МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В.Н. Каразіна

Навчально-науковий інститут комп’ютерних наук та штучного інтелекту

ПРАКТИЧНА РОБОТА №1

на тему: «Розробка програми для обробки великого обсягу даних»

Виконав: студент 2 курсу групи КС22
Спеціальності 122 «Комп’ютерні науки»
Скрипняк Тарас Артемович
Прийняв: к.т.н., доцент каф. ІІПО
Бережний А. А.

Харків – 2024

Мета роботи: Розробити програму для пошуку та обробки інформації про учнів початкової школи, яка зберігається у текстовому файлі students.txt. Програма повинна надавати користувачеві зручний інтерфейс для виконання пошукових запитів за різними критеріями, такими як прізвище учня, прізвище викладача, номер класу чи автобусного маршруту, а також оперативно виводити результати пошуку з вимірюванням часу виконання кожної операції.

Завдання:

використання.

The Task

Завдання складається з двох частин. Перша частина завдання дається вам зараз, друга частина буде надана команді, як тільки команда повідомить про виконання першого завдання. Вам надається список учнів місцевої початкової школи разом із їх класом. Список зберігається у файлі **students.txt** (його можна скачати з **stm.khai.edu**). У кожному рядку файлу зберігається інформація про студента.

Формат рядка:

StLastName, StFirstName, Grade, Classroom, Bus, TLastName, TFirstName

Тут, StLastName і StFirstName ідентифікують студента; TFirstName ідентифікує вчителя учня. Bus - це маршрут шкільного автобуса, яким учень їде, щоб добратися до школи. Bus, Grade і Classroom - цілі числа (якщо значення цього поля 0, то це означає -**дитячий садок**, учень, який самостійно прибуває до школи – значення відмінне від 0), а всі інші поля - це рядки.

Ось зразок рядка з файлу:

DROP, SHERMAN, 0, 104, 51, NIBLER, JERLENE

Не дивно, що рядок слід читати «Sherman Drop, який їде автобусом 51 маршрутом, є вихованцем дитячого садка, Mrs. Jerlene Nibler у класі 104».

Ваша мета - написати програму, яка здійснює пошук студентів у текстовому файлі та видає результати пошуку.

Потрібно здійснити такі пошукові запити:

