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 №2
«Класи та об’єкти»
з дисципліни «Основи програмування 2.
Методології програмування»

Варіант 12

Виконала студентка групи ІП-15, Коваленко Марія Олександрівна
Перевірила Вечерковська Анастасія Сергіївна

Київ 2022

Лабораторна робота 2

Тема: «Класи та об’єкти»
Мета: Вивчити особливості створення і використання класів та об’єктів

Варіант 12
Завдання:

12. Розробити клас ”Абонент”, який характеризується ПІБ абонента, його адресою і номером телефону (у форматі XXX-XXX-XX-XX). Створити масив об’єктів даного класу. Визначити абонента, у якого найбільша сума цифр телефону.

1 Виконання мовою Python
1.1 Код програми
main.py

```python
import functions as f
persons = f.list_of_abonents() #creating a list of persons
persons_with_max_phone = f.max_phone(persons) #finding persons with maximum sum of phone digits
f.print_result(persons, persons_with_max_phone)
```

functions.py

```python
from abonent import Abonent

#creating an object
def object_create():
    print("Please, input name:")
    name = input()
    print("Please, input surname:")
    surname = input()
    print("Please, input patronymic:")
    patronymic = input()
    print("Pleas, input address:")
    address = input()
    print("Pleas, input phone in format XXX-XXX-XX-XX:")
    phone = input()
    while not phone_correctness(phone):
        print("please input normal phone:")
        phone = input()
    object = Abonent(name, surname, patronymic, address, phone)
    return object


#checking if input of phone number is valid
def phone_correctness(phone):
    length = len(phone)
    if length != 13:
        return 0
    counter = 0
    for i in range(length):
        if ((i == 3) or (i == 7) or (i == 10)) and (phone[i] != "-"):
            counter += 1
        if ((i != 3) and (i != 7) and (i != 10)) and (not phone[i].isdigit()):
            counter += 1
    if counter != 0:
        return 0
    else:
        return 1

```

```
38
39 #creating a list of persons
40 def list_of_abonents():
41     print("Hello, how many abonents do you wanna create?")
42     num = int(input())
43     abonent_lst = []
44     for i in range (num):
45         abonent_lst.append(object_create())
46     return abonent_lst
47
48
49 #finding a person with max sum of phone digits (or persons if their sums is equal)
50 def max_phone(lst):
51     index = 0
52     max_lst = []
53     for i in range(len(lst)):
54         if lst[i].numbers_count() > lst[index].numbers_count():
55             index = i
56     for item in lst:
57         if item.numbers_count() == lst[index].numbers_count():
58             max_lst.append(item)
59     return max_lst
60
61
62 #printing abonents and persons with maximum sum of phone digits
63 def print_result(list_of_persons, list_of_max_phone):
64     print("\nHere is all abonents you had inputed:")
65     for person in list_of_persons:
66         print(f"\n{person.name} {person.surname} {person.patronymic} \n{person.address}
   \n{person.phone}\n")
67     print("\nPerson(s) with the biggest sum of phone numbers:")
68     for person in list_of_max_phone:
69         print(f"\n{person.name} {person.surname} {person.patronymic} \n{person.address}
   \n{person.phone}\nsum of phone "
70               f"digits: {person.numbers_count()}")
71
```

abonent.py

```
1 class Abonent:
2     #class constructor
3     def __init__(self, name, surname, patronymic, address, phone):
4         self.__name = name
5         self.__surname = surname
6         self.__patronymic = patronymic
7         self.__address = address
8         self.__phone = phone
9
10    #getters
11    @property
12    def name(self):
13        return self.__name
14
15    @property
16    def surname(self):
17        return self.__surname
18
19    @property
20    def patronymic(self):
21        return self.__patronymic
22
23    @property
24    def address(self):
25        return self.__address
26
27    @property
28    def phone(self):
29        return self.__phone
30
31    #setters
32    @name.setter
33    def name(self, value):
34        self.__name = value
35
36    @surname.setter
37    def surname(self, value):
38        self.__surname = value
39
40    @patronymic.setter
41    def patronymic(self, value):
42        self.__patronymic = value
```

```
41      def patronymic(self, value):
42          self.__patronymic = value
43
44      @address.setter
45      def address(self, value):
46          self.__address = value
47
48      @phone.setter
49      def phone(self, value):
50          self.__phone = value
51
52      #function that counts sum ogf phone number digits
53      def numbers_count(self):
54          summ = 0
55          for item in self.__phone:
56              if item != "-":
57                  summ += int(item)
58          return summ
```

2.2 Результат роботи програми

```
Hello, how many abonents do you wanna create?
3
Please, input name:
Dmytro
Please, input surname:
Sighov
Please, input patronymic:
Petrovych
Pleas, input address:
Fareighn street, 7
Pleas, input phone in format XXX-XXX-XX-XX:
123-123-12-12
Please, input name:
Anna
Please, input surname:
Shevtsova
Please, input patronymic:
Mykolayivna
Pleas, input address:
Sunny street, 16
Pleas, input phone in format XXX-XXX-XX-XX:
111-111-12-11
Please, input name:
Lisa
Please, input surname:
Violet
Please, input patronymic:
Ighoryvna
Pleas, input address:
Fog Street, 15
Pleas, input phone in format XXX-XXX-XX-XX:
987-112-14-12

Here is all abonents you had inputed:

Dmytro Sighov Petrovych
Fareighn street, 7
123-123-12-12


Anna Shevtsova Mykolayivna
Sunny street, 16
111-111-12-11


Lisa Violet Ighoryvna
Fog Street, 15
987-112-14-12


Person(s) with the biggest sum of phone numbers:

Lisa Violet Ighoryvna
Fog Street, 15
987-112-14-12
```

Висновок: Я вивчила особливості створення і використання класів та об’єктів шляхом написання програми мовою програмування Python. Розглянула особливості створення приватних полів класу у ці мові.