Plan

2023-11-21

1. Функциональные интерфейсы
2. Лямбда-выражения
3. Method references
4. Практика

Theory

https://raw.githubusercontent.com/ait-tr/cohort34.2/main/basic*programming/lesson*51/resources/51.Lambdas,Functional*Interfaces,Method*references.pptm

Homework

Реализовать ещё три фильтра для студентов - при помощи обычного класса, анонимного класса и лямбда-выражения.

Code

src/homework/task_1/Main.java

```
package homework.task_1;

import java.util.HashMap;
import java.util.Map;

public class Main {

    public static void main(String[] args) {

        Map<String, String> flights = new HashMap<>();

        flights.put("Berlin", "London");
        flights.put("Tokyo", "Seoul");
        flights.put("Mumbai", "Berlin");
        flights.put("Seoul", "Mumbai");
        flights.put("London", "Madrid");
        flights.put("Barcelona", "Tokyo");
        flights.put("Moscow", "Paris");
        flights.put("Paris", "Rome");

        String startCity = getStartCity(flights);
        System.out.println("Стартовый город - " + startCity);
        System.out.println("Маршрут - " + createRoute(flights, startCity));
```

```
    }

    public static String getStartCity(Map<String, String> flights) {
        for (String currentCity : flights.keySet()) {
            if (!flights.containsValue(currentCity)) {
                return currentCity;
            }
        }
        return null;
    }

    public static String createRoute(Map<String, String> flights, String startCity) {
        // Barcelona -> Tokyo, Tokyo -> Seoul,
        StringBuilder builder = new StringBuilder();
        String landingCity = flights.get(startCity);

        while (landingCity != null) {
            builder.append(startCity).append(" -> ").append(landingCity).append(", ");
            startCity = landingCity;
            landingCity = flights.get(startCity);
        }
        builder.setLength(builder.length() - 2);
        return builder.toString();
    }
}
```

src/homework/task_2/Main.java

```
package homework.task_2;

import java.util.*;

public class Main {

    public static void main(String[] args) {

        String[] array =
                {"AAA", "BBB", "CCC", "DDD", "AAA", "AAA", "AAA", "BBB", "CCC", "BBB", "

        int countOfElements = 3;

        System.out.println(getElementByEncounters(array, countOfElements));
    }

    public static Set<String> getElementByEncounters(String[] array, int counter) {

        Map<String, Integer> map = new LinkedHashMap<>();

        for (String currentElement : array) {
```

```
            if (map.containsKey(currentElement)) {
                int currentCounter = map.get(currentElement);
                map.put(currentElement, ++currentCounter);
            } else {
                map.put(currentElement, 1);
            }
        }

        Set<String> result = new HashSet<>();

//        for (Map.Entry<String, Integer> pair : map.entrySet()) {
//            if (pair.getValue() == counter) {
//                return pair.getKey();
//            }
//        }

        for (String element : array) {
            if (map.get(element) == counter) {
                result.add(element);
            }
        }

        return result;
    }
}
```

src/task_01/Degree.java

```
package task_01;

public enum Degree {

    BACHELOR("Бакалавр"),
    MASTER("Магистр");

    private String description;

    Degree(String description) {
        this.description = description;
    }

    public String getDescription() {
        return description;
    }
}
```

src/task_01/Filter.java

```
package task_01;

@FunctionalInterface
public interface Filter {

    boolean test(Student student);

//    int test1(String s);
}
```

src/task_01/FirstStudentFilter.java

```
package task_01;

public class FirstStudentFilter implements Filter {

    @Override
    public boolean test(Student student) {
        return student.getCourse() >= 3 && student.getAverageRate() >= 4.5;
    }
}
```

src/task_01/Main.java

```
package task_01;

import java.util.ArrayList;
import java.util.List;

public class Main {

    public static void main(String[] args) {

        List<Student> students = new ArrayList<>();

        students.add(new Student("Сергей", 25, 3, 4.76, Degree.MASTER));
        students.add(new Student("Дмитрий", 32, 2, 4.23, Degree.BACHELOR));
        students.add(new Student("Надежда", 25, 3, 4.71, Degree.MASTER));
        students.add(new Student("Алексей", 21, 1, 4.12, Degree.BACHELOR));
        students.add(new Student("Александра", 23, 4, 4.94, Degree.BACHELOR));
        students.add(new Student("Макар", 29, 1, 4.51, Degree.MASTER));
        students.add(new Student("Степан", 31, 5, 3.98, Degree.MASTER));

        System.out.println("Список всех студентов:");
        for (Student student : students) {
            System.out.println(student);
        }
        System.out.println();
```

```
            // Фильтруем студентов: курс и балл
            // курс >= 3, балл >= 4.5

            System.out.println("Фильтруем студентов по курсу и баллу:");
            StudentService.printStudents(students, new FirstStudentFilter());
            System.out.println();

            // возраст и степень
            // старше 23 и магистр
            System.out.println("Фильтруем студентов по возрасту и степени:");
            Filter secondFilter = new SecondStudentFilter();
            StudentService.printStudents(students, secondFilter);
            System.out.println();

            // курс и имя
            // имя длиннее 6 символов и нечётный курс

            System.out.println("Фильтруем студентов по имени и курсу:");
            StudentService.printStudents(students, new Filter() {
                @Override
                public boolean test(Student student) {
                    return student.getName().length() > 6 && student.getCourse() % 2 != 0;
                }
            });
            System.out.println();

            // имя начинается с А и возраст больше или равен 23
            System.out.println("Фильтруем студентов по имени и возрасту:");
            StudentService.printStudents(students, x -> x.getName().startsWith("A") && x.get
        }
    }
```

src/task_01/SecondStudentFilter.java

```
package task_01;

public class SecondStudentFilter implements Filter {

    @Override
    public boolean test(Student student) {
        return student.getAge() > 23 && student.getDegree().equals(Degree.MASTER);
    }
}
```

src/task_01/Student.java

```
package task_01;

import java.util.Objects;
```

```
public class Student {

    private String name;
    private int age;
    private int course;
    private double averageRate;
    private Degree degree;

    public Student(String name, int age, int course, double averageRate, Degree degree)
        this.name = name;
        this.age = age;
        this.course = course;
        this.averageRate = averageRate;
        this.degree = degree;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public int getCourse() {
        return course;
    }

    public void setCourse(int course) {
        this.course = course;
    }

    public double getAverageRate() {
        return averageRate;
    }

    public void setAverageRate(double averageRate) {
        this.averageRate = averageRate;
    }

    public Degree getDegree() {
```

```
            return degree;
        }

        public void setDegree(Degree degree) {
            this.degree = degree;
        }

        @Override
        public boolean equals(Object o) {
            if (this == o) return true;
            if (o == null || getClass() != o.getClass()) return false;
            Student student = (Student) o;
            return age == student.age && course == student.course && Double.compare(student.
        }

        @Override
        public int hashCode() {
            return Objects.hash(name, age, course, averageRate, degree);
        }

        @Override
        public String toString() {
            return "Student{" +
                    "name='" + name + '\'' +
                    ", age=" + age +
                    ", course=" + course +
                    ", averageRate=" + averageRate +
                    ", degree=" + degree +
                    '}';
        }
    }
```

src/task_01/StudentService.java

```
    package task_01;

    import java.util.List;

    public class StudentService {

        public static void printStudents(List<Student> students, Filter filter) {
            for (Student student : students) {
                if (filter.test(student)) {
                    System.out.println(student);
                }
            }
        }
    }
```