Data Structures and Algorithms

COMP 103 2019-20 Semester 2

Lecture 07b

Dr. Kerese Manueli

kerese.manueli@ecs.vuw.ac.nz

Victoria University of Wellington

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

VICTORIA UNIVERSITY OF WELLINGTON TE HERENGA WAKA

School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

- XMUT103 home
- Course Outline
- Lecture Schedule
- Weekly Timetable
- Assignments
- Submission
- Your Marks
- People
- Java Resources
- Java documentation
- Tutor Space
- Assignment Admin
- Plagiarism Log

School of Engineering and Computer Science ▸ Courses/XMUT103_2020T1 ▸ Assignment2PartA

Introduction to Data Structures and Algorithms
Assignment 2 Part A: Using Collections

- Due 6 May , 7pm

Resources and links

- Download zip file containing the necessary code and data.
- Java Documentation
- Submit your answers
- Marks and feedback

What To Hand In

- `WellingtonTrains.java`

Do not rename this file and do not rename any other files provided in the Template.

When you have submitted it, check that you can read the file listed on the submission page, and complete the submission process.

Wellington Trains (Weight: 2/3)

`WellingtonTrains` is a program to answer queries about Wellington train stations, train lines and the timetables for the train services on those lines.

(See https://www.metlink.org.nz/#plan for Metlink's equivalent program for the whole of the Wellington Regional Transport system. Your program is not

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

Stations, Lines, and Services:

Your program has to deal with three kinds of information:

- Train Stations
- Train Lines: a sequence of stations. For example, the Wellington_Johnsonville line, which starts at Wellington station, and ends at Johnsonville station, with 7 stations in between. To make things easier, we treat the inbound and outbound as two separate train lines, so the Wellington_Melling line is different from the Melling_Wellington line. (They have the same stations, but the sequence in one is the reverse of the sequence in the other.)
- Train Service: a schedule/timetable for a particular train running on a given train line. For example, the 11:32 train on the Wellington_Johnsonville line that leaves Wellington station at 11:32, leaves Crofton-Downs at 11:40, ... gets in to Johnsonville at 11:55.
  A Train Service is specified by a sequence of times - the time that the train leaves the first station, followed by the times that the train gets to each of the remaining stations on the line. Note that some train services don't stop at every station on the line. If a train doesn't stop at a station, the corresponding time will be -1.

Queries

The basic queries the program should be able to handle are the following:

Hints provided in Lec6c & Lec7a

1. List all the stations in the region.
2. List all the train lines in the region
3. List the train lines that go through a given station
4. List the stations along a given train line
5. Print the name of a train line that goes from a station to a destination station. (The train line must go the correct direction.)
6. (Completion) Find the next train service for each line at a station after the specified time
7. (Completion) Find a trip between two stations (on the same line), after the specified time.
   Find the train line, the time that next service on that line will leave the first station, the time that the service will arrive at the destination station, and the number of fare zones the trip goes through.
8. (Challenge:) More complex trips involve going from the first station to an exchange station on one train line, then going from the exchange station to the second station on another train line. (In larger cities, trips might require more than 2 train lines, but the Wellington region has a central hub, so you never need more than two segments for a trip.)
   Find the best trip (earliest arrival) between a starting station and a destination station. If the trip requires two train lines, print out the starting time, the first train line, the exchange station, the arriving and departing times at the exchange station, the second train line, and the arrival time at the destination, and the total number of fare zones.

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

Stations, Lines, and Services:

Your program has to deal with three kinds of information:

