# Startdocument for Taxi

Startdocument of **Terry Ioannou. Student number** 4976355**.

# Problem Description

A taxi company owns a number of taxis and operates the following prices: € 0.50 per kilometre driven. An additional € 0.17 for each minute driven. A 15% surcharge applies from 10 pm on Friday evening until 7.00 am on Monday morning (determined by the start time). A program must be developed in which the start and end time (format hhmm) and the day of the week and the distance can be entered for each taxi journey. The payable amount for each journey must be calculated and shown. The total income, the average distance and the longest distance must also be shown cumulatively.

## Input & Output

In this section the in- and output of the application will be described.

### Input

In the table below all the input will be from the company when adding a new driver to the company

| | | |
|---|---|---|
| Driver | String | not empty |
| Taxi | Int | not empty |
| Kilometers | int | not empty |
| Seconds | int | not empty |
| Date Time | int | not empty |
| Price | float | not empty |
| Start Time | float | not empty |

### Output

| Case | Type |
|---|---|
| Number of Drivers | `ArrayList<Taxis>`:int |
| Journeys | `ArrayList<Calculation of Journey>`:int |
| Longest Journey | int |
| Shortest Journey | int |
| Average Journey | int |
| Most Profit | float |
| Lowest Profit | float |
| Check Calendar | int |
| Calculate Time | float |
| Calculate Kilometers | int |
| Cost of Destination | float |

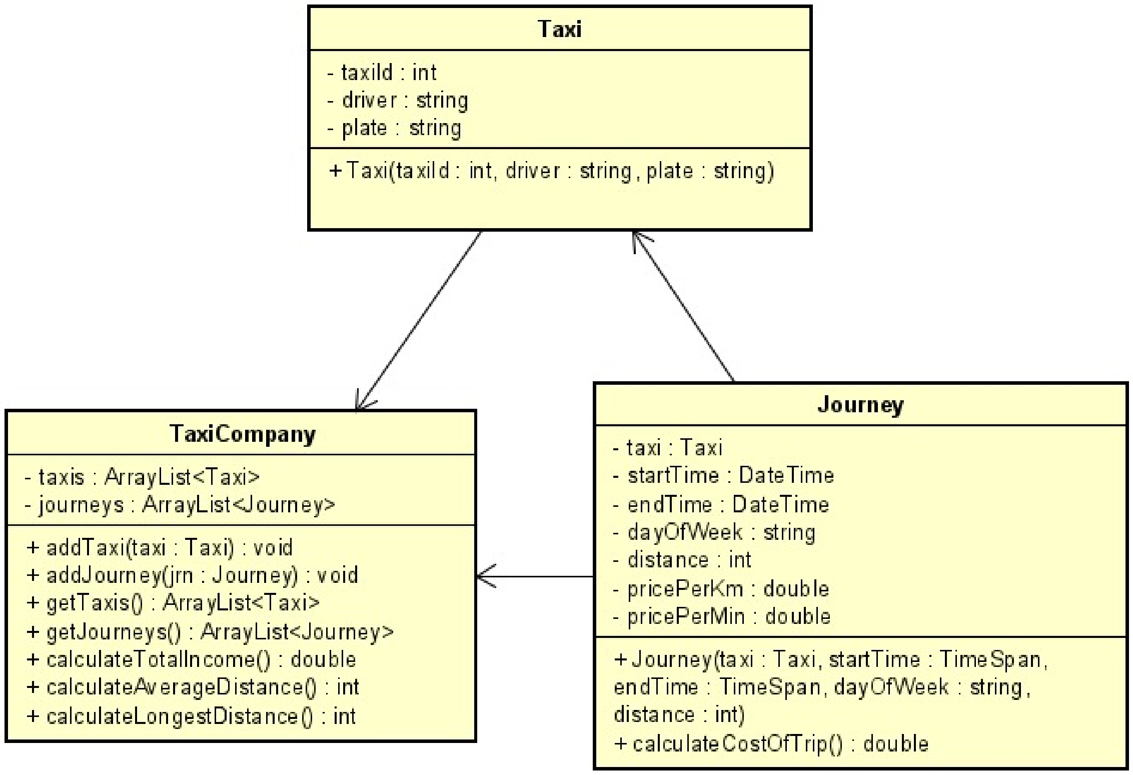
## Calculations

| Case | Calculation |
|---|---|
| Longest Journey | Will calculate the longest Journey in the database |
| Shortest Journey | Will calculate the shortest Journey in the database |
| Average Journey | Will calculate the Average Journey in the database |
| Most Profit | Will calculate the most profit from all Journeys in the database |
| Lowest Profit | Will calculate the lowest profit from all Journeys in the database |
| Check Calendar | Will check the date and time if it's 10:00 P.M. Friday there will be a bonus 15% profit of everything until 07:00 A.M. Monday |
| Calculate Time | Will calculate the time of the journey from point A to B |
| Calculate Kilometers | Will calculate the Kilometers of journey |
| Cost of Destination | Will calculate all the cost including Kilometers, Minutes/Seconds and the date |

## Remarks

- Input will be validated.
- Only the Client class will contain `Client.out.println`
- Unit Tests will be provided.

