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

«Объектно-ориентированное программирование»

**Выполнил:**

Ермолаев Кирилл Александрович

Студент 2 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).

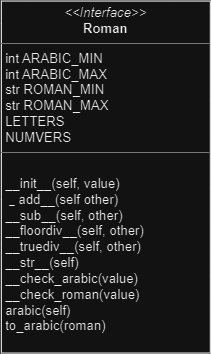
Реализовать фундаментальные принципы объектно-ориентированного программирования.

**Ход работы:**

4.3.1. Римское число

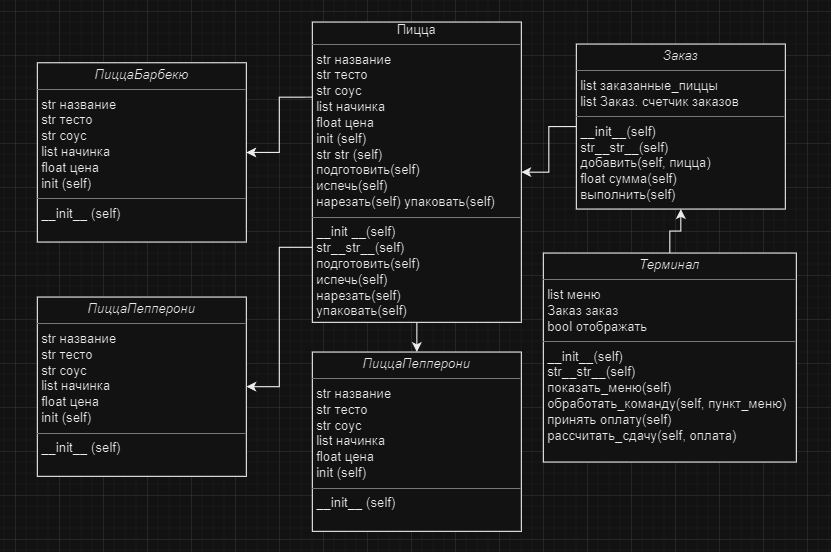
# main.py

from roman import Roman  
  
if \_\_name\_\_ == "\_\_main\_\_":  
  
 r1 = Roman("X")  
 r2 = Roman(5)  
  
 print(" Числа:", r1, r2, r1.arabic, r2.arabic)  
 print(" Сумма:", r1 + r2)  
 print(" Разность:", r1 - r2)  
 print("Произведение:", r1 \* r2)  
 print(" Частное:", r1 // r2)  
  
 print("\nПреобразование без создания объекта:")  
 print(2016, "=", Roman.to\_roman(2016))  
 print("MMXVI", "=", Roman.to\_arabic("MMXVI"))



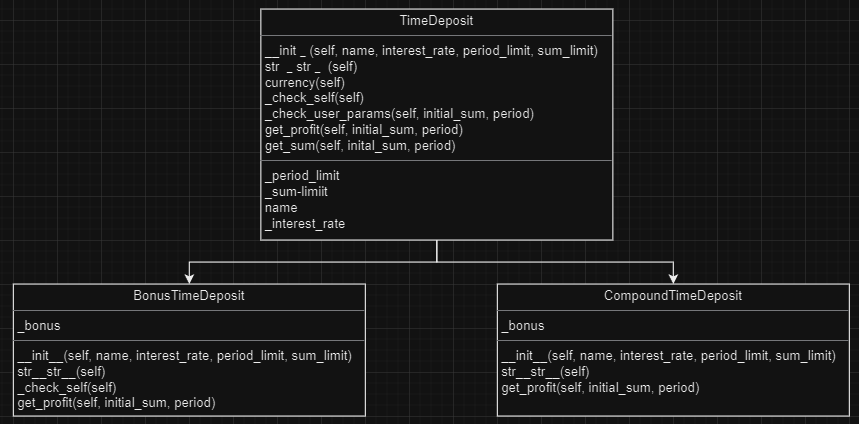
4.3.2. Пицерия

from терминал import Терминал  
  
if \_\_name\_\_ == "\_\_main\_\_":  
  
 терминал1 = Терминал()  
 print(терминал1)  
 while True:  
 терминал1.показать\_меню()  
 пункт\_меню = input()  
 терминал1.обработать\_команду(пункт\_меню)



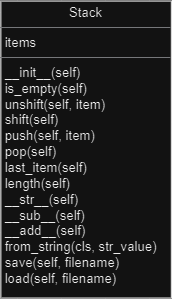
4.3.3. Банковские вклады

from deposit import deposits  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 print("Добро пожаловать в систему подбора вкладов!")  
  
 while True:  
 print("\n-----")  
 print("Нажмите 1, чтобы подобрать вклад, или что угодно для выхода.")  
  