- Враховуючи прізвище студента, знайдіть клас студента, та клас викладача (якщо є більше одного студента з однаковим прізвищем, знайдіть цю інформацію для всіх студентів);
- Враховуючи прізвище студента, знайдіть автобусний маршрут, яким їде студент (якщо є більше одного студента з однаковим прізвищем, знайдіть цю інформацію для всіх студентів);
- Дан викладач, знайдіть список його учнів;
- Враховуючи автобусний маршрут, знайдіть усіх учнів, які ним користуються;
- Знайдіть усіх учнів на певному рівні класу;

```
const fs = require('fs');
const readline = require('readline');

// Функція для завантаження та парсингу файлу students.txt
function loadStudents() {
  const data = fs.readFileSync('students.txt', 'utf-8');
  return data.split('\n').map(line => {
    const [StLastName, StFirstName, Grade, Classroom, Bus, TLastName, TFirstName] =
line.split(',').map(x => x.trim());
    return { StLastName, StFirstName, Grade: +Grade, Classroom: +Classroom, Bus: +Bus, TLastName,
TFirstName };
  });
}

// Функції для пошуку
// 1. За фамілією
const findStudentByLastName = (students, lastname) => {
  return students.filter(student => student.StLastName.toLowerCase() === lastname.toLowerCase())
```

```
}
// 2. 3
const findBusRouteByStudent = (students, lastname, bus) => {
  return students.filter(student => student.StLastName.toLowerCase() === lastname.toLowerCase() &&
student.Bus == bus);
}

const findStudentsByTeacher = (students, teacherLastName) => {
  return students.filter(student => student.TLastName.toLowerCase() ===
teacherLastName.toLowerCase());
}

const findStudentsByClassroom = (students, classroom) => {
  return students.filter(student => student.Classroom === classroom);
}

const findStudentsByBusRoute = (students, busRoute) => {
  return students.filter(student => student.Bus === busRoute);
}

// Інтерфейс командного рядка
const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});

const students = loadStudents();

function promptUser() {
  rl.question('Enter command. [S/SB/T/C/B/Q]: ', (input) => {
    const [command, arg1, arg2] = input.split(' ');

    let startTime, endTime;
    if (!command) console.log('ERROR! You\'re required to enter a command.')
    else if (command !== 'Q' && !arg1) console.log('ERROR! Each command requires supplemental info.
Here\'s each one:\nS[tudents] <lastname> [B]; T[eacher] <lastname>; C[lassroom] <number>; B[us]
<number>');
    else switch (command.toLowerCase()) {
      case 's':
        startTime = Date.now();
        const results = findStudentByLastName(students, arg1);
        results.forEach(student => console.log(`${student.StFirstName} ${student.StLastName} -
Grade: ${student.Grade}, Classroom: ${student.Classroom}, Teacher: ${student.TFirstName} $
{student.TLastName}, Bus: ${student.Bus}`));
        endTime = Date.now();
        console.log(`Search time: ${endTime - startTime}ms`);
        break;

      case 'sb':
        startTime = Date.now();
        results.forEach(student => console.log(`${student.StFirstName} ${student.StLastName} -
Grade: ${student.Grade}, Classroom: ${student.Classroom}, Teacher: ${student.TFirstName} $
{student.TLastName}, Bus: ${student.Bus}`));
        endTime = Date.now();
        console.log(`Search time: ${endTime - startTime}ms`);
        break;

      case 't':
        startTime = Date.now();
        const studentsByTeacher = findStudentsByTeacher(students, arg1);
        studentsByTeacher.forEach(student => console.log(`${student.StFirstName} $
{student.StLastName}`));
        endTime = Date.now();
        console.log(`Search time: ${endTime - startTime}ms`);
        break;

      case 'c':
        startTime = Date.now();
        const studentsByClassroom = findStudentsByClassroom(students, parseInt(arg1));
        studentsByClassroom.forEach(student => console.log(`${student.StFirstName} $
{student.StLastName}`));
```

```
        endTime = Date.now();
        console.log(`Search time: ${endTime - startTime}ms`);
        break;

      case 'b':
        startTime = Date.now();
        const studentsByBusRoute = findStudentsByBusRoute(students, parseInt(arg1));
        studentsByBusRoute.forEach(student => console.log(`${student.StFirstName} $
{student.StLastName} - Grade: ${student.Grade}, Classroom: ${student.Classroom}`));
        endTime = Date.now();
        console.log(`Search time: ${endTime - startTime}ms`);
        break;

      case 'q':
        rl.close();
        return;

      default:
        console.log('Invalid command.');
    }

    // Повертаємо підказку після виконання команди
    promptUser();
  });
}

// Запуск програми
promptUser();
```

