Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №3

По дисциплине «СПП»

Выполнил
студент 2 курса
группы ПО-5:
Филатов Д.Д.
Проверил:
Крощенко А.А.

Брест, 2020

Вариант 8

Цель: научиться создавать и использовать классы в программах на языке программирования Java.

Задание 1:

Реализовать простой класс.
Множество целых чисел переменной мощности – Предусмотреть возможность пересечения двух множеств, вывода на печать элементов множества, а так же метод, определяющий, принадлежит ли указанное значение множеству. Класс должен содержать методы, позволяющие добавлять и удалять элемент в/из множества. Конструктор должен позволить создавать объекты с начальной инициализацией. Реализацию множества осуществить на базе структуры ArrayList. Реализовать метод equals, выполняющий сравнение объектов данного типа.

Код программы:

Main.java

```
package com.company;
import java.util.ArrayList;

public class Main {
    public static void main(String[] args) {
        IntegerSet integerSet1 = new IntegerSet();
        integerSet1.addItem(1);
        integerSet1.addItem(23);
        integerSet1.addItem(456);
        ArrayList<Integer> integerList = new ArrayList() {{
            add(1);
            add(46);
            add(45);
        }};
        IntegerSet integerSet2 = new IntegerSet(integerList);
        System.out.println("First set: " + integerSet1);
        System.out.println("Second set: " + integerSet2);
        System.out.println("integerSet1 == integerSet2: " +
                (integerSet1.equals(integerSet2)));
        integerSet2.addItem(123);
        integerSet1.deleteItemById(2);
        System.out.println("Second item of integerSet2 = " +
                integerSet2.getItemById(2));
        System.out.println("integerSet2 contains 23: " + integerSet2.contains(23));
        System.out.println("intersections: " + integerSet2.intersections(integerSet1));
    }
}
```

IntegerSet.java

```
package com.company;
import java.util.ArrayList;

public class IntegerSet {
    private ArrayList<Integer> setOfIntegers;

    public IntegerSet() {
        this.setOfIntegers = new ArrayList();
    }

    public IntegerSet(ArrayList<Integer> setOfIntegers) {
        this.setOfIntegers = new ArrayList<>();
        for(Integer  el: setOfIntegers) {
            if (!this.setOfIntegers.contains(el))
```

```
                this.setOfIntegers.add(el);
            }
        }

        public ArrayList<Integer> intersections(IntegerSet set) {
            ArrayList<Integer> list = set.getSetOfIntegers();
            list.retainAll(this.setOfIntegers);
            return list;
        }

        public boolean contains(int item) {
            return this.setOfIntegers.contains(item);
        }

        public int getItemById(int id) {
            --id;
            return (Integer)this.setOfIntegers.get(id);
        }

        public void addItem(int item) {
            this.setOfIntegers.add(item);
        }

        public void deleteItemById(int id) {
            this.setOfIntegers.remove(id);
        }

        public ArrayList<Integer> getSetOfIntegers() {
            return this.setOfIntegers;
        }

        public void setSetOfIntegers(ArrayList<Integer> setOfIntegers) {
            this.setOfIntegers = setOfIntegers;
        }

        public String toString() {
            return "IntegerSet = " + this.setOfIntegers;
        }

        public boolean equals(Object o) {
            if (this == o) {
                return true;
            } else if (o != null && this.getClass() == o.getClass()) {
                IntegerSet that = (IntegerSet)o;
                return this.setOfIntegers.equals(that.setOfIntegers);
            } else {
                return false;
            }
        }
}
```

Результат выполнения:

```
First set: IntegerSet = [1, 23, 456]
Second set: IntegerSet = [1, 46, 45]
integerSet1 == integerSet2: false
Second item of integerSet2 = 46
integerSet2 contains 23: false
intersections: [1]
```

Задание 2:

Разработать автоматизированную систему на основе некоторой структуры данных, манипулирующей объектами пользовательского класса. Реализовать требуемые функции обработки данных

Автоматизированная система обработки информации об авиарейсах

Написать программу для обработки информации об авиарейсах (Airlines): Каждый рейс имеет следующие характеристики:

• Пункт назначения;

• Номер рейса;

• Тип самолета;

• Время вылета;

• Дни недели, по которым совершаются рейсы.

Программа должна обеспечить:

• Генерацию списка рейсов;

• Вывод списка рейсов для заданного пункта назначения;

• Вывод списка рейсов для заданного дня недели;

• Вывод списка рейсов для заданного дня недели, время вылета для которых больше заданного;

• Все рейсы самолетов некоторого типа;

• Группировка рейсов по числу пассажиров (маломестные - 1-100 чел, средместные (100-200), крупные рейсы (200-350));

• Все рейсы самолетов туда-обратно.

