DEPARTMENT OF PHYSICS & ASTRONOMY
PHAS3459: Scientific Computing Using Object Oriented Languages
Final Examination
13:30 – 16:30: 17th January 2018

Please read the exam guidelines, rules, instructions and marking criteria at `https://moodle.ucl.ac.uk/mod/page/view.php?id=2016117` (linked from the PHAS3459 Examinations Moodle page).

This exam is worth 50% of your final mark for the course. The duration of the exam is 3 hours. The Java source code of your solution to the programming exercise should be uploaded using Moodle under the section headed "Exam II".

Each class should be uploaded as a separate file. Your classes should be created in a package called "`exam2`". This package must contain at least one "`main`" class: you should also include any other classes you create. You must upload all your classes used in your solution, including any you have copied or imported from earlier coursework modules. The code you upload **must be self-contained**: the marker must be able to compile and run it using only the classes uploaded and the Java API. If you use your own classes from earlier modules, make sure you copy them into the `exam2` package and upload them along with any new classes you create during the exam.

You are advised to read the entire exam paper before starting work.

In this exam you will write an application to plan journeys using information about scheduled flights between various airports.

The input data files are provided in the web directory `http://www.hep.ucl.ac.uk/undergrad/3459/exam-data/2017-18/` and are CSV files: the fields on each line are separated by commas and possibly white space after each comma. Their contents are described below.

The file **`airports.txt`** contains details of airports. Each line contains the following information for a single airport:

- airport code (e.g. LHR)
- full name (e.g. London Heathrow)
- time zone (e.g. Europe/London)

The file **`flights.txt`** contains details of flights. Each line contains the following information for a single flight:

- flight code (e.g. AB002)
- origin airport code (e.g. LHR for a flight departing from London Heathrow)
- destination airport code (e.g. CDG for a flight arriving at Paris Charles de Gaulle)
- departure date in the format yyyy-mm-dd, e.g. 2018-01-02
- departure time in local time at the origin airport (e.g. 13:10)
- arrival date in the format yyyy-mm-dd, e.g. 2018-01-02
- arrival time in local time at the destination airport (e.g. 14:55)
- cost of the flight in pounds

Part 1: 20/50 marks

Write a program, with a `main` class called `ExamPart1`, and any other classes you choose, to do the following:

- Read the data from the files detailed above, and store them in one or more appropriate data structures.
- For each flight, print the following information:
  - the flight code;
  - the full name of the origin and destination airports;
  - the date and local time of departure and arrival;
  - the duration of the flight in minutes;
  - the cost of the flight.

You may adapt the Java code given here to calculate the duration between two times in different time zones:

```
LocalDateTime lt1 = LocalDateTime.parse("2018-01-02T09:05");
ZoneId         z1 = ZoneId.of("Europe/London");
ZonedDateTime zt1 = ZonedDateTime.of(lt1, z1);

LocalDateTime lt2 = LocalDateTime.parse("2018-01-02T11:20");
ZoneId         z2 = ZoneId.of("Europe/Paris");
ZonedDateTime zt2 = ZonedDateTime.of(lt2, z2);

long duration = zt1.until(zt2, ChronoUnit.MINUTES);
```

Part 2: 20/50 marks

- Create an interface to represent a way of filtering flights: it should declare a method that takes a list of flights and returns a new list of the flights that meet some criterion.
- Create implementations to select journeys:
  - with a specified origin and destination airport
  - with a duration no more than than some specified limit
  - with a cost no greater than a specified limit
- Write a program, with a `main` class called `ExamPart2`, using the interface and implementations, and any other classes you choose, to do the following:
  - print details of all flights from LHR to ATH that cost no more than £200
  - print details of all flights from LHR to ATH that take no more than four hours in total
  - find the quickest flight from LHR to ATH that costs no more than £200.

Part 3: 10/50 marks

In this section you will consider both direct flights and journeys comprising two flights, with a stopover in another airport. For example, a journey from London Heathrow (LHR) to Cape Town (CPT) might comprise a flight from LHR to FRA (Frankfurt) followed by a flight from FRA to CPT. The cost of a journey is the sum of the cost of the flights included.

Write a program, with a `main` class called `ExamPart3`, and any other classes you choose, to find the cheapest journey from LHR to CPT, including both direct flights and two-flight journeys, where the total journey duration is less than 24 hours.

END OF PAPER