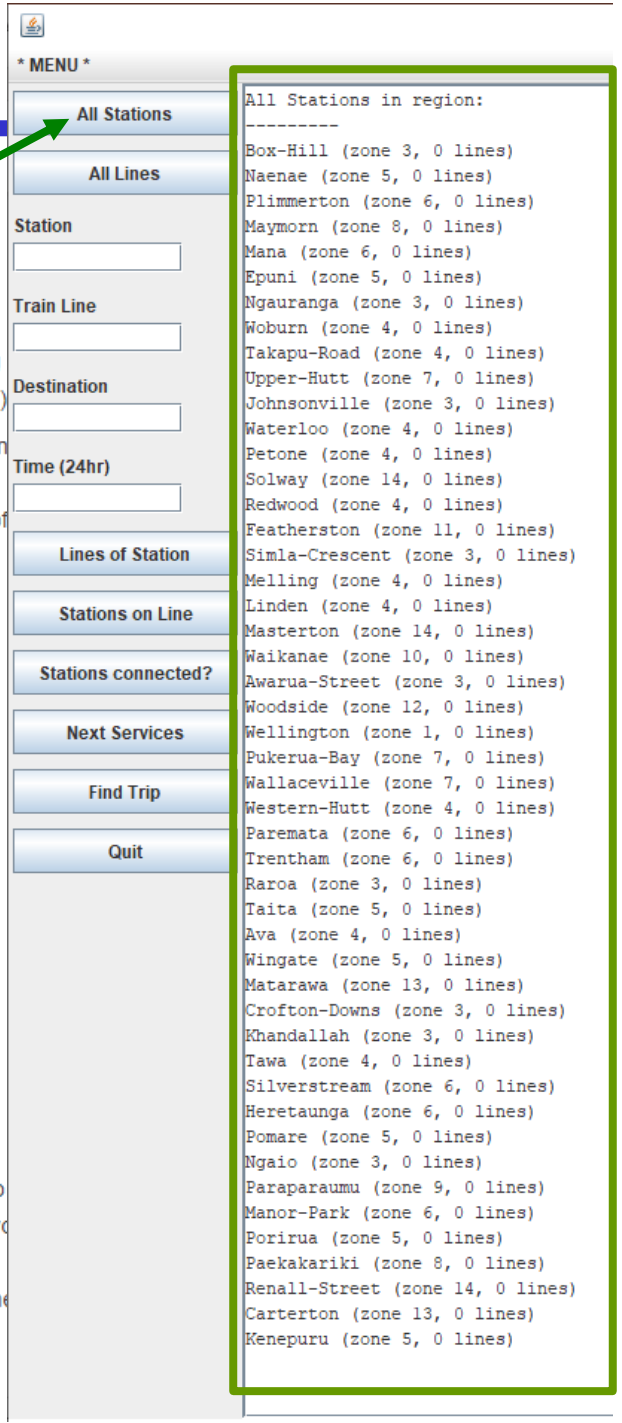
- Train Stations
- Train Lines: a sequence of stations. For example, the Wellington_Johnsonville line, which starts at Wellington station, and ends at Johnsonville station, with 7 stations in between. To make things easier, we treat the inbound and outbound as two separate train lines, so the Wellington_Melling line is different from the Melling_Wellington line. (They have the same stations, but the sequence in one is the reverse of the sequence in the other.)
- Train Service: a schedule/timetable for a particular train running on a given train line. For example, the 11:32 train on the Wellington_Johnsonville lin that leaves Wellington station at 11:32, leaves Crofton-Downs at 11:40, ... gets in to Johnsonville at 11:55.
  A Train Service is specified by a sequence of times - the time that the train leaves the first station, followed by the times that the train gets to each of the remaining stations on the line. Note that some train services don't stop at every station on the line. If a train doesn't stop at a station, the corresponding time will be -1.

Queries

The basic queries the program should be able to handle are the following:

Hints provided in Lec6c & Lec7a

1. List all the stations in the region.
2. List all the train lines in the region
3. List the train lines that go through a given station
4. List the stations along a given train line
5. Print the name of a train line that goes from a station to a destination station. (The train line must go the correct direction.)
6. (Completion) Find the next train service for each line at a station after the specified time
7. (Completion) Find a trip between two stations (on the same line), after the specified time.
   Find the train line, the time that next service on that line will leave the first station, the time that the service will arrive at the destination station, and the number of fare zones the trip goes through.
8. (Challenge:) More complex trips involve going from the first station to an exchange station on one train line, then going from the exchange station to the second station on another train line. (In larger cities, trips might require more than 2 train lines, but the Wellington region has a central hub, so yo never need more than two segments for a trip.)
   Find the best trip (earliest arrival) between a starting station and a destination station. If the trip requires two train lines, print out the starting time, the first train line, the exchange station, the arriving and departing times at the exchange station, the second train line, and the arrival time at the destination, and the total number of fare zones.



Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

Stations, Lines, and Services:

Your program has to deal with three kinds of information:

- Train Stations
- Train Lines: a sequence of stations. For example, the Wellington_Johnsonville line, which starts at Wellington station, and ends at Johnsonville station, with 7 stations in between. To make things easier, we treat the inbound and outbound as two separate train lines, so the Wellington_Melling line is different from the Melling_Wellington line. (They have the same stations, but the sequence in one is the reverse of the sequence in the other.)
- Train Service: a schedule/timetable for a particular train running on a given train line. For example, the 11:32 train on the Wellington_Johnsonville line that leaves Wellington station at 11:32, leaves Crofton-Downs at 11:40, ... gets in to Johnsonville at 11:55.
  A Train Service is specified by a sequence of times - the time that the train leaves the first station, followed by the times that the train gets to each of the remaining stations on the line. Note that some train services don't stop at every station on the line. If a train doesn't stop at a station, the corresponding time will be -1.

Queries

Hints provided in Lec6c & Lec7a

The basic queries the program should be able to handle are the following:

1. List all the stations in the region.
2. List all the train lines in the region
3. List the train lines that go through a given station
4. List the stations along a given train line
5. Print the name of a train line that goes from a station to a destination station. (The train line must go the correct direction.)
6. (Completion) Find the next train service for each line at a station after the specified time
7. (Completion) Find a trip between two stations (on the same line), after the specified time.
   Find the train line, the time that next service on that line will leave the first station, the time that the service will arrive at the destination station, and the number of fare zones the trip goes through.
8. (Challenge:) More complex trips involve going from the first station to an exchange station on one train line, then going from the exchange station to the second station on another train line. (In larger cities, trips might require more than 2 train lines, but the Wellington region has a central hub, so you never need more than two segments for a trip.)
   Find the best trip (earliest arrival) between a starting station and a destination station. If the trip requires two train lines, print out the starting time, the first train line, the exchange station, the arriving and departing times at the exchange station, the second train line, and the arrival time at the destination, and the total number of fare zones.

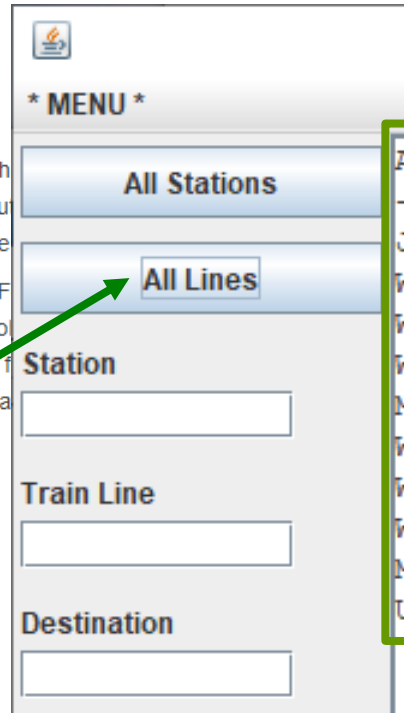
Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

Stations, Lines, and Services:

Your program has to deal with three kinds of information:

- Train Stations
- Train Lines: a sequence of stations. For example, the Wellington_Johnsonville line, wh
  station, with 7 stations in between. To make things easier, we treat the inbound and ou
  line is different from the Melling_Wellington line. (They have the same stations, but the
- Train Service: a schedule/timetable for a particular train running on a given train line. F
  that leaves Wellington station at 11:32, leaves Crofton-Downs at 11:40, ... gets in to Jo
  A Train Service is specified by a sequence of times - the time that the train leaves the f
  the remaining stations on the line. Note that some train services don't stop at every sta
  corresponding time will be -1.




```
All Train Lines in region:
---------
Johnsonville_Wellington (9 stations, 45 services)
Wellington_Johnsonville (9 stations, 45 services)
Wellington_Melling (5 stations, 23 services)
Wellington_Waikanae (14 stations, 58 services)
Melling_Wellington (5 stations, 23 services)
Wellington_Masterton (12 stations, 5 services)
Waikanae_Wellington (14 stations, 59 services)
Wellington_Upper-Hutt (17 stations, 57 services)
Masterton_Wellington (12 stations, 5 services)
Upper-Hutt_Wellington (17 stations, 54 services)
```

