

Санкт-Петербургский Национальный Исследовательский Университет
Информационных Технологий, Механики и Оптики

Кафедра Систем Управления и Информатики

Лабораторная работа №1

Выполнили:

Панина А.С.,

Озеров А.А.,

Умралиева Н.Р.

Проверил

Мусаев А.А.

Санкт-Петербург,

2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1 ЗАДАНИЕ №1.	4
2 ЗАДАНИЕ №2.	5
3 ЗАДАНИЕ №3	7
ЗАКЛЮЧЕНИЕ	8
Список использованных источников	9

ВВЕДЕНИЕ

Цель лабораторной работы №1 – знакомство с графами. В ходе работы необходимо выполнить три задания:

1. Написать программу для бинарного поиска. Результатом должно быть количество шагов, которое потребуется, чтобы найти загаданное число из массива.
2. Для своей учебной группы составить словарь, который будет описывать характеристики каждого из студентов. Реализовать программу, которая по определенным характеристикам будет угадывать студента.
3. Составить граф для задания №2 и определить, к какому типу относится данный граф.

1 ЗАДАНИЕ №1.

1.1 Описание кода

Алгоритм заключается в том, чтобы смещать левую и правую границы до момента, пока центром получившегося промежутка не станет загаданное число.

Сначала создадим список отсортированных чисел от 1 до 100000 и назовём его "nums". Переменная "task" принимает число от пользователя, которое мы будем искать с помощью бинарного поиска. Созданы также переменные:

—"left" и "right" - являются границами слева и справа соответственно

—"mid" - центр диапазона/число, от которого будем отталкиваться при поиске

—"step" - считает кол-во шагов, необходимых для нахождения загаданного числа

Алгоритм начинается с 'while'. Мы будем искать число до момента, пока "left" меньше "right": добавляем 1 шаг к "step" и вычисляем центр (строка 10). Далее проверяем, является ли центр загаданным числом. Если да, то выходим из цикла и выводим результат. Иначе сравниваем: если центр меньше загаданного числа, то смещаем левую границу правее центра на 1 (т.к. всё, что левее "mid + 1", уже автоматически будет меньше загаданного числа. Однако если центр больше загаданного числа, то смещаем правую границу и приравниваем её к "mid - 1" (по аналогии с левой границей).

Программа продолжает работу, пока не найдёт нужное число или между левой и правой границ не останется чисел.

1.2 Примеры ввода-вывода.

Приведем пример работы алгоритма:

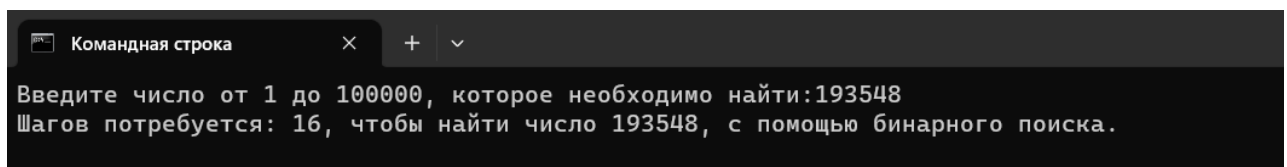


Рисунок 1 - Пример работы бинарного поиска

2 ЗАДАНИЕ №2.

2.1 Описание кода

Изначально мы предложили всем студентам группы K3123 пройти опрос в виде гугл-формы, результаты которого сохранили в файл “opros_k3123.txt”. Далее создали файл “task.py”, в котором и написали код.

Первым шагом мы создали словарь “students”, чтобы сохранить результаты опроса там. Далее, используя конструкцию “with ... as ...”, сохранили вопросы с файла “opros_k3123.txt” в список “questions” и заполнили словарь “students” так, что ключами являются ФИО студентов, а значения ключей – ответы студентов на вопросы.

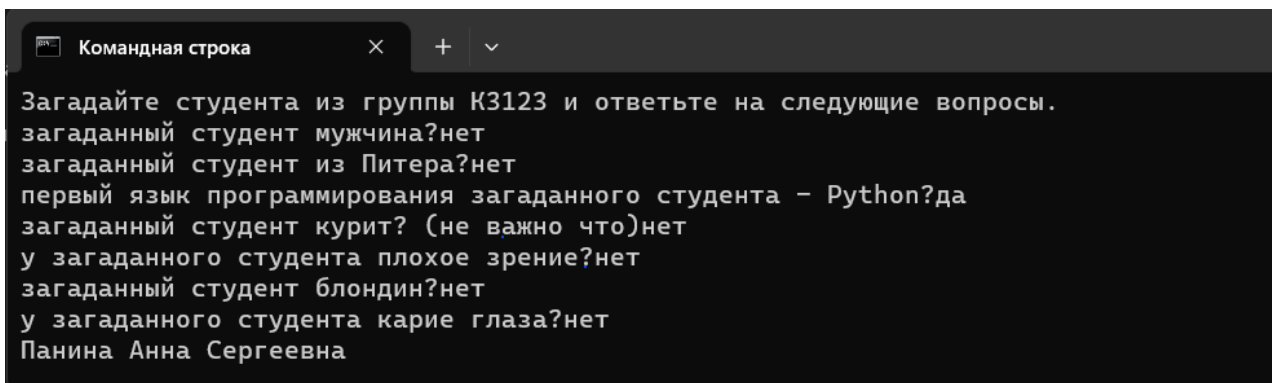
Следующим шагом стало создание пустого списка “user_answer”, для сохранения в него в будущем ответы пользователя. Также был создан список “list_before” с ключами словаря (или же ФИО всех студентов) и пустой список “list_after” для сохранения информации до и после процедуры поиска, реализуемой далее.