Код программы:

Main.java

```
package com.company;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Arrays;

public class Main {
    private static final SimpleDateFormat dateFormat = new SimpleDateFormat("dd.MM.yyyy
HH:mm:ss");
    public static void main(String[] args) throws ParseException {
        Airline airline = getAirline();
        System.out.println("All flights:");
        airline.printListOfFlights();
        System.out.println("\nFlights for selected destination:");
        airline.printListOfFlightsForSelectedDestination("Brest");
        System.out.println("\nFlights for selected day:");
        airline.printListOfFlightsForSelectedDay(DayOfWeek.MONDAY);
        System.out.println("\nFlights for selected day and time:");
        airline.printListOfFlightsForSelectedDayAndTime(DayOfWeek.MONDAY,
                dateFormat.parse("15.10.2021 03:12:10"));
        System.out.println("\nFlights for selected type:");
        airline.printListOfFlightsForSelectedType(TypeOfAircraft.AVERAGE);
```

```
    }
    private static Airline getAirline() throws ParseException {
        Airline airline = new Airline(new ArrayList<>());
        airline.addFlights(new Flight("Brest", 1, TypeOfAircraft.AVERAGE,
                dateFormat.parse("12.09.2020 10:12:10"),
                Arrays.asList(DayOfWeek.MONDAY, DayOfWeek.FRIDAY)));
        airline.addFlights(new Flight("Vitebsk", 2, TypeOfAircraft.SMALL,
                dateFormat.parse("14.10.2020 10:56:10"),
                Arrays.asList(DayOfWeek.THURSDAY, DayOfWeek.FRIDAY)));
        airline.addFlights(new Flight("Grodno", 3, TypeOfAircraft.MAJOR,
                dateFormat.parse("16.03.2020 09:30:10"),
                Arrays.asList(DayOfWeek.SATURDAY, DayOfWeek.SUNDAY)));
        airline.addFlights(new Flight("Mogilev", 4, TypeOfAircraft.AVERAGE,
                dateFormat.parse("19.02.2023 11:42:22"),
                Arrays.asList(DayOfWeek.WEDNESDAY, DayOfWeek.FRIDAY)));
        airline.addFlights(new Flight("Gomel", 5, TypeOfAircraft.MAJOR,
                dateFormat.parse("15.05.2020 04:33:10"),
                Arrays.asList(DayOfWeek.TUESDAY, DayOfWeek.THURSDAY)));
        airline.addFlights(new Flight("Minsk", 6, TypeOfAircraft.MAJOR,
                dateFormat.parse("14.10.2017 08:21:11"),
                Arrays.asList(DayOfWeek.MONDAY, DayOfWeek.TUESDAY)));
        airline.addFlights(new Flight("Brest", 7, TypeOfAircraft.SMALL,
                dateFormat.parse("15.10.2013 05:13:10"),
                Arrays.asList(DayOfWeek.WEDNESDAY, DayOfWeek.SUNDAY)));
        return airline;
    }
}
```

Flight.java

```
package com.company;
import java.util.Date;
import java.util.List;

public class Flight {
    private String destination;
    private int flightNumber;
    private TypeOfAircraft typeOfAircraft;
    private Date departureTime;
    private List<DayOfWeek> daysOfTheWeek;
    public Flight(String destination, int flightNumber, TypeOfAircraft typeOfAircraft,
                  Date departureTime,
                  List<DayOfWeek> daysOfTheWeek) {
        this.destination = destination;
        this.flightNumber = flightNumber;
        this.typeOfAircraft = typeOfAircraft;
        this.departureTime = departureTime;
        this.daysOfTheWeek = daysOfTheWeek;
    }
    public String getDestination() {
        return destination;
    }
    public void setDestination(String destination) {
        this.destination = destination;
    }
    public int getFlightNumber() {
        return flightNumber;
    }
    public void setFlightNumber(int flightNumber) {
        this.flightNumber = flightNumber;
    }
    public TypeOfAircraft getTypeOfAircraft() {
        return typeOfAircraft;
    }
    public void setTypeOfAircraft(TypeOfAircraft typeOfAircraft) {
        this.typeOfAircraft = typeOfAircraft;
    }
```

```
    public Date getDepartureTime() {
        return departureTime;
    }
    public void setDepartureTime(Date departureTime) {
        this.departureTime = departureTime;
    }
    public List<DayOfWeek> getDaysOfTheWeek() {
        return daysOfTheWeek;
    }
    public void setDaysOfTheWeek(List<DayOfWeek> daysOfTheWeek) {
        this.daysOfTheWeek = daysOfTheWeek;
    }
    @Override
    public String toString() {
        return "Flight{" +
                "Destination='" + destination + '\'' +
                ", flightNumber=" + flightNumber +
                ", typeOfAircraft=" + typeOfAircraft +
                ", departureTime=" + departureTime +
                ", daysOfTheWeek=" + daysOfTheWeek +
                '}';
    }
}
```