Queries

The basic queries the program should be able to handle are the following:

Hints provided in Lec6c & Lec7a

1. List all the stations in the region.
2. List all the train lines in the region
3. List the train lines that go through a given station
4. List the stations along a given train line
5. Print the name of a train line that goes from a station to a destination station. (The train line must go the correct direction.)
6. (Completion) Find the next train service for each line at a station after the specified time
7. (Completion) Find a trip between two stations (on the same line), after the specified time.
   Find the train line, the time that next service on that line will leave the first station, the time that the service will arrive at the destination station, and the number of fare zones the trip goes through.
8. (Challenge:) More complex trips involve going from the first station to an exchange station on one train line, then going from the exchange station to the second station on another train line. (In larger cities, trips might require more than 2 train lines, but the Wellington region has a central hub, so you never need more than two segments for a trip.)
   Find the best trip (earliest arrival) between a starting station and a destination station. If the trip requires two train lines, print out the starting time, the first train line, the exchange station, the arriving and departing times at the exchange station, the second train line, and the arrival time at the destination, and the total number of fare zones.

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

Stations, Lines, and Services:

Your program has to deal with three kinds of information:

- Train Stations
- Train Lines: a sequence of stations. For example, the Wellington_Johnsonville line, which starts at Wellington station, and ends at Johnsonville station, with 7 stations in between. To make things easier, we treat the inbound and outbound as two separate train lines, so the Wellington_Melling line is different from the Melling_Wellington line. (They have the same stations, but the sequence in one is the reverse of the sequence in the other.)
- Train Service: a schedule/timetable for a particular train running on a given train line. For example, the 11:32 train on the Wellington_Johnsonville line that leaves Wellington station at 11:32, leaves Crofton-Downs at 11:40, ... gets in to Johnsonville at 11:55.
  A Train Service is specified by a sequence of times - the time that the train leaves the first station, followed by the times that the train gets to each of the remaining stations on the line. Note that some train services don't stop at every station on the line. If a train doesn't stop at a station, the corresponding time will be -1.

Queries

The basic queries the program should be able to handle are the following:

Hints provided in Lec6c & Lec7a

1. List all the stations in the region.
2. List all the train lines in the region
3. List the train lines that go through a given station
4. List the stations along a given train line
5. Print the name of a train line that goes from a station to a destination station. (The train line must go the correct direction.)
6. (Completion) Find the next train service for each line at a station after the specified time
7. (Completion) Find a trip between two stations (on the same line), after the specified time.
   Find the train line, the time that next service on that line will leave the first station, the time that the service will arrive at the destination station, and the number of fare zones the trip goes through.
8. (Challenge:) More complex trips involve going from the first station to an exchange station on one train line, then going from the exchange station to the second station on another train line. (In larger cities, trips might require more than 2 train lines, but the Wellington region has a central hub, so you never need more than two segments for a trip.)
   Find the best trip (earliest arrival) between a starting station and a destination station. If the trip requires two train lines, print out the starting time, the first train line, the exchange station, the arriving and departing times at the exchange station, the second train line, and the arrival time at the destination, and the total number of fare zones.

