Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №6

за 3 семестр

По дисциплине: «Языки программирования»

Тема: «Классы. Наследование Python»

Выполнил:

Студент 2 курса

Группы ПО-6(1)

Мартынович Д. М.

Проверил:

Хацкевич М. В.

2021

Цель работы: ознакомиться с принципом инкапсуляции, с использованием принципа наследования в Python.

Задание 1:

Вариант 22

Для своего варианта (класс стран) выполнить следующее:

1. Определить пользовательский класс в соответствии с вариантом задания.

2. Определить счетчик.

3. Определить в классе конструкторы с параметрами и без. Конструктор должен выводить сообщение о количестве объектов.

4. Определить в классе внешние компоненты-функции для получения и установки полей данных.

5. Написать демонстрационную программу, в которой объекты пользовательского класса создаются с помощью неявного использования конструкторов без параметров.

6. Показать в программе явное использование конструкторов с параметрами.

Текст программы:

class Country:  
 count = 0  
 name = str()  
 pravitel = str()  
 people\_number = int()  
  
 def \_\_init\_\_(self):  
 Country.count += 1  
 print("Количество: ", Country.count)  
  
 def \_\_init\_\_(self, name="", pravitel="", people\_number=0):  
 Country.count += 1  
 print("Количество: ", Country.count)  
 try:  
 self.name = name  
 self.pravitel = pravitel  
 self.people\_number = people\_number  
 except ValueError as error:  
 print(error)  
  
 def set\_name(self, name):  
 self.name = name  
  
 def set\_pravitel(self, pravitel):  
 self.pravitel = pravitel  
  
 def set\_people\_number(self, people\_number):  
 self.people\_number = people\_number  
  
 def get\_name(self):  
 return self.name  
  
 def get\_pravitel(self):  
 return self.pravitel  
  
 def get\_people\_number(self):  
 return self.people\_number  
  
 def reading(self):  
 try:  
 self.name = input("Введите название страны: ")  
 self.pravitel = input("Введите имя правителя: ")  
 self.people\_number = int(input("Введите кол-во населения: "))  
 except ValueError:  
 print("Ошибка ввода")  
  
 def show(self):  
 print("\nНазвание страны: ", self.name, "\nИмя правителя:", self.pravitel, "\nКол-во населения: ", self.people\_number, "\n")  
  
  
a1 = Country()  
a1.set\_name("Беларусь")  
a1.set\_pravitel("Николай")  
a1.set\_people\_number(123333)  
a1.show()  
  
b2 = Country("Россия", "Путин", 5000000)  
b2.show()  
  
c3 = Country()  
c3.reading()  
c3.show()

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Задание 2.

Вариант 15

Для своего варианта выполнить следующее:

1. Построить модель предметной области в соответствии со своим вариантом (Военнослужащий-Офицер-Человек).

2. Для каждого класса создать конструктор и деструктор, выдающий сообщение о своей работе.

3. Для каждого класса создать внешние функции установки и получения полей данных.

4. Для каждого класса разработать функции, позволяющие представить на экране значения полей данных.

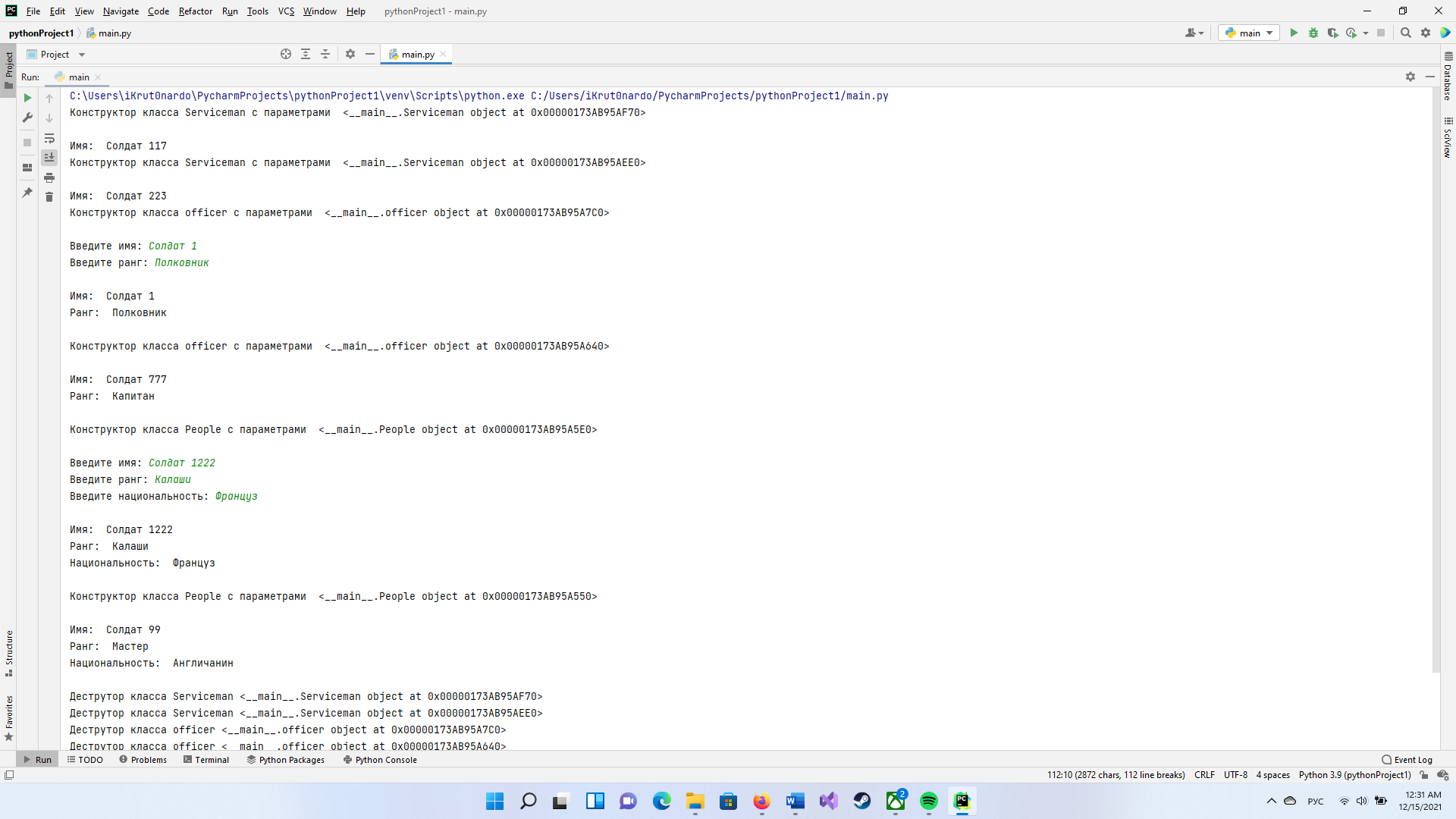
5. Для каждого класса разработать функции, позволяющие вводить с консоли значения полей данных.

