Министерство образования Республики Беларусь

УО «Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №6

По дисциплине: “Алгоритмы и структуры данных”

Тема: «Классы. Инкапсуляция.Наследование»Вариант №12

**Выполнил**:

студент 2 курса группы ПО-7

**Малинин Е.А.**

**Проверила:**

Дряпко А.В.

Брест 2021

**Постановка задачи 1:**

Для своего варианта 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.

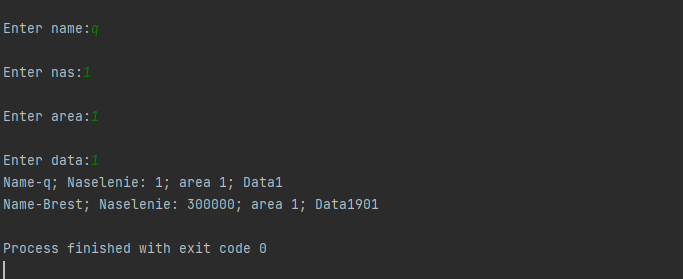
6. Показать в программе явное использование конструкторов с параметрами.

**Вариант задания:**

Город

class gorod:  
 cout = 0  
 name = str()  
 nas = int()  
 area = int()  
 data = int()  
  
  
  
  
 def \_\_init\_\_(self):  
 gorod.count += 1  
 print(gorod.count)  
  
 def \_\_init\_\_(self, name="", nas=0, area=0, data="noy"):  
 self.name = name  
 self.nas = nas  
 self.area = area  
 self.data = data  
  
 def set\_name(self,nm):  
 self.name = nm  
  
 def set\_nas(self,ns):  
 self.nas = ns  
  
 def set\_area(self,ar):  
 self.area = ar  
  
 def set\_data(self,dt):  
 self.data = dt  
  
 def get\_name(self):  
 return self.name  
  
 def get\_nas(self):  
 return self.nas  
  
 def get\_area(self):  
 return self.area  
  
 def get\_data(self):  
 return self.data  
  
 def read(self):  
 self.name = str(input("\nEnter name:"))  
 self.nas = str(input("\nEnter nas:"))  
 self.area = str(input("\nEnter area:"))  
 self.data = str(input("\nEnter data:"))  
  
 def show(self):  
 print("Name-" + self.name + "; Naselenie: " + str(self.nas) + "; area " + str(self.area) + "; Data-" + str(self.data))  
  
one = gorod()  
two = gorod("Brest",300000,1,1901)  
one.read()  
one.show()  
two.show()

**Результат программы:**

****

**Постановка задачи 2:**

Для своего варианта выполнить следующее:

1. Построить модель предметной области в соответствии со своим вариантом (см. ниже)

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

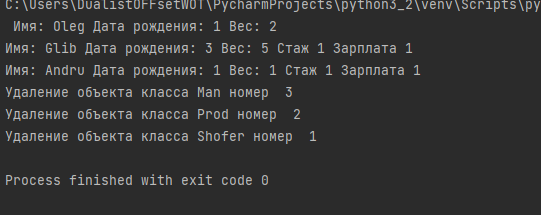
6. Написать демонстрационную программу, иллюстрирующую поочередный вызов конструкторов и деструкторов базового и производного классов

7. Построить диаграмму классов

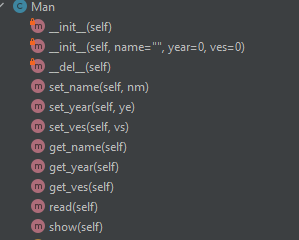
**Вариант задания:**

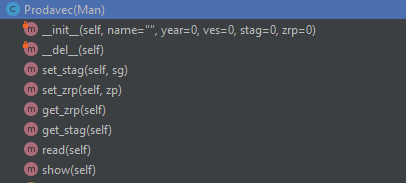
**Человек-шофёр-продавец**

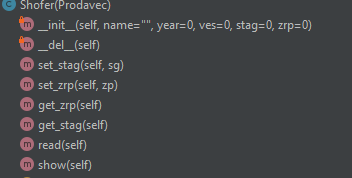
class Man:  
 count = 0  
 name = str()  
 year = int()  
 ves = int()  
  
