

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ И ТЕЛЕКОММУНИКАЦИЙ
КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ

Лабораторная работа № 1
по дисциплине
«Информационные технологии и программирования»

Выполнил:
Попов Антон Андреевич
Студент 1 курса группы ПИН-б-о-22-1
Направления подготовки
09.03.03 Прикладная информатика
очной формы обучения

Ставрополь, 2023 г.

Тема: Основы объектно-ориентированного программирования на ЯП Python

Цель работы: изучить базовые понятия (классы, подклассы и методы). Реализовать фундаментальные принципы объектно-ориентированного программирования.

Ход работы

Вариант - 17

№ 4.3.1 Римское число. Листинг приведён в файлах:

[main.py](#)

[roman.py](#)

[test_roman.py](#)

Также приведена [UML-диаграмма](#) проекта.

В модуле *roman.py* были доработаны несколько методов у класса *Roman*, включая обработку некорректных входных данных. В основном приложении реализован простой тест работоспособности класса *Roman*.

№ 4.3.2 Пиццерия. Листинг приведён в файлах:

[main.py](#)

[заказ.py](#)

[пицца.py](#)

[терминал.py](#)

[test_заказ.py](#)

[test_пицца.py](#)

[test_терминал.py](#)

Также приведена [UML-диаграмма](#) проекта.

Реализованы модули *заказ.py*, *пицца.py*, *терминал.py*. В их классах были реализованы методы и связи между друг другом. В основном приложении реализован алгоритм пиццерии, указанный в задании.

№ 4.3.3 Банковские вклады. Листинг приведён в файлах:

[main.py](#)

[deposit.py](#)

[test_time_deposit.py](#)

[test_bonus_time_deposit.py](#)

[test_compound_time_deposit.py](#)

Также приведена [UML-диаграмма](#) проекта.

Реализован базовый класс *TimeDeposit*, на его основе были реализованы ещё два класса: *BonusTimeDeposit* и *CompoundTimeDeposit*. В основном приложении реализован алгоритм для вычисления прибыли от каждого из вкладов на заданный срок и заданную сумму.

№ 4.3.4 Простой класс. Листинг приведён в файлах:

[main.py](#)

[stack.py](#)

[test_stack.py](#)

Также приведена [UML-диаграмма](#) проекта.

Реализован класс *Stack* с основными его базовыми методами.

№ 4.3.5 Класс-контейнер. Листинг приведён в файлах:

[main.py](#)

[stack_collection.py](#)

[test_stack_collection.py](#)

Также приведена [UML-диаграмма](#) проекта.

Реализован класс *StackCollection* на основе класса *Stack*, а также ряд основных методов для взаимодействия с коллекцией математических промежутков, таких как добавление/удаление/получение элемента, json импорт/экспорт, сравнение коллекций и другие методы.

№ 4.3.6 Иерархия классов. Листинг приведён в файлах:

[main.py](#)

[writing_affiliation.py](#)

[pencil.py](#)

[pen.py](#)

[gel_pen.py](#)

[test_writing_affiliation.py](#)

[test_pencil.py](#)

[test_pen.py](#)

[test_gel_pen.py](#)

Также приведена [UML-диаграмма](#) проекта.

Реализован базовый класс *WritingAffiliation*, а также на его основе классы *Pen*, *Pencil*, *GelPen*. Выстроена иерархия наследования и реализованы показательные примеры методов.

Ссылка на [репозиторий](#), содержащий полностью выполненные задания.

Вывод: изучил базовые понятия (классы, подклассы и методы) Реализовал фундаментальные принципы объектно-ориентированного программирования.