# Class Diagram Taxi


Taxi
- taxiId : int
- driver : string
- plate : string
+Taxi(taxiId : int, driver : string, plate : string)
TaxiCompany
- taxis : ArrayList<Taxi>
- journeys : ArrayList<Journey>
+ addTaxi(taxi : Taxi) : void
+ addJourney(jrn : Journey) : void
+ getTaxis() : ArrayList<Taxi>
+ getJourneys() : ArrayList<Journey>
+ calculateTotalIncome() : double
+ calculateAverageDistance() : int
+ calculateLongestDistance() : int
Journey
- taxi : Taxi
- startTime : DateTime
- endTime : DateTime
- dayOfWeek : string
- distance : int
- pricePerKm : double
- pricePerMin : double
+Journey(taxi : Taxi, startTime : TimeSpan, endTime : TimeSpan, dayOfWeek : string, distance : int)
+calculateCostOfTrip() : double

# Test plan

In this section the testcases will be described to test the application.

## Test Data

In the following table you'll find all the data that is needed for testing.

### Taxis

| Taxi | Input | Code |
|---|---|---|
| Taxi ID | Taxi ID:01 | +addTaxi(taxi:Taxi ID) |
| Taxi ID | Taxi ID:02 | +addTaxi(taxi:Taxi ID) |

| Driver | Input | Code |
|---|---|---|
| Jake | number: 1 | new Driver(Jake) |
| Daniel | number: 2 | new Driver(Daniel) |

### Company

| Input | Code |
|---|---|
| Number of journeys | +getJourneys(journeys: Journeys) |
| Number of Drivers | +getDrivers(drives: Drivers) |
| Longest Journey | +getLongestJourney(journey highest :int) |
| Shortest Journey | +getLongestJourney(journey lowest: int) |
| Average Journey | +getLongestJourney(journey average: int) |
| Most Profit | +getMostProfit(Price highest:float) |
| Lowest Profit | +getMostProfit(Price lowest:float) |

### Calculation of Journey

| input | Code |
|---|---|
| Insert in Calendar the day of the week | Date Time startTime = new DateTimepicker1 |
| Start timer and Stop timer | timer1.Start(); |
| Calculate the distance of KM | Insert textbox:int |
| Calculate the Destination of the Journey | Price(Kilometers+timer1+Date Time) |

## Test Cases

In this section the testcases will be described. Every test case should be executed with the test data as starting point.

### #1 Taxi

Verifying the name of the Driver. The user should not leave empty

| Step | Input | Action | Expected output |
|---|---|---|---|
| 1 | Driver Name | `getTaxiID()`:notempty | Taxi ID & Driver |

### #2 Add Taxi Driver to Company

Testing the user should be able to add songs to Custom Play List

| Step | Input | Action | Expected output |
|---|---|---|---|
| 1 | Add new Driver | +addTaxi(taxi:Taxi) | added to Company |

## #3 Calculation of Journey from Taxi Driver

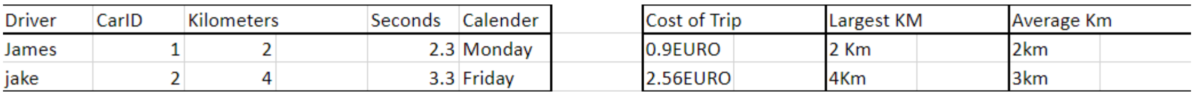
Calculating the price of the destination step by step

| Step | Input | Action | Expected output |
|---|---|---|---|
| 1 | Check how much Kilometers is the destination | input:(Kilometers: int) | echo Kilometers |
| 2 | Check how much time it took to reach said destination | timer1:Start /timer1.Stop | echo lbl.timer1 |
| 3 | Check the calendar to see day of the week | lblCalender(toString) | echo lblCalender |
| 4 | Sum up all the Cost of that journey | Price: (Kilometers+timer1+Calendar) | echo Price |

## #4 Company Information

In the company that owns Taxis will be able to see the information that they decide from the database.

| Step | Input | Action | Expected output |
|---|---|---|---|
| 1 | Company records | Input a quarry that delivers all drivers | List of all drivers |
| 2 | Company records | input a quarry that delivers all Journeys | List of records of Journeys |
| 3 | Company records | input a quarry that displays Longest journey | Display Longest Journey |
| 4 | Company records | input a quarry that displays Shortest journey | Display shortest Journey |
| 5 | Company records | input a quarry that displays Average journey | Display average Journey |
| 6 | Company records | input a quarry that displays Most profit | Display Highest profit from Journey |
| 7 | Company records | input a quarry that displays Lowest profit | Display lowest profit from Journey |



| Driver | CarID | Kilometers | Seconds | Calender |
|---|---|---|---|---|
| James | 1 | 2 | 2.3 | Monday |
| jake | 2 | 4 | 3.3 | Friday |

| Cost of Trip | Largest KM | Average Km |
|---|---|---|
| 0.9EURO | 2 Km | 2km |
| 2.56EURO | 4Km | 3km |