6. Написать демонстрационную программу, иллюстрирующую поочередный вызов конструкторов и деструкторов базового и производного классов.

7. Построить диаграмму классов.

Текст программы:

class Serviceman:  
 name = str()  
 def \_\_init\_\_(self):  
 print ("Конструктор класса Serviceman без параметров ", self)  
 def \_\_init\_\_(self, name = ""):  
 print ("Конструктор класса Serviceman с параметрами ", self)  
 try:  
 self.name = name  
 except ValueError as error:  
 print(error)  
  
 def \_\_del\_\_(self):  
 print ("Деструтор класса Serviceman", self)  
  
 def set\_name(self, name):  
 self.name = name  
 def get\_name(self):  
 return self.name  
  
 def reading (self):  
 try:  
 self.name = input("\nВведите имя: ")  
 except ValueError:  
 print("Ошибка ввода")  
  
 def show(self):  
 print("\nИмя: ", self.name)  
  
class officer(Serviceman):  
 Rank = str()  
 def \_\_init\_\_(self):  
 print ("Конструктор класса officer без параметров ", self)  
 def \_\_init\_\_(self, name = "", Rank = ""):  
 print ("Конструктор класса officer с параметрами ", self)  
 try:  
 self.name = name  
 self.Rank = Rank  
 except ValueError as error:  
 print(error)  
  
 def \_\_del\_\_(self):  
 print ("Деструтор класса officer", self)  
  
 def set\_Rank(self, Rank = ""):  
 self.Rank = Rank  
 def get\_Rank(self):  
 return self.Rank  
  
 def reading (self):  
 try:  
 self.name = input("\nВведите имя: ")  
 self.Rank = input("Введите ранг: ")  
 except ValueError:  
 print("Ошибка ввода")  
  
 def show(self):  
 print("\nИмя: ", self.name)  
 print("Ранг: ", self.Rank, "\n")  
  
class People(officer):  
 nation = str()  
 def \_\_init\_\_(self):  
 print ("Конструктор класса People без параметров ", self)  
 def \_\_init\_\_(self, name = "", Rank = "", nation = ""):  
 print ("Конструктор класса People с параметрами ", self)  
 try:  
 self.name = name  
 self.Rank = Rank  
 self.nation = nation  
 except ValueError as error:  
 print(error)  
  
 def \_\_del\_\_(self):  
 print ("Деструтор класса People", self)  
  
 def set\_nation(self, nation):  
 self.nation = nation  
 def get\_nation(self):  
 return self.nation  
  
 def reading (self):  
 try:  
 self.name = input("\nВведите имя: ")  
 self.Rank = input("Введите ранг: ")  
 self.nation = input("Введите национальность: ")  
 except ValueError:  
 print("Ошибка ввода")  
  
 def show(self):  
 print("\nИмя: ", self.name)  
 print("Ранг: ", self.Rank)  
 print("Национальность: ", self.nation, "\n")  
  
a1 = Serviceman()  
a1.set\_name("Солдат 117")  
a1.show()  
  
b2 = Serviceman("Солдат 223")  
b2.show()  
  
s1 = officer()  
s1.reading()  
s1.show()  
  
c2 = officer("Солдат 777", "Капитан")  
c2.show()  
  
g1 = People()  
g1.reading()  
g1.show()  
  
d2 = People("Солдат 99", "Мастер", "Англичанин")  
d2.show()



Вывод: ознакомился с основами файловой системы языка Python, с использованием принципа наследования в Python.