COMPUTER SCIENCE A
SECTION II
Time—1 hour and 45 minutes
Number of questions—4
Percent of total grade—50

Directions: SHOW ALL YOUR WORK. REMEMBER THAT PROGRAM SEGMENTS ARE TO BE WRITTEN IN JAVA.

Notes:
- Assume that the classes listed in the Quick Reference found in the Appendix have been imported where appropriate.
- Unless otherwise noted in the question, assume that parameters in method calls are not `null` and that methods are called only when their preconditions are satisfied.
- In writing solutions for each question, you may use any of the accessible methods that are listed in classes defined in that question. Writing significant amounts of code that can be replaced by a call to one of these methods may not receive full credit.

1. A travel agency maintains a list of information about airline flights. Flight information includes a departure time and an arrival time. You may assume that the two times occur on the same day. These times are represented by objects of the `Time` class.

   The declaration for the `Time` class is shown below. It includes a method `minutesUntil` that returns the difference (in minutes) between the current `Time` object and another `Time` object.

```
public class Time
{
  /** @return difference, in minutes, between this time and other;
   *          difference is negative if other is earlier than this time
   */
  public int minutesUntil(Time other)
  {  /* implementation not shown */  }

  // There may be instance variables, constructors, and methods that are not shown.
}
```

   For example, assume that `t1` and `t2` are `Time` objects where `t1` represents 1:00 P.M. and `t2` represents 2:15 P.M. The call `t1.minutesUntil(t2)` will return `75` and the call `t2.minutesUntil(t1)` will return `-75`.

The declaration for the `Flight` class is shown below. It has methods to access the departure time and the arrival time of a flight. You may assume that the departure time of a flight is earlier than its arrival time.

```
public class Flight
{
    /** @return time at which the flight departs
     */
    public Time getDepartureTime()
    { /* implementation not shown */ }

    /** @return time at which the flight arrives
     */
    public Time getArrivalTime()
    { /* implementation not shown */ }

   // There may be instance variables, constructors, and methods that are not shown.
}
```

AP® COMPUTER SCIENCE A
2008 SCORING GUIDELINES

Question 1: Flight List

Part A:	getDuration	**4 points**

+1 handle empty case
- **+1/2** check if `flights` is empty
- **+1/2** return 0 if empty

+1 access start time
- **+1/2** access `flights.get(0)`
- **+1/2** correctly call `getDepartureTime` on a flight

+1 access end time
- **+1/2** access `flights.get(flights.size()-1)`
- **+1/2** correctly call `getArrivalTime` on a flight

+1 calculate and return duration
- **+1/2** call `minutesUntil` using `Time` objects
- **+1/2** return correct duration (using `minutesUntil`)

Part B:	getShortestLayover	**5 points**

+1 handle case with 0 or 1 flight
- **+1/2** check if `flights.size() < 2`
- **+1/2** return -1 in that case

+1 traverse `flights`
- **+1/2** correctly access an element of `flights` (in context of loop)
- **+1/2** access all elements of `flights` (lose this if index out-of-bounds)

+2 1/2 find shortest layover (in context of loop)
- **+1** get layover time between successive flights (using `minutesUntil`)
- **+1/2** compare layover time with some previous layover
- **+1** correctly identify shortest layover

+1/2 return shortest layover