 answer = input()  
 if answer == "1":  
  
 initial\_sum = float(input("1/2: Введите начальную сумму вклада: "))  
 period = int(input("2/2: Введите срок вклада (мес.): "))  
  
 matched\_deposits = []  
 for deposit in deposits:  
 try:  
 deposit.\_check\_user\_params(initial\_sum, period)  
 matched\_deposits.append(deposit)  
 except AssertionError as err:  
 pass  
  
 if len(matched\_deposits) > 0:  
 print("{0:18} | {1:13} | {2:13}".format(  
 "Вклад", "Прибыль", "Итоговая сумма"  
 ))  
 for deposit in matched\_deposits:  
 print("{0:18} | {1:8,.2f} {3:4} | {2:8,.2f} {3:4}".format(  
 deposit.name,  
 deposit.get\_profit(initial\_sum, period),  
 deposit.get\_sum(initial\_sum, period),  
 deposit.currency))  
 else:  
 print("К сожалению, нет подходящих Вам вкладов.")  
  
 else:  
 break  
  
 print("\nСпасибо, что воспользовались терминалом банка! До встречи!")



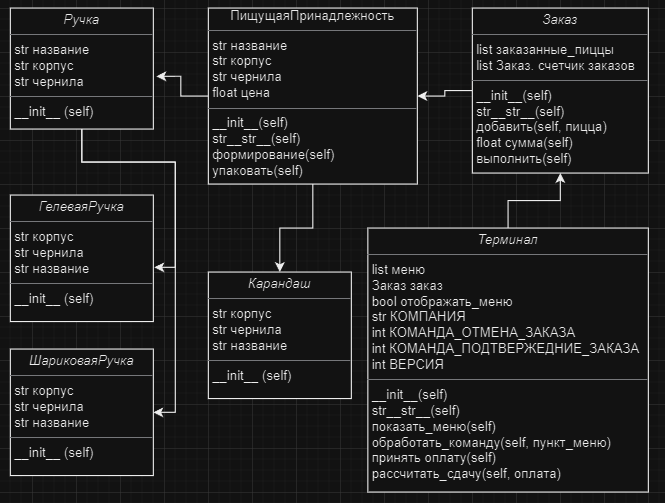
4.3.4. Простой класс

import json  
  
class Stack:  
 def \_\_init\_\_(self, \*args):  
 *"""  
 Инициализация пустого стека.  
 """* self.items = [\*args]  
  
 def is\_empty(self):  
 *"""  
 Проверка на пустоту стека.  
 """* return len(self.items) == 0  
   
 def unshift(self, item):  
 *"""  
 Добавляет элемент в начало стека.  
 """* self.items.insert(0, item)  
  
 def shift(self):  
 *"""  
 Забирает первый элемент стека и возвращает его.  
 """* if not self.is\_empty():  
 return self.items.pop(0)  
 else:  
 raise IndexError("Стек пуст.")  
  
 def push(self, item):  
 *"""  
 Добавляет элемент в конец стека.  
 """* self.items.append(item)  
  
 def pop(self):  
 *"""  
 Забирает последний элемент стека и возвращает его.  
 """* if not self.is\_empty():  
 return self.items.pop()  
 else:  
 raise IndexError("Стек пуст.")  
  
 def last\_item(self):  
 *"""  
 Возвращает элемент с конца стека без его удаления.  
 """* if not self.is\_empty():  
 return self.items[-1]  
 else:  
 raise IndexError("Стек пуст.")  
  
 def length(self):  
 *"""  
 Возвращает количество элементов в стеке.  
 """* return len(self.items)  
  
 def \_\_str\_\_(self):  
 *"""  
 Возвращает строковое представление стека.  
 """* return str(self.items)  
  
 @classmethod  
 def from\_string(cls, str\_value):  
 *"""  
 Создает объект на основе строки str\_value.  
 """* items = [int(item) for item in str\_value.split(',')]  
 return cls(\*items)  
  
 def save(self, filename):  
 *"""  
 Сохраняет объект в JSON-файл filename.  
 """* with open(filename, 'w') as file:  
 json.dump({'items': self.items}, file)  
  
 def load(self, filename):  
 *"""  
 Загружает объект из JSON-файла filename.  
 """* with open(filename, 'r') as file:  
 data = json.load(file)  
 self.items = data['items']



4.3.6. Иерархия классов

from терминал import Терминал  
  
if \_\_name\_\_ == "\_\_main\_\_":  
  
 терминал1 = Терминал()  
 print(терминал1)  
 while True:  
 терминал1.показать\_меню()  
 пункт\_меню = input()  
 терминал1.обработать\_команду(пункт\_меню)



<https://github.com/MoeTomatoki/OOP>

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