HBV401G - Þróun Hugbúnaðar
Verkefni 04 - 2

Steinar Darri - sth319@hi.is
Olgeir Ingi - oia2@hi.is
Háskóli Íslands

Test fixture

Nýrri og betri prófun á Travel portalið.

```
public class SearchManagerTest {

    public SearchManagerTest() {
    }

    @BeforeAll
    public static void setUpClass() {
    }
    @AfterAll
    public static void tearDownClass() {
    }
    @BeforeEach
    public void setUp() {
    }
    @AfterEach
    public void tearDown() {
    }

    /**
     * Test of searchForPackage method, of class SearchManager.
     */
    @Test
    public void testSearchForPackage() {
        Calendar startDate = Calendar.getInstance();
        startDate.set(Calendar.YEAR, 2019);
        startDate.set(Calendar.MONTH, 1);
        startDate.set(Calendar.DATE, 1);

        Calendar endDate = Calendar.getInstance();
        endDate.set(Calendar.YEAR, 2019);
        endDate.set(Calendar.MONTH, 1);
        endDate.set(Calendar.DATE, 10);

        Calendar date = Calendar.getInstance();
        date.set(Calendar.YEAR, 2019);
        date.set(Calendar.MONTH, 1);
```

```
        date.set(Calendar.DATE, 1);

        Flight RVKAK = new Flight("Reykjavík", "Akureyri", startDate);
        Flight AKRVK = new Flight("Akureyri", "Reykjavík", endDate);
        Hotel hotelAK = new Hotel("hotelAK", "Akureyri", 3.8);
        Daytour daytourAK = new Daytour("daytourAK", "Akureyri", date,
        ↪  20000);

        List<Flight> packageFlight = new ArrayList<>();
        packageFlight.add(RVKAK);
        packageFlight.add(AKRVK);
        List<Hotel> packageHotel = new ArrayList<>();
        packageHotel.add(hotelAK);
        List<Daytour> packageDaytour = new ArrayList<>();
        packageDaytour.add(daytourAK);

        Package minnPakki = new Package();
        minnPakki.setConfirmedFlights(packageFlight);
        minnPakki.setConfirmedHotels(packageHotel);
        minnPakki.setConfirmedDaytours(packageDaytour);

        List<Package> packageList = new ArrayList<>();
        packageList.add(minnPakki);

        System.out.println("searchForPackage");
        String from = "Reykjavík";
        String to = "Akureyri";
        //Calendar startDate = null;
        //Calendar endDate = null;
        SearchManager instance = new SearchManager();
        List<Package> expResult = packageList;
        List<Package> result = instance.searchForPackage(from, to,
        ↪  startDate, endDate);
        assertEquals(expResult, result);
        // TODO review the generated test code and remove the default
        ↪  call to fail.
        fail("The test case is a prototype.");
    }
```

