Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 3 з дисципліни

«Основи програмування

2. Модульне програмування»

«Класи та об’єкти»

Виконав студент ІП-14 Нікулін Павло Юрійович
(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська Ірина Іванівна
(прізвище, ім'я, по батькові)

Київ 2022

Лабораторна робота №3

Класи та об’єкти

Мета: вивчити механізми створення і використання класів та об’єктів.

Хід роботи

Задача:

Розробити клас «Студент», який характеризується ПІБ студента, номером його групи, датою народження (у форматі ММ-ДД-РРРР). Створити масив об’єктів даного класу. Визначити найстаршого студента вказаної групи (на вказану дату).

Розв’язання

1. **Постановка задачі**: результатом роботи має бути об’єкт класу «Студент», що характеризується даними найстаршого студента. Створимо початковий список студентів, з можливістю додати себе для користувача. З нового списку виберемо усіх студентів вказаної користувачем групи і визначимо найстаршого з них.

2. Побудова **математичної моделі**. Складемо таблицю імен змінних.

Змінна	Тип	Ім’я	Призначення
Файл з функціями	Файловий	FUNCS.PY	Файл
Список студентів	Список	STUDENTS	Початкове дане
Створення списку студентів	Універсальний	CREATE_LIST	Функція
Додавання користувача	Універсальний	ADD_YOURSELF	Функція
Пошук найстаршого	Універсальний	OLDEST_STUDENT	Функція
Клас студента	Клас	STUDENT	Початкове дане
Підклас ПІБ	Підклас	FULLNAME	Початкове дане
Підклас дати народження	Підклас	BIRTHDAY	Початкове дане
Конструктор класів	Універсальний	__INIT__	Метод
Виведення інформації про студента	Універсальний	SHOW	Атрибут/функція
Група	Рядковий	GROUP	Атрибут
Прізвище	Рядковий	SURNAME	Атрибут

Ім’я	Рядковий	NAME	Атрибут
По-батькові	Рядковий	PATRONYMIC	Атрибут
Місяць народження	Рядковий	MONTH	Атрибут
День народження	Рядковий	DAY	Атрибут
Рік народження	Рядковий	YEAR	Атрибут
Частина конструктора	Класовий	SELF	Проміжне дане
Студент	Класовий	S1 – S7	Початкове дане
Список студентів	Список	S_LIST	Початкове дане
Інформація про користувача	Класовий	USER	Проміжне дане
Список груп	Список	GROUPS_LIST	Проміжне дане
Обрана група	Рядковий	GROUP_NAME	Проміжне дане
Список студентів обраної групи	Список	GROUP_STUDENTS	Проміжне дане
Пошук за роком	Рядковий	OLDEST_YEAR	Проміжне дане
Пошук за місяцем	Рядковий	OLDEST_MONTH	Проміжне дане
Пошук за днем	Рядковий	OLDEST_DAY	Проміжне дане
Найстарший студент обраної групи	Класовий	OLDEST	Результат

Випробування коду

Код

main.py

```python
from funcs import *

students = []
create_list(students)

students = add_yourself(students)

oldest_student(students)

