Практическое занятие №16

Тема: Составление программы с использование ООП

Цель: Закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки работы с ООП в IDE Pycharm

Задание 1

Постановка задачи.

Создайте класс Банк, который имеет аттрибуты суммы денег и процентной ставки. Добавьте методы для вычисления процентных начислений и снятия денег

Текст программы:

```
# Создайте класс Банк, который имеет аттрибуты суммы денег и процентной
ставки.
# Добавьте методы для вычисления процентных начислений и снятия денег

class Bank:
    def __init__(self, percent: float, money: float):
        self.percent = percent
        self.money = money

    def payment(self):
        newmoney = round(self.money * (1 + self.percent/100), 2) - self.money
        print("начисления: ", round(newmoney, 2))
        self.money += newmoney
        print("процент зачислен, на счету:", self.money)

    def take_money(self, amount: float):
        if self.money >= amount:
            self.money -= amount
            print("деньги сняты, на счету осталось:", self.money)
            return amount
        else:
            print("недостаточно средств на счету")
            return 0


if __name__ == "__main__":
    score = Bank(14.5, 300)
    score.payment()
    score.take_money(220)
    score.payment()
    score.payment()
```

Протокол работы:

начисления: 43.5

процент зачислен, на счету: 343.5

деньги сняты, на счету осталось: 123.5

начисления: 17.91

процент зачислен, на счету: 141.41

начисления: 20.5

процент зачислен, на счету: 161.91

Process finished with exit code 0

Задание 2

Постановка задачи.

Создайте класс Фрукт, который содержит информацию о наименовании и весе фрукта. Создайте классы Яблоко и Апельсин, которые наследуются от класса Фрукт и содержат информацию о цвете

Текст программы:

```
# Создайте класс Фрукт, который содержит информацию о наименовании и весе
фрукта. Создайте классы Яблоко и Апельсин,
# которые наследуются от класса Фрукт и содержат информацию о цвете

class Fruit:
    def __init__(self, name, weight):
        self.name = name
        self.weight = weight
        self.color = None

    def get_information(self):
        print(f'название {self.name}, вес {self.weight}{", цвет " +
self.color if self.color is not None else ""} ')


class Apple(Fruit):
    def __init__(self, weight, color):
        super().__init__('Яблоко', weight)
        self.color = color


class Orange(Fruit):
    def __init__(self, weight, color):
        super().__init__('яблоко', weight)
        self.color = color


banana = Fruit('банан', 10)
apple = Apple('24', 'красный')
orange = Orange('13', 'оранжевый')

banana.get_information()
```

```
apple.get_information()
orange.get_information()
```

Протокол работы:

название банан, вес 10

название Яблоко, вес 24, цвет красный

название яблоко, вес 13, цвет оранжевый

Process finished with exit code 0

Задание 3

Постановка задачи.

Для класса из 1 задания создать функции save и load, позовляющие сохранять информацию из экземпляров класса в файл и загружать обратно.
Использовать pickle для сериализации и десериализации объектов Python в бинарном формате

Текст программы:

```
# Для класса из 1 задания создать функции save и load, позовляющие сохранять
информацию из экземпляров класса в файл и
# загружать обратно. Использовать pickle для сериализации и десериализации
объектов Python в бинарном формате

import pickle
import PZ_16_1


class SaveBank(PZ_16_1.Bank):
    def save(self, name):
        with open(f"{name}.bin", "wb") as fl:
            pickle.dump(self.money, fl)
            pickle.dump(self.percent, fl)

    def load(self, name):
        with open(f"{name}.bin", "rb") as fl:
            self.money = pickle.load(fl)
            self.percent = pickle.load(fl)


score1 = SaveBank(14.5, 300)
score1.save('score1.bin')

score1backup = SaveBank(0, 0)
score1backup.load('score1.bin')
print(score1backup.money, score1backup.percent)


apple.get_information()
orange.get_information()
```

Протокол работы:

300 14.5

Process finished with exit code 0

Вывод: в процессе выполнения практических заданий выработал навыки составления программ с использованием ООП в IDE Pycharm. Были использованы конструкции class, __init__, __name__. Выполнены разработка кода, тестирование, отладка. Готовые коды программ выложены на GitHub