Лістинг на мові node.js

```
INGER COOKUS - Grade: 2, Classroom: 110, Teacher: REUBEN GAMBREL, Bus: 54
TOMAS COOKUS - Grade: 4, Classroom: 109, Teacher: BENITO CHIONCHIO, Bus: 54
ZANDRA COOKUS - Grade: 6, Classroom: 106, Teacher: REUBEN FAFARD, Bus: 51
HYE COOKUS - Grade: 5, Classroom: 107, Teacher: JONATHAN CHIONCHIO, Bus: 56
XUAN COOKUS - Grade: 1, Classroom: 105, Teacher: JED BODZIONY, Bus: 52
INGER COOKUS - Grade: 5, Classroom: 103, Teacher: BENITO HANTZ, Bus: 51
DICK COOKUS - Grade: 2, Classroom: 104, Teacher: BENITO HANTZ, Bus: 54
MANIE COOKUS - Grade: 2, Classroom: 102, Teacher: NANCY GAMBREL, Bus: 53
TAMESHA COOKUS - Grade: 3, Classroom: 104, Teacher: JAE HANTZ, Bus: 53
BILLY COOKUS - Grade: 3, Classroom: 111, Teacher: JED HAMER, Bus: 51
BILLY COOKUS - Grade: 3, Classroom: 111, Teacher: NANCY STEIB, Bus: 55
BILLY COOKUS - Grade: 6, Classroom: 109, Teacher: JAE STEIB, Bus: 51
Search time: 73ms
Enter command. [S/SB/T/C/B/Q]: S
ERROR! Each command requires supplemental info. Here's each one:
S[tudents] <lastname> [B]; T[eacher] <lastname>; C[lassroom] <number>; B[us] <number>
Enter command. [S/SB/T/C/B/Q]: S COOKUS
```

```
XUAN COOKUS - Grade: 3, Classroom: 107, Teacher: ROCIO FAFARD, Bus: 52
RANDOLPH COOKUS - Grade: 5, Classroom: 111, Teacher: PERLA HANTZ, Bus: 52
XUAN COOKUS - Grade: 1, Classroom: 105, Teacher: JED BODZIONY, Bus: 52
XUAN COOKUS - Grade: 3, Classroom: 107, Teacher: ROCIO FAFARD, Bus: 52
RANDOLPH COOKUS - Grade: 5, Classroom: 111, Teacher: PERLA HANTZ, Bus: 52
XUAN COOKUS - Grade: 1, Classroom: 105, Teacher: JED BODZIONY, Bus: 52
XUAN COOKUS - Grade: 3, Classroom: 107, Teacher: ROCIO FAFARD, Bus: 52
RANDOLPH COOKUS - Grade: 5, Classroom: 111, Teacher: PERLA HANTZ, Bus: 52
XUAN COOKUS - Grade: 1, Classroom: 105, Teacher: JED BODZIONY, Bus: 52
XUAN COOKUS - Grade: 3, Classroom: 107, Teacher: ROCIO FAFARD, Bus: 52
RANDOLPH COOKUS - Grade: 5, Classroom: 111, Teacher: PERLA HANTZ, Bus: 52
XUAN COOKUS - Grade: 1, Classroom: 105, Teacher: JED BODZIONY, Bus: 52
XUAN COOKUS - Grade: 3, Classroom: 107, Teacher: ROCIO FAFARD, Bus: 52
RANDOLPH COOKUS - Grade: 5, Classroom: 111, Teacher: PERLA HANTZ, Bus: 52
XUAN COOKUS - Grade: 1, Classroom: 105, Teacher: JED BODZIONY, Bus: 52
Search time: 30ms
Enter command. [S/SB/T/C/B/Q]: SB
ERROR! Each command requires supplemental info. Here's each one:
S[tudents] <lastname>; SB <lastname> <busroute>; T[eacher] <lastname>; C[lassroom] <number>; B[us] <number>
Enter command. [S/SB/T/C/B/Q]: SB COOKUS
No bus route specified!
```

```
DIMPLE BREVO - Grade: 6, Classroom: 110, Teacher: ROCIO ALPERT, Bus: 55
DICK BEX - Grade: 5, Classroom: 108, Teacher: NANCY ALPERT, Bus: 52
INGER CORKER - Grade: 4, Classroom: 102, Teacher: JED ALPERT, Bus: 56
CARTER COVINGTON - Grade: 4, Classroom: 107, Teacher: PERLA ALPERT, Bus: 54
MANIE BOYTER - Grade: 4, Classroom: 102, Teacher: GAVIN ALPERT, Bus: 56
RANDOLPH BREVO - Grade: 6, Classroom: 112, Teacher: JED ALPERT, Bus: 55
RANDOLPH COMO - Grade: 1, Classroom: 112, Teacher: LUZ ALPERT, Bus: 51
JANNETTE COVINGTON - Grade: 5, Classroom: 102, Teacher: GAVIN ALPERT, Bus: 51
FLOY COMO - Grade: 5, Classroom: 104, Teacher: ROCIO ALPERT, Bus: 56
TAMESHA CORONADO - Grade: 5, Classroom: 111, Teacher: PERLA ALPERT, Bus: 54
WAN COVINGTON - Grade: 3, Classroom: 109, Teacher: NANCY ALPERT, Bus: 53
TOMAS CREMEANS - Grade: 6, Classroom: 103, Teacher: LUZ ALPERT, Bus: 53
TAMESHA BEAN - Grade: 6, Classroom: 109, Teacher: JED ALPERT, Bus: 54
INGER BEX - Grade: 6, Classroom: 101, Teacher: BENITO ALPERT, Bus: 52
INGER BUSSMANN - Grade: 4, Classroom: 101, Teacher: PERLA ALPERT, Bus: 52
Search time: 166ms
Enter command. [S/SB/T/C/B/Q]: T ALPERT
```

```
INGER BOYTER
WAN BEAN
TAMESHA BUSSMANN
BILLY CREMEANS
FLOY BEX
CARTER BEX
MANIE COVINGTON
TAMESHA COVINGTON
TOMAS CORONADO
WAN COMO
INGER BEX
MANIE CORONADO
ZANDRA COVINGTON
INGER BUSSMANN
Search time: 69ms
Enter command. [S/SB/T/C/B/Q]: C 101
```

```
MANIE COVINGTON - Grade: 6, Classroom: 101
FLOY BERBES - Grade: 3, Classroom: 112
CARTER CORONADO - Grade: 2, Classroom: 106
RANDOLPH BEX - Grade: 2, Classroom: 105
FLOY BEAN - Grade: 1, Classroom: 107
FLOY BEX - Grade: 6, Classroom: 102
BILLY CORKER - Grade: 1, Classroom: 103
TAMESHA COVINGTON - Grade: 6, Classroom: 101
WAN BEX - Grade: 6, Classroom: 103
INGER CLECKLER - Grade: 1, Classroom: 103
CARTER BUSSMANN - Grade: 4, Classroom: 106
RANDOLPH CIGANEK - Grade: 2, Classroom: 102
INGER BEX - Grade: 6, Classroom: 101
INGER BUSSMANN - Grade: 4, Classroom: 101
DIMPLE CORKER - Grade: 5, Classroom: 103
Search time: 205ms
Enter command. [S/SB/T/C/B/Q]: B 52
```