Далее мы вводим функцию поиска “search”, в процессе которой программа сравнивает характеристики пользователя с ответами студентов, ответы которых “подошли” на предыдущем шаге (из списка “list_before”), и если характеристика подходит, добавляет их в пустой список “list_after”.

После начинается процедура взаимодействия с пользователем, где пользователь отвечает на вопросы из списка “questions”. Каждый ответ сохраняется в ранее созданный список “user_answer” (если данный ответ существует), после чего запускается функция “search”, которая “фильтрует” студентов и оставляет только студентов, у которых подходящие характеристики. Процесс ввода ответов происходит до того момента, пока не останется один подходящий студент или подходящих студентов не будет вовсе. Если студент в списке один, то программа выводит его имя. Если в списке осталось больше одного студента – продолжает процедуру ввода характеристик от пользователя. Если же подходящих студентов не нашлось, то программа останавливается с сообщением для пользователя, что подходящих под его характеристику студентов нет в группе K3123.

2.2 Примеры ввода-вывода.

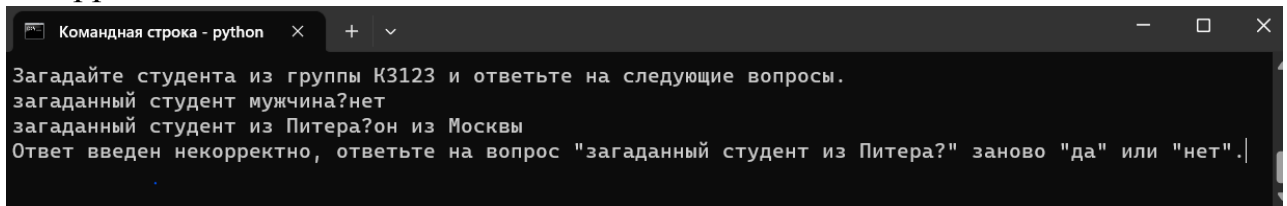
Приведем пример работы алгоритма, когда загаданный студент существует и при вводе характеристик пользователем без ошибок.



```
Командная строка
Загадайте студента из группы K3123 и ответьте на следующие вопросы.
загаданный студент мужчина?нет
загаданный студент из Питера?нет
первый язык программирования загаданного студента - Python?да
загаданный студент курит? (не важно что)нет
у загаданного студента плохое зрение?нет
загаданный студент блондин?нет
у загаданного студента карие глаза?нет
Панина Анна Сергеевна
```

Рисунок 2 – Пример успешной работы алгоритма.

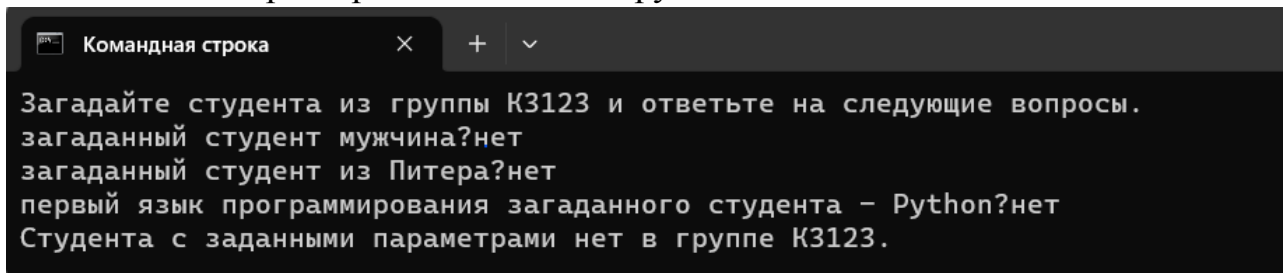
Приведем пример работы алгоритма, когда пользователь ввел некорректный ответ.



```
Командная строка - python  x  +  v
Загадайте студента из группы K3123 и ответьте на следующие вопросы.
загаданный студент мужчина?нет
загаданный студент из Питера?он из Москвы
Ответ введен некорректно, ответьте на вопрос "загаданный студент из Питера?" заново "да" или "нет".|
```

Рисунок 3 – Пример вывода программы при некорректном вводе пользователя.

+Приведем пример работы алгоритма, когда студента с подходящими характеристиками нет в группе.



```
Командная строка  x  +  v
Загадайте студента из группы K3123 и ответьте на следующие вопросы.
загаданный студент мужчина?нет
загаданный студент из Питера?нет
первый язык программирования загаданного студента – Python?нет
Студента с заданными параметрами нет в группе K3123.
```

Рисунок 4 – Пример вывода программы при отсутствии студента с заданными параметрами.

