Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ульяновский государственный технический университет»

Кафедра «Вычислительная техника»

Разработка профессиональных приложений

(название дисциплины)

**Отчёт по лабораторной работе №4**

«Классы»

(название (тема) работы)

**Вариант №11**

Выполнил:

студент группы ИВТАПбд-21

Хованский А.П.

Проверил(а):

преподаватель кафедры «ВТ»

Исхаков И. И.

Ульяновск

2023

**Постановка задачи**

Необходимо переделать лабораторную работу №3 с использованием классов, описывающих предметную область, заданную вариантом, с реализацией следующих особенностей (вполне возможно, что предлагаемое в 3 лабе задание для этого нужно будет расширить):

1. Класс должен содержать итератор

2. Должна быть реализована перегрузка стандартных операций (repr, например)

3. Должно быть реализовано наследование

4. Запись значений в свойства - только через \_\_setattr\_\_

5. Возможность доступа к элементам коллекции по индексу (\_\_getitem\_\_)

6. Должны быть реализованы статические методы

7. Должны быть реализованы генераторы

**Описание реализации**

За основу был взят исходный код лабораторной работы №3. Был реализован родительский класс человек, в котором реализованы метод инициализации экземпляров класса, методы сеттер для получения имени, пола и возраста, а также методы геттер, для изменения этих свойств. Был реализован дочерний класс пользователь, наследованный от родительского класса человек. В этом классе был реализован переопределенный метод вывода информации об объекте класса пользователь, реализован генератор для перебора списка дат совершения операций. Также бал реализован класс кастомного итератора для класса пользователь. Далее по аналогии с третьей лабораторной работой, открываем csv-файл для чтения, далее с помощью класса DictReader модуля csv создается объект, который работает как обычный reader(), но отображает информацию в каждой строке в качестве словаря dict. Далее создаем объект класса пользователя и добавляем его в список, сортируем список при помощи лямбда-функции. Создаем файл для записи и заносим туда результат.

**Исходный код реализации**

import os  
from pathlib import Path  
import csv  
  
def countFiles():  
 fileName = input("Введите путь к папке: ")  
 file = Path(fileName)  
 files = os.listdir(file)  
 print(f"В данной папке содержатся следующие объекты: {files}\nИх количество: {len(files)}")  
  
  
  
def get\_posts(posts\_history):  
 print("post number;nickname;post text;number of likes")  
 with open("Post.csv", "r", encoding='utf-8') as f:  
 reader = csv.reader(f)  
 for row in reader:  
 print(\*row)  
 posts\_history[row[0]] = row[1:]  
  
  
def sort\_posts\_number(posts\_history):  
 print("\nОтсортируем по количеству лайков: ")  
 for elem in sorted(posts\_history.items(), key = lambda para: int(para[1][2])):  
 print(elem[0], \*elem[1])  
  
  
def sort\_posts\_str(posts\_history):  
 print("\nОтсортируем по тексту поста: ")  
 for elem in sorted(posts\_history.items(), key = lambda para: para[1][1]):  
 print(elem[0], \*elem[1])  
 print("")  
  
  
def output\_by\_criterion(posts\_history):  
 criterion = input("Введите количсество лайков: \n")  
 print(f'\nСтроки, в которых количество лайков больше "{criterion}": ')  
 for elem in posts\_history:  
 if posts\_history[elem][2] >= criterion:  
 print(elem, \*posts\_history[elem])  
  
  
def write\_to\_csv(posts\_history):  
 newKey = input("Введите номер поста: ")  
 new\_nickname = input("Введите никнейм пользователя который создал пост: ")  
 newtextpost = input("Введите текст поста: ")  
 new\_number\_of\_likes = input("Введите количество лайков поста: ")  
 posts\_history[newKey] = [new\_nickname, newtextpost, new\_number\_of\_likes]  
 with open('Post.csv', 'w', encoding='utf-8') as f:  
 for elem in posts\_history:  
 f.write(elem + ',' + posts\_history[elem][0] + ',' + posts\_history[elem][1] + ',' + posts\_history[elem][2] + '\n')  
  
  
def main():  
 countFiles()  
 posts\_history = {}  
 get\_posts(posts\_history)  
 sort\_posts\_number(posts\_history)  
 sort\_posts\_str(posts\_history)  
 output\_by\_criterion(posts\_history)  
 write\_to\_csv(posts\_history)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

**Тестирование**



Рис. 1 Выходной файл

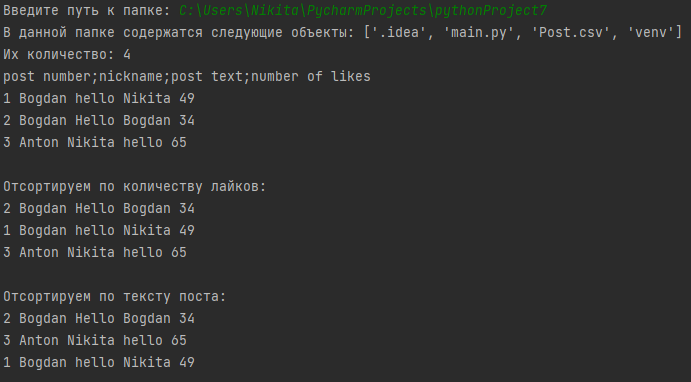


Рис. 2 Вывод в консоль

**Описание возникших затруднений**

При выполнении данной лабораторной работы возникли затруднения, так как я редко использую ООП, мне нужно было основательно повторить материал. Также я не знал, что такое генераторы и с нуля изучал данную тему.

**Вывод**

В ходе выполнения лабораторной работы, были улучшены навыки работы с классами, приобретен опыт работы с ООП на языке программирования Python. Данную работу можно считать выполненной, так как все требования были выполнены, написанная программа работает исправно и выполняет все поставленные задачи.