 def \_\_init\_\_(self):  
 Man.count += 1  
  
 def \_\_init\_\_(self, name="", year=0, ves=0):  
 Man.count += 1  
 self.name = name  
 self.year = year  
 self.ves = ves  
  
 def \_\_del\_\_(self):  
 print("Удаление объекта класса Man номер ", self.count)  
 Man.count -= 1  
  
 def set\_name(self, nm):  
 self.name = nm  
  
 def set\_year(self, ye):  
 self.year = ye  
  
 def set\_ves(self, vs):  
 self.ves = vs  
  
 def get\_name(self):  
 return self.name  
  
 def get\_year(self):  
 return self.year  
  
 def get\_ves(self):  
 return self.ves  
  
 def read(self):  
 self.name = str(input("Введите имя: "))  
 self.year = int(input("Введите датy рождения:"))  
 self.ves = int(input("Введите вес:"))  
  
 def show(self):  
 print(" Имя: " + self.name + " Дата рождения: " + str(self.year) + " Вес: " + str(self.ves))  
  
  
class Prodavec(Man):  
 stag = int()  
 zrp = int()  
  
 def \_\_init\_\_(self,name="", year=0, ves=0,stag=0,zrp=0):  
 Man.count += 1  
 self.name = name  
 self.year = year  
 self.ves = ves  
 self.stag = stag  
 self.zrp = zrp  
  
 def \_\_del\_\_(self):  
 print("Удаление объекта класса Prod номер ", self.count)  
 Man.count -= 1  
  
 def set\_stag(self, sg):  
 self.stag = sg  
  
 def set\_zrp(self, zp):  
 self.zrp = zp  
  
 def get\_zrp(self):  
 return self.zrp  
  
 def get\_stag(self):  
 return self.stag  
  
 def read(self):  
 self.name = str(input("Введите имя: "))  
 self.year = int(input("Введите датy рождения:"))  
 self.ves = str(input("Введите вес:"))  
 self.stag = int(input("Введите стаж:"))  
 self.zrp = int(input("Ввведите зарплату:"))  
  
 def show(self):  
 print("Имя: " + self.name + " Дата рождения: " + str(self.year) + " Вес: " + str(self.ves) + " Стаж " + str(self.stag) + " Зарплата " + str(self.zrp))  
  
class Shofer(Prodavec):  
  
  
 def \_\_init\_\_(self, name="", year=0, ves=0, stag=0, zrp=0):  
 Man.count += 1  
 self.name = name  
 self.year = year  
 self.ves = ves  
 self.stag = stag  
 self.zrp = zrp  
  
 def \_\_del\_\_(self):  
 print("Удаление объекта класса Shofer номер ", self.count)  
 Man.count -= 1  
  
 def set\_stag(self, sg):  
 self.stag = sg  
  
 def set\_zrp(self, zp):  
 self.zrp = zp  
  
 def get\_zrp(self):  
 return self.zrp  
  
 def get\_stag(self):  
 return self.stag  
  
 def read(self):  
 self.name = str(input("Введите имя: "))  
 self.year = int(input("Введите датy рождения:"))  
 self.ves = str(input("Введите вес:"))  
 self.stag = int(input("Введите стаж:"))  
 self.zrp = int(input("Ввведите зарплату:"))  
  
 def show(self):  
 print("Имя: " + self.name + " Дата рождения: " + str(self.year) + " Вес: " + str(self.ves) + " Стаж " + str(  
 self.stag) + " Зарплата " + str(self.zrp))  
  
man = Man("Oleg",1,2)  
man.show()  
Prodavec = Prodavec("Glib",3,5,1,1)  
Prodavec.show()  
Shofer=Shofer("Andru",1,1,1,1)  
Shofer.show()

****

**Диаграмма**

****

****

****

**Вывод:**

Задания выполнены согласно варианту.