Пояснення: Програма виконує пошук даних про студентів, вчителів, класи та автобуси, які знаходяться в текстовому файлі students.txt. Користувач може вводити команди в консоль для пошуку інформації. Програма продовжує працювати в інтерактивному режимі, доки користувач не введе команду виходу.

Підсумок:

У ході проекту було розроблено програму для пошуку інформації про учнів місцевої початкової школи на основі даних із файлу `students.txt`. Програма забезпечує швидкий і ефективний пошук за такими критеріями, як:

- Прізвище учня;
- Номер автобусного маршруту;
- Прізвище викладача;
- Номер класу.

Інтерфейс програми реалізовано у вигляді командного рядка, що дозволяє користувачам легко виконувати пошукові запити та отримувати результати в зручному форматі. Програма також вимірює час виконання кожного пошуку.

Висновки:

1. **Функціональність**: Програма успішно реалізує всі основні функції пошуку, визначені в завданні. Вона надає користувачам можливість отримувати детальну інформацію про учнів, викладачів і маршрути автобусів.
2. **Продуктивність**: Завдяки ефективним методам обробки даних з текстового файлу, програма виконує пошуки швидко і без затримок. Вимірювання часу кожної операції дозволяє контролювати продуктивність системи.
3. **Зручність використання**: Інтерфейс командного рядка є інтуїтивним і простим для використання. Користувачі можуть швидко виконувати пошукові запити без потреби у складних інструкціях.
4. **Гнучкість**: Програма побудована таким чином, що її можна легко розширювати новими функціями або змінювати відповідно до нових вимог, що робить її гнучкою для майбутнього розвитку.

Посилання на репозиторій: *https://github.com/bobsytoch/homework-bd/tree/main/pr1*