МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В.Н. Каразіна

Навчально-науковий інститут комп’ютерних наук та штучного інтелекту

ПРАКТИЧНА РОБОТА №2

на тему: «Використання нормалізації і вдосконалення 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;

```
import fs from 'fs';
import readline from 'readline';
const rl = readline.createInterface({
       input: process.stdin,
       output: process.stdout
});

const students = fs.readFileSync('list.txt', 'utf-8').split('\n').map(line => {
       const [last, first, grade, classroom, bus] = line.split(',').map(x => x.trim());
       return { last, first, grade: +grade, classroom: +classroom, bus: +bus };
});

const teachers = fs.readFileSync('teachers.txt', 'utf-8').split('\n').map(line => {
       const [last, first, classroom] = line.split(',').map(x => x.trim());
       return { last, first, classroom: +classroom };
});

const findStudentByLastName = (lastname: string) =>
```

```
	students.filter(student => student.last.toLowerCase() === lastname.toLowerCase())

const findBusRouteByStudent = (lastname: string, bus: number) =>
	students.filter(student => student.last.toLowerCase() === lastname.toLowerCase() &&
student.bus == bus);

const findStudentsByTeacher = (lastname: string) =>
	findTeachersByLastName(lastname).map(v => findStudentsByClassroom(v.classroom).map(s =>
({ ...s, tfirst: v.first, tlast: v.last }))).flat(1);

const findStudentsByClassroom = (classroom: number) =>
	students.filter(student => student.classroom === classroom);

const findStudentsByBusRoute = (bus: number) =>
	students.filter(student => student.bus === bus);

const findStudentsByGrade = (grade: number) =>
	students.filter(student => student.grade == grade);

const findTeachersByClasroom = (clasroom: number) =>
	teachers.filter(teacher => teacher.classroom == clasroom);

const findTeachersByLastName = (lastname: string) =>
	teachers.filter(teacher => teacher.last.toLowerCase() == lastname.toLowerCase());

const findTeachersByGrade = (grade: number) =>
	students.filter(student => student.grade == grade).map(s =>
findTeachersByClasroom(s.classroom).map(v => ({...v, grade: s.grade, sfirst: s.first, slast:
s.last }))).flat(1);

function promptUser() {
	rl.question('Enter command. [S/T/C/B/G/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> (number); T[eacher] <lastname>;
C[lassroom] <number> (T); B[us] <number>; G[rade] <number> (T)');
		else switch (command.toLowerCase()) {
			case 's':
				startTime = Date.now();
				((!arg2 || isNaN(+arg2))
					? findStudentByLastName(arg1)
					: findBusRouteByStudent(arg1, +arg2))
					.forEach(student => console.log(`${student.first} $
{student.last} - Grade: ${student.grade}, Classroom: ${student.classroom}, Bus: $
{student.bus}`));
				endTime = Date.now();
				console.log(`Search time: ${endTime - startTime}ms`);
				break;
			case 't':
				startTime = Date.now();
				findStudentsByTeacher(arg1)
					.forEach(student => console.log(`Teacher: ${student.tfirst}
${student.tlast}, Student: ${student.first} ${student.last} - Grade: ${student.grade},
Classroom: ${student.classroom}, Bus: ${student.bus}`));
				endTime = Date.now();
				console.log(`Search time: ${endTime - startTime}ms`);
				break;
			case 'c':
				startTime = Date.now();
				((arg2 == 'T')
					? findTeachersByClasroom(parseInt(arg1))
					: findStudentsByClassroom(parseInt(arg1)))
					.forEach(student => console.log(`Class: ${arg1}: $
{student.first} ${student.last}`));
				endTime = Date.now();
				console.log(`Search time: ${endTime - startTime}ms`);
				break;
```

```
                        case 'b':
                                startTime = Date.now();
                                findStudentsByBusRoute(parseInt(arg1))
                                        .forEach(student => console.log(`Bus: ${student.bus}
Student: ${student.first} ${student.last} - Grade: ${student.grade}, Classroom: $
{student.classroom}`));
                                endTime = Date.now();
                                console.log(`Search time: ${endTime - startTime}ms`);
                                break;

                        case 'g':
                                startTime = Date.now();
                                if (arg2 == 'T') findTeachersByGrade(parseInt(arg1))
                                        .forEach(student => console.log(`Grade: ${student.grade}
Teacher: ${student.first} ${student.last} Student: ${student.sfirst} ${student.slast} `))
                                else findStudentsByGrade(parseInt(arg1))
                                        .forEach(student => console.log(`Grade: ${student.grade}
Student: ${student.first} ${student.last}`))
                                endTime = Date.now();
                                console.log(`Search time: ${endTime - startTime}ms`);
                                break;
                        case 'q': return rl.close();

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

                promptUser();
        });
}

promptUser();
```