```

funcs.py

```python
class Student: # group-FullName-Birthday
    def __init__(self, group, surname, name, patronymic, month, day, year):
        self.group = group
        self.fullname = self.FullName(surname, name, patronymic)
        self.birthday = self.Birthday(month, day, year)

    class FullName:  # surname-name-patronymic
        def __init__(self, surname, name, patronymic):
            self.surname = surname
            self.name = name
            self.patronymic = patronymic

    class Birthday:  # mm-dd-yyyy
        def __init__(self, month, day, year):
            self.month = month
            self.day = day
            self.year = year

    def show(self):
        print("%s %s %s %s %s.%s.%s" % (self.group, self.fullname.surname, self.fullname.name, self.fullname.patronymic,
              self.birthday.month, self.birthday.day, self.birthday.year))

def create_list(s_list):
    # pre-list
    s1 = Student("AD-01", "Melnyk", "Andriy", "Olegovych", "12", "21", "2004")
    s_list.append(s1)
    s2 = Student("AD-01", "Kovalenko", "Oleksiy", "Oleksandrovych", "10", "28", "2002")
    s_list.append(s2)
    s3 = Student("AD-01", "Kravchenko", "Mark", "Evgenovych", "01", "03", "2003")
    s_list.append(s3)

    s4 = Student("IO-05h", "Tkachuk", "Evgen", "Bohdanovych", "11", "17", "2003")
    s_list.append(s4)
    s5 = Student("IO-05h", "Moroz", "Egor", "Markovych", "11", "05", "2003")
    s_list.append(s5)

    s6 = Student("IP-14", "Gaiduchek", "Maxim", "Andriyovych", "05", "28", "2004")
    s_list.append(s6)
    s7 = Student("IP-14", "Kotkov", "Timur", "Maximovych", "07", "07", "2004")
    s_list.append(s7)

    print("List of students:")
    for i in range(len(s_list)):
        s_list[i].show()

def add_yourself(s_list):
    print("\nAdd yourself to list")
    user = Student(input("Group: "), input("Surname: "), input("Name: "), input("Patronymic: "),
                   input("Birth month (mm): "), input("Birth day (dd): "), input("Birth year (yyyy): "))
```

```
    s_list.append(user)

    print("\nNew list of students:")
    for i in range(len(s_list)):
        s_list[i].show()

    return s_list

def oldest_student(s_list):
    groups_list = []
    for i in range(len(s_list)):
        if s_list[i].group not in groups_list:
            groups_list.append(s_list[i].group)

    print("\nList of groups:")
    for i in range(len(groups_list)):
        print("%d. %s" % (i + 1, groups_list[i]))

    group_name = groups_list[int(input("\nChoose group number: ")) - 1] # chosen group

    group_students = [] # students from chosen group
    for i in range(len(s_list)):
        if s_list[i].group == group_name:
            group_students.append(s_list[i])

    # looking for the oldest student
    oldest_year = "2023"
    for i in range(len(group_students)):
        if int(group_students[i].birthday.year) < int(oldest_year):
            oldest_year = group_students[i].birthday.year

    oldest_month = "13"
    for i in range(len(group_students)):
        if int(group_students[i].birthday.year) == int(oldest_year) and int(group_students[i].birthday.month) < int(oldest_month):
            oldest_month = group_students[i].birthday.month

    oldest_day = "32"
    for i in range(len(group_students)):
        if int(group_students[i].birthday.year) == int(oldest_year) and int(group_students[i].birthday.month) == int(oldest_month) and int(group_students[i].birthday.day) < int(oldest_day):
            oldest_day = group_students[i].birthday.day

    oldest = group_students[0]
    for i in range(len(group_students)):
        if group_students[i].birthday.year == oldest_year and group_students[i].birthday.month == oldest_month and group_students[i].birthday.day == oldest_day:
            oldest = group_students[i] # the oldest student in chosen group

    print("\nOldest student in group '%s': " % group_name)
    oldest.show()
```

Результат

```
List of students:
AD-01 Melnyk Andriy Olegovych 12.21.2004
AD-01 Kovalenko Oleksiy Oleksandrovych 10.28.2002
AD-01 Kravchenko Mark Evgenovych 01.03.2003
IO-05h Tkachuk Evgen Bohdanovych 11.17.2003
IO-05h Moroz Egor Markovych 11.05.2003
IP-14 Gaiduchek Maxim Andriyovych 05.28.2004
IP-14 Kotkov Timur Maximovych 07.07.2004

Add yourself to list
Group: IP-14
Surname: Nikulin
Name: Pavlo
Patronymic: Yuriyovych
Birth month (mm): 07
Birth day (dd): 13
Birth year (yyyy): 2004

New list of students:
AD-01 Melnyk Andriy Olegovych 12.21.2004
AD-01 Kovalenko Oleksiy Oleksandrovych 10.28.2002
AD-01 Kravchenko Mark Evgenovych 01.03.2003
IO-05h Tkachuk Evgen Bohdanovych 11.17.2003
IO-05h Moroz Egor Markovych 11.05.2003
IP-14 Gaiduchek Maxim Andriyovych 05.28.2004
IP-14 Kotkov Timur Maximovych 07.07.2004
IP-14 Nikulin Pavlo Yuriyovych 07.13.2004

List of groups:
1. AD-01
2. IO-05h
3. IP-14

Choose group number: 3

Oldest student in group 'IP-14':
IP-14 Gaiduchek Maxim Andriyovych 05.28.2004
```

Висновок

Під час виконання лабораторної роботи було досліджено механізми створення і використання класів та об’єктів. У роботі використано клас *Student* та його підкласи *FullName* і *Birthday* зі своїми атрибутами. Було створено список студентів, розбитих по групам, та можливість для користувача додати себе у список. Вибір групи, для пошуку у ній найстаршого студента, здійснюється за номером у списку груп. Роботу виконано на мові програмування *Python*, програма працює коректно.