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

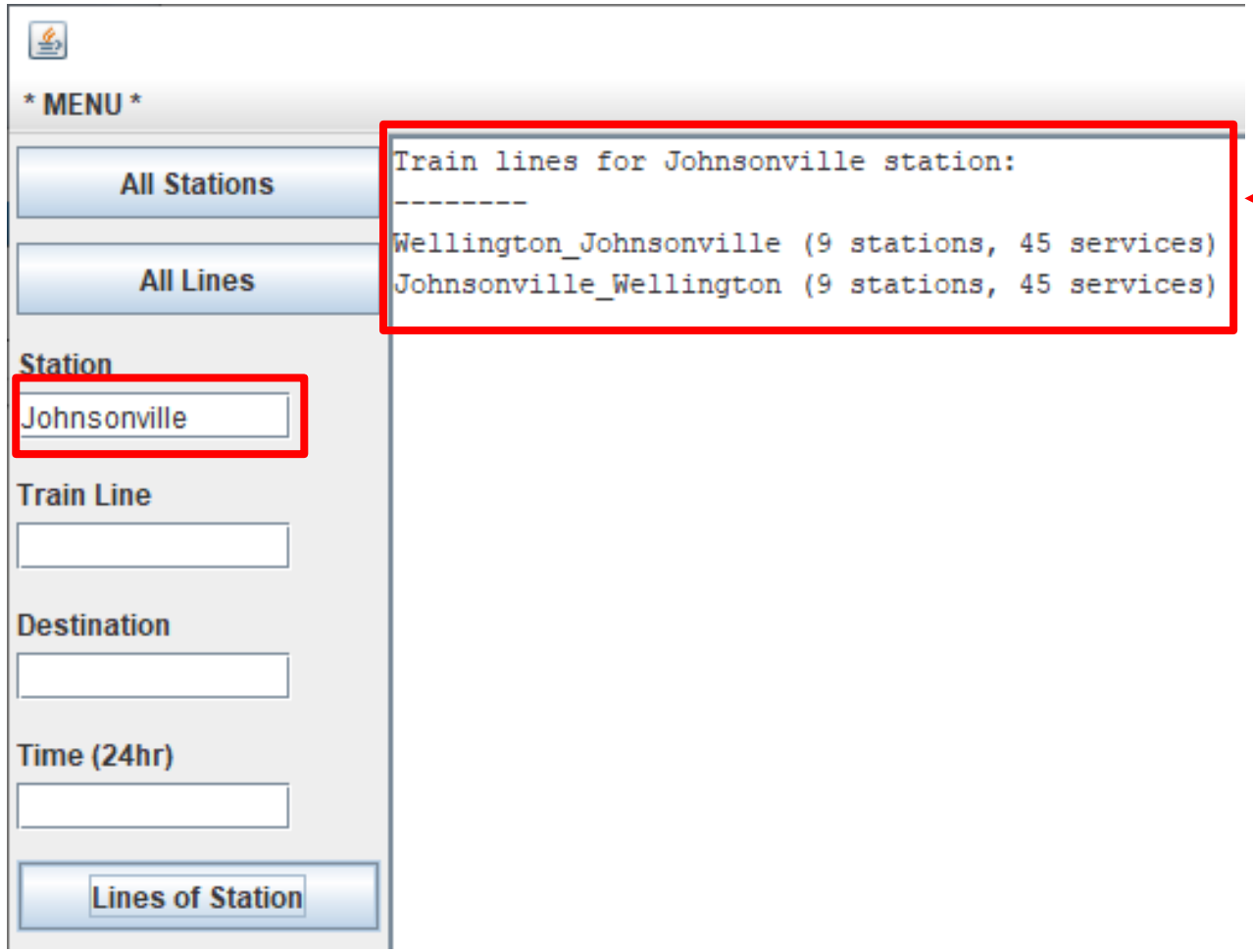
- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`

```
public void setupGUI(){
    UI.addButton("All Stations",    this::listAllStations);
    UI.addButton("All Lines",       this::listAllTrainLines);
    UI.addTextField("Station",      (String name) -> {this.stationName=name;});
    UI.addTextField("Train Line",   (String name) -> {this.lineName=name;});
    UI.addTextField("Destination", (String name) -> {this.destinationName=name;});
    UI.addTextField("Time (24hr)", (String time) ->
        {try{this.startTime=Integer.parseInt(time);}catch(Exception e){UI.println("Enter four
    UI.addButton("Lines of Station",    () -> {listLinesOfStation(this.stationName);});
    UI.addButton("Stations on Line",    () -> {listStationsOnLine(this.lineName);});
```

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`
- List the stations along a given train line by writing a `listStationsOnLine` method that **only** prints each station using `UI.println(station)`



