Министерство науки и высшего образования Российской Федерации
Муромский институт (филиал)
Федерального государственного бюджетного образовательного учреждения
высшего образования
«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»

Факультет ИТР

Кафедра ПИн

ЛАБОРАТОРНАЯ РАБОТА №5

Руководитель

Кульков Я.Ю.
(фамилия, инициалы)

(подпись) (дата)

Студент ПИн - 121
(группа)

Ермилов М.В.
(фамилия, инициалы)

(подпись) (дата)

Муром 2023

Лабораторная работа №5

Тема: Коллекции.
Цель: изучение структуры Java Collections Framework и способов работы с ним.
Задачи:

4. Сравнение скорости работы Напишите метод, который добавляет 1000000 элементов в ArrayList и LinkedList. Напишите еще один метод, который выбирает из заполненного списка элемент наугад 100000 раз. Замерьте время, которое потрачено на это. Сравните результаты и предположите, почему они именно такие.

5. Коллекция элементов Создать класс Student, содержащий следующие характеристики – имя, группа, курс, оценки по предметам (не менее 8 предметов). Создать коллекцию, содержащую объекты класса Student (не менее 20). Инициализацию списка производить из текстового файла. Написать метод, который удаляет студентов со средним баллом =3, студент переводится на следующий курс. Напишите метод List getStudents(List students, int course), который получает список студентов и номер курса. Метод возвращает список тех студентов, которые обучаются на данном курсе. Напишите метод, который выводит в отформатированном виде информацию о студентах из переданного списка void printStudents(List students). Выводить на экран используя следующий порядок сортировки: курс, группа, имя по алфавиту.

6. Коллекция элементов (по варианту) Создайте класс сущности по варианту. Создать коллекцию, содержащую объекты разработанного класса (не менее 20). Инициализацию списка производить из текстового файла. Реализуйте метод в соответствии с заданием по варианту. Напишите метод, который возвращает в отформатированном виде информацию о объекте из переданного списка String printSorted(List entity). Формировать строку используя в сортированном порядке (порядок сортровки выбрать самостоятельно исходя из хранимой сущности).

Вариант	Задание
«Покупатель»	фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); номер кредитной карточки; банковского счета. Вывести данные о покупателях с города Муром

Код программы:

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.List;
import java.util.Random;

public class Main
{
	public static void main(String[] args)
	{
		try
		{
			task6();
		}catch(Exception e) {}
	}



	static void task4()
	{
		List<Integer> list1 = new ArrayList<Integer>();
		List<Integer> list2 = new LinkedList<Integer>();
		Random rand = new Random(1);
		for(int i = 0; i < 1000000; i++)
		{
			int r = rand.nextInt();
			list1.add(r);
			list2.add(r);
		}
		task4(list1);
		task4(list2);
	}
```

```
static void task4(List<Integer> list)
{
	long start = System.currentTimeMillis();
	Random rand = new Random(20);
	for(int i = 0; i < 100000; i++)
	{
		list.get(rand.nextInt(list.size()));
	}
	long end = System.currentTimeMillis();
	double time = (end - start) / 1000d;
	System.out.println(time + "sec.");
}


static void task5() throws IOException
{
	List<Student> students = new ArrayList<Student>();

	String[] arr1 = Files.readString(Paths.get("student.txt")).split("\n");
	for(int i = 0; i < arr1.length; i++) {
		String[] arr2 = arr1[i].split(":");
		students.add(task5(arr2[0], arr2[1]));
	}

	Student.printStudents(students);



	List<Student> students2 = new ArrayList<Student>();
	for(Student student: students)
	{
		if(student.getAverageEvaluations() >= 3)
		{
			student.setCourse(student.getCourse() + 1);
			students2.add(student);
```

```
                }
        }

        System.out.println("\n");
        Student.printStudents(students2);

}
static Student task5(String name, String group)
{
        Random random = new Random(name.hashCode() + group.hashCode());
        Student student = new Student(name, group,  random.nextInt(4) + 1);
        student.getEvaluations().put(Student.Course.mathematics, random.nextInt(5) + 1);
        student.getEvaluations().put(Student.Course.physics, random.nextInt(5) + 1);
        student.getEvaluations().put(Student.Course.physical, random.nextInt(5) + 1);
        student.getEvaluations().put(Student.Course.history, random.nextInt(5) + 1);
        student.getEvaluations().put(Student.Course.biology, random.nextInt(5) + 1);
        student.getEvaluations().put(Student.Course.chemistry, random.nextInt(5) + 1);
        student.getEvaluations().put(Student.Course.programming, random.nextInt(5) + 1);
        student.getEvaluations().put(Student.Course.philosophy, random.nextInt(5) + 1);
        student.getEvaluations().put(Student.Course.cultural, random.nextInt(5) + 1);
        return student;
}



static void task6() throws IOException
{
        List<Item> items = new ArrayList<Item>();

        String[] arr1 = Files.readString(Paths.get("item.txt")).split("\n");
        for(int i = 0; i < arr1.length; i++) {
                items.add(Item.parse(arr1[i]));
        }

        System.out.println(Item.printSorted(items));
```

```
	}

}
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.HashMap;
import java.util.List;
import java.util.Map;


public class Student {
	public Student(String name, String group)
	{
		this.group = group;
		this.name = name;
	}
	public Student(String name, String group, int course)
	{
		this(name, group);
		this.course = course;
	}
	private String name;
	private String group;
	private int course = 1;
	private Map<Course, Integer> evaluations = new HashMap<Course, Integer>();
	public enum Course
	{
		mathematics,
		physics,
		physical,
		history,
		biology,
		chemistry,
		programming,
```

```
        philosophy,
        cultural,
}
public String getName() { return name; }
public void setName(String name) { this.name = name; }
public String getGroup() { return group; }
public void setGroup(String group) { this.group = group; }
public int getCourse() { return course; }
public void setCourse(int course) { this.course = course; }
public Map<Course, Integer> getEvaluations() { return evaluations; }

public double getAverageEvaluations()
{
        double a = 0;
        for(Map.Entry<Course, Integer> b: evaluations.entrySet())
        {
                a += b.getValue();
        }
        a /= evaluations.size();
        return a;
}


static public List<Student> getStudents(List<Student> students, int course)
{
        List<Student> students2 = new ArrayList<Student>();
        for(Student student: students)
        {
                if(student.getCourse() == course)
                {
                        students2.add(student);
                }
        }
        return students2;
}
```

```
    static public void printStudents(List<Student> students)
    {
Comparator<Student> comparator = Comparator.comparing(Student::getCourse);
comparator.thenComparing(Student::getGroup);
comparator.thenComparing(Student::getName);
Collections.sort(students, comparator);


        for(Student student: students)
        {
            System.out.println(student);
        }
    }

    @Override
    public String toString() {

        return getCourse() + " " + getGroup() + " " + getName();
    }
}
```