Airline.java

```
package com.company;

import java.util.Date;
import java.util.List;

public class Airline {
    private List<Flight> flights;
    public Airline(List<Flight> flights) {
        this.flights = flights;
    }
    public List<Flight> getFlights() {
        return flights;
    }
    public void setFlights(List<Flight> flights) {
        this.flights = flights;
    }
    public void addFlights(Flight flight) {
        flights.add(flight);
    }
    public void deleteFlights(int id) {
        flights.remove(--id);
    }
    public void printListOfFlights() {
        flights.forEach(System.out::println);
    }
    public void printListOfFlightsForSelectedDestination(String destination) {
        flights.stream().filter(flight ->

flight.getDestination().equals(destination)).forEach(System.out::println);
    }
    public void printListOfFlightsForSelectedDay(DayOfWeek dayOfWeek) {
        flights.stream().filter(flight ->

flight.getDaysOfTheWeek().contains(dayOfWeek)).forEach(System.out::println);
    }
    public void printListOfFlightsForSelectedDayAndTime(DayOfWeek dayOfWeek, Date date)
    {
        flights.stream().filter(flight ->
                flight.getDaysOfTheWeek().contains(dayOfWeek))
                .filter(flight ->
```

```
flight.getDepartureTime().before(date)).forEach(System.out::println);
    }
    public void printListOfFlightsForSelectedType(TypeOfAircraft typeOfAircraft) {
        flights.stream().filter(flight ->

flight.getTypeOfAircraft().equals(typeOfAircraft)).forEach(System.out::println);
    }
}
```

DayOfWeek.java

```
public enum DayOfWeek {
    MONDAY(), TUESDAY(), WEDNESDAY(), THURSDAY(), FRIDAY(), SATURDAY(), SUNDAY()
}
```

TypeOfAircraft.java

```
public enum TypeOfAircraft {
    SMALL(), AVERAGE(), MAJOR()
}
```

Результат выполнения:

```
All flights:
Flight{Destination='Brest', flightNumber=1, typeOfAircraft=AVERAGE, departureTime=Sat Sep 12 10:12:10 MSK 2020, daysOfTheWeek=[MONDAY, FRIDAY]}
Flight{Destination='Vitebsk', flightNumber=2, typeOfAircraft=SMALL, departureTime=Wed Oct 14 10:56:10 MSK 2020, daysOfTheWeek=[THURSDAY, FRIDAY]}
Flight{Destination='Grodno', flightNumber=3, typeOfAircraft=MAJOR, departureTime=Mon Mar 16 09:30:10 MSK 2020, daysOfTheWeek=[SATURDAY, SUNDAY]}
Flight{Destination='Mogilev', flightNumber=4, typeOfAircraft=AVERAGE, departureTime=Sun Feb 19 11:42:22 MSK 2023, daysOfTheWeek=[WEDNESDAY, FRIDAY]}
Flight{Destination='Gomel', flightNumber=5, typeOfAircraft=MAJOR, departureTime=Fri May 15 04:33:10 MSK 2020, daysOfTheWeek=[TUESDAY, THURSDAY]}
Flight{Destination='Minsk', flightNumber=6, typeOfAircraft=MAJOR, departureTime=Sat Oct 14 08:21:11 MSK 2017, daysOfTheWeek=[MONDAY, TUESDAY]}
Flight{Destination='Brest', flightNumber=7, typeOfAircraft=SMALL, departureTime=Tue Oct 15 05:13:10 MSK 2013, daysOfTheWeek=[WEDNESDAY, SUNDAY]}

Flights for selected destination:
Flight{Destination='Brest', flightNumber=1, typeOfAircraft=AVERAGE, departureTime=Sat Sep 12 10:12:10 MSK 2020, daysOfTheWeek=[MONDAY, FRIDAY]}
Flight{Destination='Brest', flightNumber=7, typeOfAircraft=SMALL, departureTime=Tue Oct 15 05:13:10 MSK 2013, daysOfTheWeek=[WEDNESDAY, SUNDAY]}

Flights for selected day:
Flight{Destination='Brest', flightNumber=1, typeOfAircraft=AVERAGE, departureTime=Sat Sep 12 10:12:10 MSK 2020, daysOfTheWeek=[MONDAY, FRIDAY]}
Flight{Destination='Minsk', flightNumber=6, typeOfAircraft=MAJOR, departureTime=Sat Oct 14 08:21:11 MSK 2017, daysOfTheWeek=[MONDAY, TUESDAY]}

Flights for selected day and time:
Flight{Destination='Brest', flightNumber=1, typeOfAircraft=AVERAGE, departureTime=Sat Sep 12 10:12:10 MSK 2020, daysOfTheWeek=[MONDAY, FRIDAY]}
Flight{Destination='Minsk', flightNumber=6, typeOfAircraft=MAJOR, departureTime=Sat Oct 14 08:21:11 MSK 2017, daysOfTheWeek=[MONDAY, TUESDAY]}

Flights for selected type:
Flight{Destination='Brest', flightNumber=1, typeOfAircraft=AVERAGE, departureTime=Sat Sep 12 10:12:10 MSK 2020, daysOfTheWeek=[MONDAY, FRIDAY]}
Flight{Destination='Mogilev', flightNumber=4, typeOfAircraft=AVERAGE, departureTime=Sun Feb 19 11:42:22 MSK 2023, daysOfTheWeek=[WEDNESDAY, FRIDAY]}

Process finished with exit code 0
```

Вывод: Научилась создавать и использовать классы в программах на языке программирования Java.