Лістинг на мові TypeScript

```
bouncytorch@AORUS:~/Repos/homework-bd/pr2$ node .
Enter command. [S/T/C/B/G/Q]: S COOKUS
XUAN COOKUS - Grade: 3, Classroom: 107, Bus: 52
XUAN COOKUS - Grade: 3, Classroom: 104, Bus: 55
MANIE COOKUS - Grade: 5, Classroom: 112, Bus: 53
WAN COOKUS - Grade: 5, Classroom: 109, Bus: 56
RANDOLPH COOKUS - Grade: 5, Classroom: 111, Bus: 52
JANNETTE COOKUS - Grade: 5, Classroom: 105, Bus: 51
INGER COOKUS - Grade: 2, Classroom: 110, Bus: 54
TOMAS COOKUS - Grade: 4, Classroom: 109, Bus: 54
ZANDRA COOKUS - Grade: 6, Classroom: 106, Bus: 51
HYE COOKUS - Grade: 5, Classroom: 107, Bus: 56
XUAN COOKUS - Grade: 1, Classroom: 105, Bus: 52
INGER COOKUS - Grade: 5, Classroom: 103, Bus: 51
DICK COOKUS - Grade: 2, Classroom: 104, Bus: 54
MANIE COOKUS - Grade: 2, Classroom: 102, Bus: 53
TAMESHA COOKUS - Grade: 3, Classroom: 104, Bus: 53
BILLY COOKUS - Grade: 3, Classroom: 111, Bus: 51
BILLY COOKUS - Grade: 3, Classroom: 111, Bus: 55
BILLY COOKUS - Grade: 6, Classroom: 109, Bus: 51
Search time: 4ms
```

```
Enter command. [S/T/C/B/G/Q]: S COOKUS 52
XUAN COOKUS - Grade: 3, Classroom: 107, Bus: 52
RANDOLPH COOKUS - Grade: 5, Classroom: 111, Bus: 52
XUAN COOKUS - Grade: 1, Classroom: 105, Bus: 52
Search time: 0ms
```

```
Teacher: JONATHAN COOL, Student: BILLY BUSSMANN - Grade: 6, Classroom: 102, Bus: 56
Teacher: JONATHAN COOL, Student: JANNETTE COVINGTON - Grade: 5, Classroom: 102, Bus: 51
Teacher: JONATHAN COOL, Student: HYE CLECKLER - Grade: 6, Classroom: 102, Bus: 54
Teacher: JONATHAN COOL, Student: JANNETTE CORONADO - Grade: 6, Classroom: 102, Bus: 56
Teacher: JONATHAN COOL, Student: MANIE BEX - Grade: 5, Classroom: 102, Bus: 55
Teacher: JONATHAN COOL, Student: FLOY BEX - Grade: 6, Classroom: 102, Bus: 52
Teacher: JONATHAN COOL, Student: DICK CORKER - Grade: 6, Classroom: 102, Bus: 56
Teacher: JONATHAN COOL, Student: WAN COMO - Grade: 5, Classroom: 102, Bus: 55
Teacher: JONATHAN COOL, Student: RANDOLPH CIGANEK - Grade: 2, Classroom: 102, Bus: 52
Teacher: JONATHAN COOL, Student: INGER BUSSMANN - Grade: 1, Classroom: 102, Bus: 56
Teacher: JONATHAN COOL, Student: FLOY CLECKLER - Grade: 5, Classroom: 102, Bus: 55
Teacher: JONATHAN COOL, Student: CARTER CREMEANS - Grade: 2, Classroom: 102, Bus: 56
Search time: 46ms
Enter command. [S/T/C/B/G/Q]: T COOL
```

```
Class: 102: HYE CLECKLER
Class: 102: JANNETTE CORONADO
Class: 102: MANIE BEX
Class: 102: FLOY BEX
Class: 102: DICK CORKER
Class: 102: WAN COMO
Class: 102: RANDOLPH CIGANEK
Class: 102: INGER BUSSMANN
Class: 102: FLOY CLECKLER
Class: 102: CARTER CREMEANS
Search time: 2ms
Enter command. [S/T/C/B/G/Q]: C 102
```

```
Grade: 5 Student: FLOY COMO
Grade: 5 Student: TAMESHA CORONADO
Grade: 5 Student: FLOY COMO
Grade: 5 Student: DIMPLE CORONADO
Grade: 5 Student: BILLY COMO
Grade: 5 Student: DIMPLE CIGANEK
Grade: 5 Student: MANIE BEX
Grade: 5 Student: MANIE CREMEANS
Grade: 5 Student: JANNETTE CIGANEK
Grade: 5 Student: JANNETTE CORKER
Grade: 5 Student: WAN COMO
Grade: 5 Student: BILLY BEX
Grade: 5 Student: FLOY CLECKLER
Grade: 5 Student: DIMPLE CORKER
Grade: 5 Student: TAMESHA CORKER
Search time: 2ms
Enter command. [S/T/C/B/G/Q]: G 5
```

```
Grade: 5 Teacher: JONATHAN CHIONCHIO Student: TAMESHA CORKER
Grade: 5 Teacher: BENITO CHIONCHIO Student: TAMESHA CORKER
Grade: 5 Teacher: GAVIN HAMER Student: TAMESHA CORKER
Grade: 5 Teacher: GAVIN ALPERT Student: TAMESHA CORKER
Grade: 5 Teacher: PERLA COOL Student: TAMESHA CORKER
Grade: 5 Teacher: PERLA ALPERT Student: TAMESHA CORKER
Grade: 5 Teacher: JAE HANTZ Student: TAMESHA CORKER
Grade: 5 Teacher: ROCIO CHIONCHIO Student: TAMESHA CORKER
Grade: 5 Teacher: JED CHIONCHIO Student: TAMESHA CORKER
Grade: 5 Teacher: GAVIN CHIONCHIO Student: TAMESHA CORKER
Grade: 5 Teacher: ROCIO FALKER Student: TAMESHA CORKER
Grade: 5 Teacher: BENITO FALKER Student: TAMESHA CORKER
Search time: 33ms
Enter command. [S/T/C/B/G/Q]: G 5 T
```

Пояснення: Програма виконує пошук даних про студентів, вчителів, класи та автобуси, які знаходяться в текстовому файлі lists.txt ta teachers.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
- Оцінка

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/pr2