3 ЗАДАНИЕ №3

3.1 Описание графа

Построим граф по тому, какой вопрос задается пользователю и после какого мы сможем получить нужный результат. Воспользуемся сайтом programforyou.ru для этого. Исходя из кода задания 2, мы понимаем, что программа проверяет, не получили ли мы конечное ФИО, после каждого вопроса. Исходя из этого, мы можем построить такой граф:

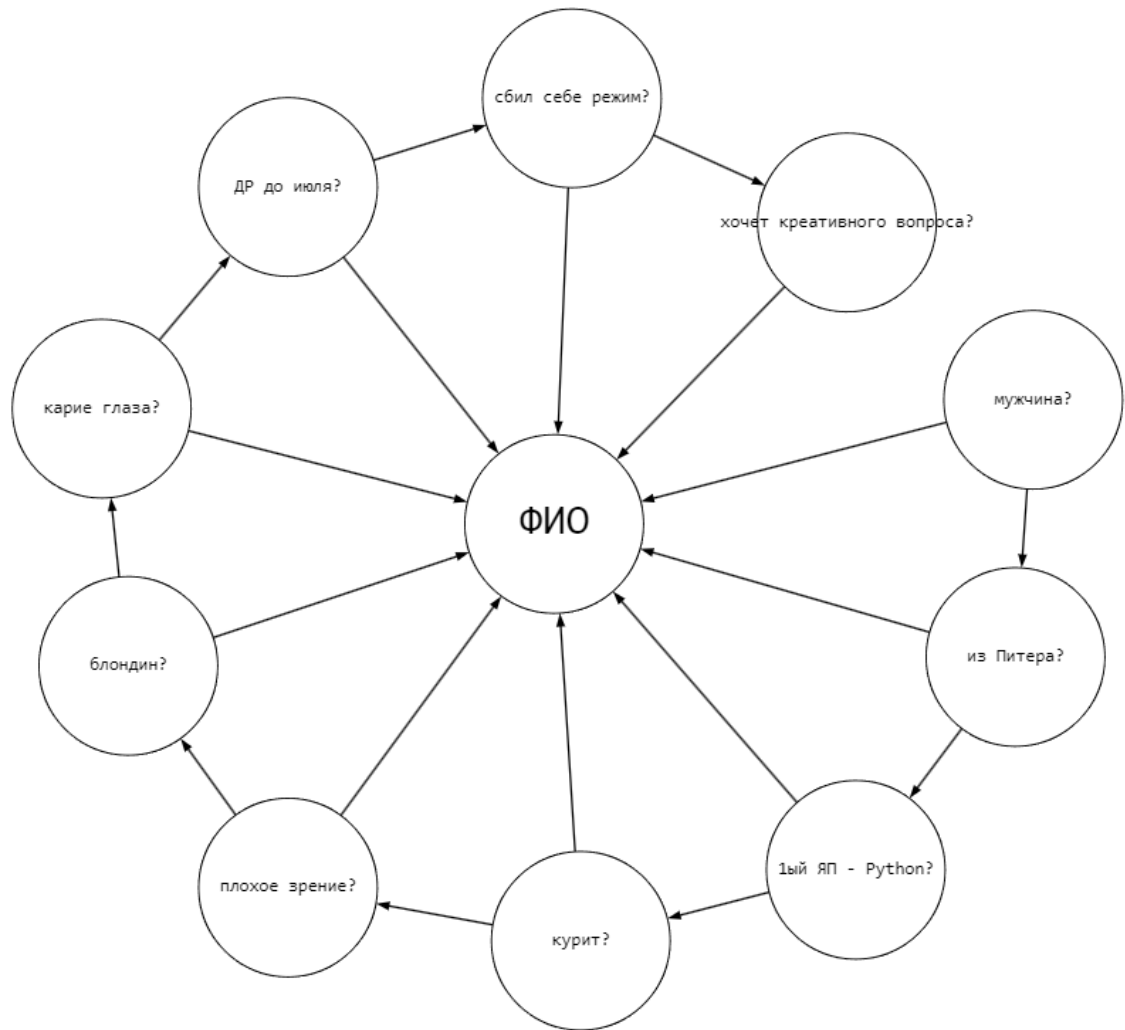


Рисунок 5 – Граф для задания №3

На рисунке мы видим, что получился ориентированный граф.

ЗАКЛЮЧЕНИЕ

В ходе лабораторной работы №1 было выполнено три задания: осуществление бинарного поиска, написание программы, которая по определенным характеристикам угадывает студента и составление графа для данного алгоритма. Благодаря этому мы познакомились с графами. Следовательно, цель работы была достигнута.

Список использованных источников

1 Репозиторий с лабораторной работой на портале GitHub: [Электронный ресурс]. URL: https://github.com/AlexMarticus/AiSD_labs/tree/main/lab_1