента значения
"""
class Counter:
    def __init__(self):
        self.value = 0

    def increment(self):
        self.value += 1

    def decrement(self):
        self.value -= 1

counter = Counter()
# print("Стартовое значение:", counter.value)
# counter.increment()
# print("Инкремент:", counter.value)
```

```
# counter.decrement()
# print("Декремент:", counter.value)
```

Протокол работы программы:

Стартовое значение: 0

Инкремент: 1

Декремент: -1

Process finished with exit code 0

Постановка задачи №2: Создайте класс "Автомобиль", который содержит информацию о марке, модели и годе выпуска. Создайте класс "Грузовик", который наследуется от класса "Автомобиль" и содержит информацию о грузоподъемности. Создайте класс "Легковой автомобиль", который наследуется от класса "Автомобиль" и содержит информацию о количестве пассажиров

Тип алгоритма: линейный

Текст программы:

```
"""
Создайте класс "Автомобиль", который содержит информацию
о марке, модели и
годе выпуска. Создайте класс "Грузовик", который
наследуется от класса
"Автомобиль" и содержит информацию о грузоподъемности.
Создайте класс
"Легковой автомобиль", который наследуется от класса
"Автомобиль" и содержит
информацию о количестве пассажиров
"""
class Car:
    def __init__(self, brand, model, year):
```

```
        self.brand = brand
        self.model = model
        self.year = year

    def __str__(self):
        return f"{self.brand} {self.model} ({self.year})"
# Выводим в виде строки

class Truck(Car):
    def __init__(self, brand, model, year, payload):
        super().__init__(brand, model, year)
        self.payload = payload

    def __str__(self):
        return f"{super().__str__()} - Грузоподъемность:
{self.payload} тонн" # Выводим в виде строки

class PassengerCar(Car):
    def __init__(self, brand, model, year,
num_passengers):
        super().__init__(brand, model, year)
        self.num_passengers = num_passengers

    def __str__(self):
        return f"{super().__str__()} - Вместимость:
{self.num_passengers} человек" # Выводим в виде строки

myCar = PassengerCar("Toyota", "Camry 40", 2018, 5)
myTruck = Truck("Mercedes", "Actross", 2020, 20)
```

```
print(myCar)
print(myTruck)
```

Протокол работы программы:

Toyota Camry 40 (2018) - Вместимость: 5 человек

Mercedes Actross (2020) - Грузоподъемность: 20 тонн

Process finished with exit code 0

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.

Тип алгоритма: линейный

Текст программы:

```
import pickle


class Counter:
    def __init__(self, initial_value=0):
        self.value = initial_value

    def increment(self):
        self.value += 1
        return self.value

    def decrement(self):
        self.value -= 1
        return self.value

def save_def(counters, filename):
```

```
    with open(filename, 'wb') as f:
        pickle.dump(counters, f)

def load_def(filename):
    with open(filename, 'rb') as f:
        return pickle.load(f)

counter1 = Counter(5)
counter2 = Counter(10)
counter3 = Counter(15)

counters = [counter1, counter2, counter3]
save_def(counters, 'counters.bin')

loaded_counters = load_def('counters.bin')

for counter in loaded_counters:
    print(f"Стартовое значение: {counter.value},
Инкремент: {counter.increment()}, Декремент:
{counter.decrement()} \n")
```

Протокол работы программы:

Стартовое значение: 5, Инкремент: 6, Декремент: 5

Стартовое значение: 10, Инкремент: 11, Декремент: 10

Стартовое значение: 15, Инкремент: 16, Декремент: 15

Process finished with exit code 0

Вывод:

В процессе выполнения практического занятия №16, я закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрел навыки составления программ с ООП в IDE PyCharm Community.