Test subject

```
public class SearchManager {

    private FlightManager fm = new FlightManager();
    private HotelManager hm = new HotelManager();
    private DaytourManager dm = new DaytourManager();

    //Pakkar eru byggðir upp þannig að hver pakki inniheldur flug frá
     ↪  staðsetningu A til B, flug til baka(frá B til A),
    //hótel á staðsetningu B, daytour á staðsetningu B
    public List<Package> searchForPackage(String from, String to,
     ↪  Calendar startDate, Calendar endDate) {

        //ArrayList<Package> packageList = new ArrayList<Package>();

        List<Package> packageList = new ArrayList<>();

        List<Flight> arrFlights = fm.getarrFlightList();
        List<Flight> depFlights = fm.getdepFlightList();
        List<Hotel> hotels = hm.getHotelList();
        List<Daytour> daytours = dm.getDaytourList();

        //maxIndex þjónar eingöngu þeim tilgangi að passa að ekki sé
         ↪  verið að ítra út fyrir listanna hotels og daytours
        int maxIndex = Math.min(hotels.size(), daytours.size());

        //ef fjöldi flugferða sem fara frá eða til staðars á tiltekinni
         ↪  dagsetningu sem notandi vill eru núll,
        //þá eru engir pakkar fyrir sett skilyrði í boði (return null)
        //einnig gildir að ef fjöldi hótela í boði á stað sem notandi
         ↪  vill er núll, þá eru engir pakkar fyrir sett skilyrði í boði
         ↪  (return null)
        if(depFlights.isEmpty() || arrFlights.isEmpty() ||
         ↪  hotels.isEmpty()) {
            return null;
        }
        //Annars búum við til lista af pökkum sem allir hafa sömu flugin
         ↪  frá og til áfangastaðar en með mismunandi hótelum og daytours
```

```
        else {
            for(int i = 0; i < maxIndex; i++) {
                packageList.add(makePackage(depFlights, arrFlights,
                ↪  hotels, daytours, i));
            }

            return packageList;
        }
    }
    //Aðferð sem býr til pakka útfrá skilyrðum notanda
    //Þessi aðferð gerir ráð fyrir að það skipti ekki máli hvaða flug
    ↪  fari fram og til baka, heldur AÐ það sé flug sem fer fram og til
    ↪  baka
    public Package makePackage(List<Flight> depFlights, List<Flight>
    ↪  arrFlights, List<Hotel> hotels, List<Daytour> daytours, int
    ↪  index) {

        Package pakki = new Package();

        List<Flight> packageFlight = Arrays.asList(depFlights.get(0),
        ↪  arrFlights.get(0));
        List<Hotel> packageHotel = Arrays.asList(hotels.get(index));
        List<Daytour> packageDaytour =
        ↪  Arrays.asList(daytours.get(index));

        pakki.setConfirmedFlights(packageFlight);
        pakki.setConfirmedHotels(packageHotel);
        pakki.setConfirmedDaytours(packageDaytour);

        return pakki;
    }
}
```

Mock Objects

```
public class Daytour {

    private String name;
    private String location;
    private Calendar date;
    private int price;

    public Daytour(String daytourName, String daytourLocation, Calendar
     ↪  daytourDate, int daytourPrice) {
        name = daytourName;
        location = daytourLocation;
        date = daytourDate;
        price = daytourPrice;
    }
}

public class DaytourManager {

    private List<Daytour> daytourList;

    public List searchForDaytours(/*...*/) {

        Calendar date = Calendar.getInstance();
        date.set(Calendar.YEAR, 2019);
        date.set(Calendar.MONTH, 1);
        date.set(Calendar.DATE, 1);

        Daytour daytourAK = new Daytour("daytourAK", "Akureyri", date,
         ↪  20000);
        daytourList.add(daytourAK);

        return daytourList;
    }

    public List getDaytourList() {
        return daytourList;
    }
```

```
}

public class Flight {

    private String from;
    private String to;
    private Calendar date;

    public Flight(String flightFrom, String flightTo, Calendar
     ↪  flightDate) {
        from = flightFrom;
        to = flightTo;
        date = flightDate;
    }
}

public class FlightManager {

    private List<Flight> depFlights;
    private List<Flight> arrFlights;

    public List searchForFlights(String from, String to, Calendar date) {

        //TEST
        Calendar departure = Calendar.getInstance();
        departure.set(Calendar.YEAR, 2019);
        departure.set(Calendar.MONTH, 1);
        departure.set(Calendar.DATE, 1);

        Calendar arrival = Calendar.getInstance();
        arrival.set(Calendar.YEAR, 2019);
        arrival.set(Calendar.MONTH, 1);
        arrival.set(Calendar.DATE, 10);

        Flight RVKAK = new Flight("Reykjavík", "Akureyri", departure);
        Flight AKRVK = new Flight("Akureyri", "Reykjavík", arrival);

        depFlights.add(RVKAK);
        arrFlights.add(AKRVK);
```

```
        return depFlights;
    }

    public List getdepFlightList() {
        return depFlights;
    }
    public List getarrFlightList() {
        return arrFlights;
    }
}

public class Hotel {

    private String name;
    private String location;
    private double rating;

    public Hotel(String hotelName, String hotelLocation, double
     ↪  hotelRating) {
        name = hotelName;
        location = hotelLocation;
        rating = hotelRating;
    }
}

public class HotelManager {

    private List<Hotel> hotelList;

    public List searchForHotels(/*...*/) {

        Hotel hotelAK = new Hotel("hotelAK", "Akureyri", 3.8);
        hotelList.add(hotelAK);

        return hotelList;
    }

    public List getHotelList() {
        return hotelList;
```

```
    }
}
```