Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский

политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчет по практическая работе № 4

тема «Поиск в ширину»

по дисциплине «Учебно-исследовательская работа»

Выполнил: студент группы ИСТ-22-1б Носов Д.О.

Проверил: ст. пр. каф. ВММБ Сахабутдинова Л.Р.

Пермь, 2023

**Содержание**

[Задание 1 3](#_Toc153141010)

[1.1. Постановка задачи 3](#_Toc153141011)

[1.2. Код программы 3](#_Toc153141012)

[1.3. Работа программы с примерами выполнения кода 4](#_Toc153141013)

# Задание 1

## 1.1. Постановка задачи

Составить хэш-таблицу по графу типа «Друзья-друзей – мои друзья», обойти ошибку перезаписи друзей с одинаковыми именами и найти продавца.

## 1.2. Код программы

class Person:  
 def \_\_init\_\_(self, Name, Job, Friend, Node):  
 self.Name = Name  
 self.Job = Job  
 self.Friend = Friend  
 self.Node = Node  
  
 def getName(self):  
 if self.Friend == 'Me':  
 return 'Me'  
 return self.Name  
  
 def getJob(self):  
 return self.Job  
  
 def getNode(self):  
 return self.Node  
  
 def getFriend(self):  
 return self.Friend  
  
  
jobs = []  
personsMas = []  
  
  
def appendPerson(Person):  
 persons = []  
 if (Person.getJob() in jobs):  
 persons = personsMas[jobs.index(Person.getJob())]  
 persons.append(Person)  
 personsMas[jobs.index(Person.getJob())] = persons  
 else:  
 jobs.append(Person.getJob())  
 persons.append(Person)  
 personsMas.append(persons)  
  
Me = Person('Me', 'Охотник', '', 0)  
PersonOne = Person('Alice', 'Программист', Me, 1)  
PersonTwo = Person('Kolya', 'Почтальон', PersonOne, 2)  
PersonThree = Person('Anna', 'Программист', PersonTwo, 3)  
PersonFour = Person('Alice', 'Продавец', PersonTwo, 3)  
  
appendPerson(PersonOne)  
appendPerson(PersonTwo)  
appendPerson(PersonThree)  
appendPerson(PersonFour)  
  
def Search(nameOfJob):  
 nameOfJob = nameOfJob[0].upper() + nameOfJob[1:]  
 if nameOfJob in jobs:  
 persons = personsMas[jobs.index(nameOfJob)]  
 for i in range(len(persons)):  
 print(persons[i].getName()+" - "+persons[i].getJob())  
 node = persons[i].getNode()  
 helpPerson = persons[i]  
 if node == 1:  
 print("^")  
 print(Me.getName()+" - "+Me.getJob())  
 while node > 1:  
 print("^")  
 node = helpPerson.getNode()  
 helpPerson = helpPerson.getFriend()  
 print(helpPerson.getName()+" - "+helpPerson.getJob())  
 print(f"\n\n")  
 elif nameOfJob == Me.getJob():  
 print(Me.getName() + " - " + Me.getJob())  
 else:  
 print(f"Профессии \"{nameOfJob}\" нет в списке")  
  
Search("Инженер")

## 1.3. Работа программы с примерами выполнения кода

Хэш-таблица реализована таким образом, что за ключи отвечает массив профессий, а за значения – массив из **Person**.

Для инициализации друзей был создан класс **Person** с параметрами **Name** – имя, **Job** – профессия, **Friend** – друг выше по графу, **Node** - уровень графа, где пользователь первый.

Функция **appendPerson** либо добавляет новую профессию в массив **jobs**, а затем и новый массив людей в **personsMas**,либо дополняет массив людей по индексу профессии из **jobs** в массив из **personsMas** с соответствующим индексом.

В функции **Search** реализован поиск по графу друга с заданной профессией. Данная функция выводит полную цепочку друзей от **Me** до нужного друга.

Далее в таблице 1 представлено тестирование работы программы.

Таблица 1

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Python |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |

Неопределенностей при решении задачи выявлено не было.

# 