Object-Based Programming (25 marks):
Answer both parts below.

The API for the `CallRecord` class is given below. Each instance of this class has a start time, a duration (in seconds), a dialled number, and a destination city.

Constructor Summary	
`CallRecord ()` Constructs a new Call Record.	
Method Summary	
`int`	`getAreaCode ()` Returns the three digit area code of the number dialled in this Call Record.
`String`	`getDestination ()` Returns the destination city of the number dialled in this Call Record.
`Time`	`getDuration ()` Returns the length of the call for this Call Record.
`int`	`getNumber ()` Returns the seven digit number dialled in this Call Record.
`Time`	`getStart ()` Returns the start time of the call for this Call Record.

The API for the `CallingCard` class is given below. Each instance of this class stores an amount of Time that can be used to make phone calls.

Constructor Summary	
`CallingCard ()` Constructs a new Calling Card with zero Time available.	
Method Summary	
`void`	`addTime (Time amount)` Adds the given amount of time to this Calling Card.
`Time`	`getTime ()` Returns the amount of Time available on this Calling Card.
`void`	`useTime (Time amount)` Subtracts the given amount of time from this Calling Card. Note: this method will produce a run-time error if the given amount of Time is greater than the available Time on this Calling Card.

The API for the `Time` class is given below. Each instance of this class represents an amount of minutes and seconds. The amount of minutes and seconds will be integers between 0 and 59 (inclusive), and the amount of hours will never be less than zero.

Field Summary	
`static int`	`HOURS` The number of hours in a day.
`static int`	`MINUTES` The number of minutes in an hour.
`static int`	`SECONDS` The number of seconds in a minute.

Constructor Summary
`Time()` Constructs a new Time amount with 0 hours, 0 minutes, and 0 seconds.
`Time(int h, int m, int s)` Constructs a new Time amount with `h` hours, `m` minutes and `s` seconds. Note: if `h`, `m`, or `s` are invalid, a run-time error will occur.

Method Summary	
`Time`	`addTime(Time amount)` Returns a Time that is the sum of this Time and the given `amount`.
`int`	`getHours ()` Returns the number of hours in this Time.
`int`	`getMinutes ()` Returns the number of minutes in this Time.
`int`	`getSeconds ()` Returns the number of seconds in this Time.
`boolean`	`isGreaterOrEquals(Time amount)` Returns `true` if this Time amount is greater than of equal to `amount`.
`Time`	`subtractTime(Time amount)` Returns a Time that is the difference of this Time and the given `amount`.

Surname:______________ First name:_______________ Student #: ________________

Part 1 (**10 marks**):

Write a code fragment in JAVA that will calculate the end Time of a given phone call.

```
// calculate end time
CallRecord phoneCall;
Time endTime;
```

Surname:______________ First name:_______________ Student #: ________________

Part 2 (**15 marks**):

Write a code fragment in JAVA that will attempt to pay for the given call with the given Calling Card. The Time used by the call will be taken off of the Calling Card. If the Calling Card has enough Time for the call, unpaid Time will be set to `null`. Otherwise, the unpaid Time will be the amount of Time from the call that was not paid for by the Calling Card. In both situations, the Calling Card will be updated to reflect the amount of Time that it still has available for calls (which may be zero).

```
// calculate unpaid time
CallRecord call;
CallingCard card;
Time unpaid;
```