1) Type Johnsonville in the Station text-box

2) Click on the Lines of Station button

3) Expected results

WellingtonTrains.java file opened in BlueJ

WellingtonTrains - WellingtonTrains

Class Edit Tools Options

WellingtonTrains

Compile Undo Cut Copy Paste Find... Close

```
/**
 * User interface has buttons for the queries and text fields to enter stations and train line
 * You will need to implement the methods here.
 */
public void setupGUI(){
    UI.addButton("All Stations",    this::listAllStations);
    UI.addButton("All Lines",       this::listAllTrainLines);
```

1) Before

3) After

2) Insert this method

```
public void setupGUI(){
    UI.addButton("All Stations",    this::listAllStations);
    UI.addButton("All Lines",       this::listAllTrainLines);
    UI.addTextField("Station",      (String name) -> {this.stationName=name;});
    UI.addTextField("Train Line",   (String name) -> {this.lineName=name;});
    UI.addTextField("Destination", (String name) -> {this.destinationName=name;});
    UI.addTextField("Time (24hr)", (String time) ->
        {try{this.startTime=Integer.parseInt(time);}catch(Exception e){UI.println("Enter four digits");}});
    UI.addButton("Lines of Station",     () -> {listLinesOfStation(this.stationName);});
    UI.addButton("Stations on Line",     () -> {listStationsOnLine(this.lineName);});
    UI.addButton("Stations connected?", () -> {checkConnected(this.stationName, this.destinationName);});
    UI.addButton("Next Services",        () -> {findNextServices(this.stationName, this.startTime);});
    UI.addButton("Find Trip",            () -> {findTrip(this.stationName, this.destinationName, this.startTime);});
    UI.addButton("Quit", UI::quit);
    UI.setDivider(1.0);
}

// Methods for loading data and answering queries

/*# YOUR CODE HERE */
public void listAllStations() {
}
```

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`

```
public void listAllTrainLines() {
}

public void listLinesOfStation(String stName) {
}
public void listStationsOnLine(String lnName) {
}

public void checkConnected(String stName, String desName) {
}

public void findNextServices(String stName, int stTime) {
}

public void findTrip(String stName, String desName, int stTime) {
}
```

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`

```
public void listAllTrainLines() {
}

public void listLinesOfStation(String stName) {
}
public void listStationsOnLine(String lnName) {
}
```

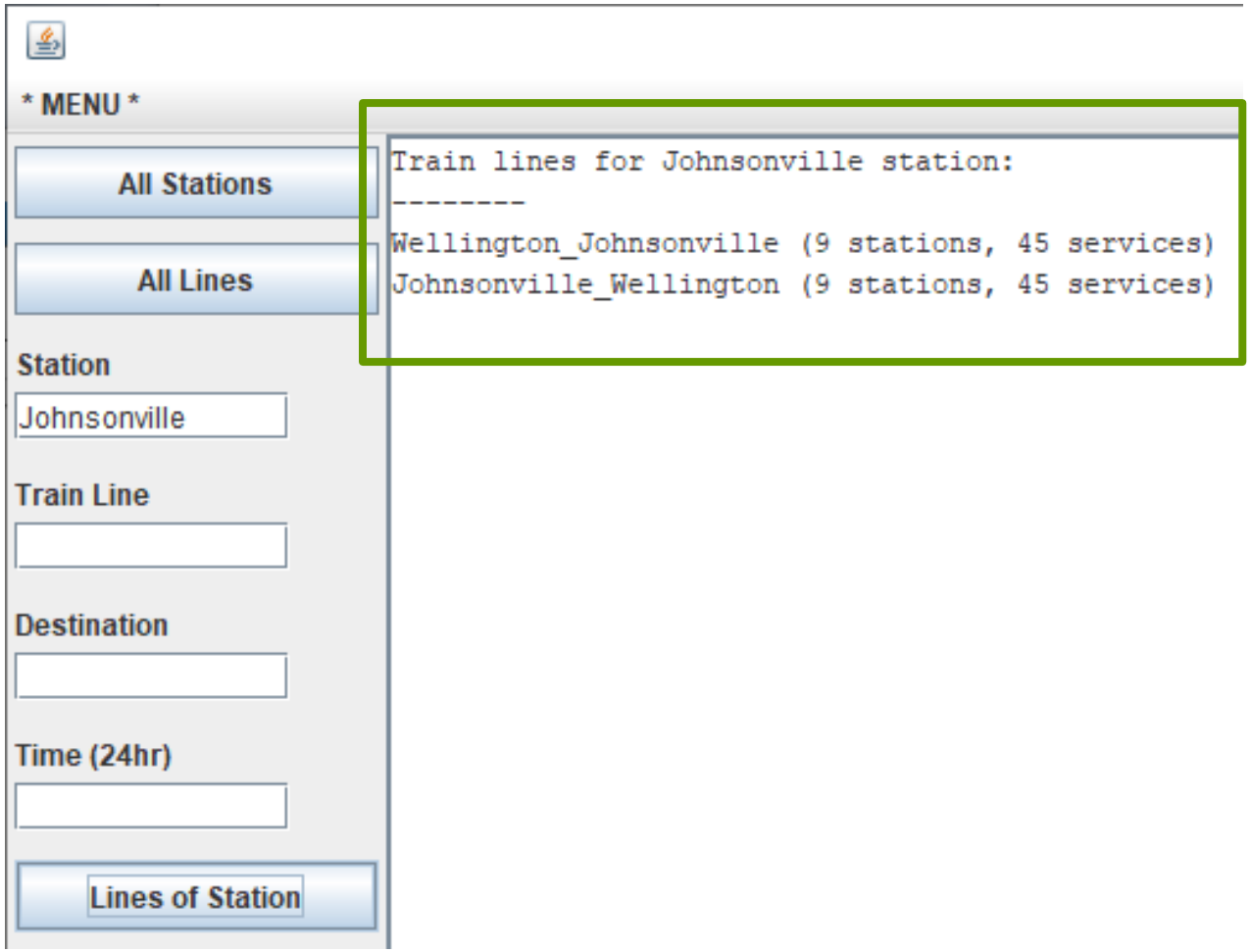
1) Type the following Java code in the listLinesOfStation method

```
Station station = allStations.get(stName);
if (station == null){UI.println("Station "+ stName + " is unknown");return;}
UI.println("Train lines for "+ stName + " station:\n--------");
for (TrainLine line : station.getTrainLines()){
    UI.println(line);
}
```

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`
- List the stations along a given train line by writing a `listStationsOnLine` method that **only** prints each station using `UI.println(station)`



Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

Stations, Lines, and Services:

Your program has to deal with three kinds of information:

- Train Stations
- Train Lines: a sequence of stations. For example, the Wellington_Johnsonville line, which starts at Wellington station, and ends at Johnsonville station, with 7 stations in between. To make things easier, we treat the inbound and outbound as two separate train lines, so the Wellington_Melling line is different from the Melling_Wellington line. (They have the same stations, but the sequence in one is the reverse of the sequence in the other.)
- Train Service: a schedule/timetable for a particular train running on a given train line. For example, the 11:32 train on the Wellington_Johnsonville line that leaves Wellington station at 11:32, leaves Crofton-Downs at 11:40, ... gets in to Johnsonville at 11:55.
  A Train Service is specified by a sequence of times - the time that the train leaves the first station, followed by the times that the train gets to each of the remaining stations on the line. Note that some train services don't stop at every station on the line. If a train doesn't stop at a station, the corresponding time will be -1.

Queries

The basic queries the program should be able to handle are the following:

Hints provided in Lec6c & Lec7a

1. List all the stations in the region.
2. List all the train lines in the region
3. List the train lines that go through a given station
4. List the stations along a given train line
5. Print the name of a train line that goes from a station to a destination station. (The train line must go the correct direction.)
6. (Completion) Find the next train service for each line at a station after the specified time
7. (Completion) Find a trip between two stations (on the same line), after the specified time.
   Find the train line, the time that next service on that line will leave the first station, the time that the service will arrive at the destination station, and the number of fare zones the trip goes through.
8. (Challenge:) More complex trips involve going from the first station to an exchange station on one train line, then going from the exchange station to the second station on another train line. (In larger cities, trips might require more than 2 train lines, but the Wellington region has a central hub, so you never need more than two segments for a trip.)
   Find the best trip (earliest arrival) between a starting station and a destination station. If the trip requires two train lines, print out the starting time, the first train line, the exchange station, the arriving and departing times at the exchange station, the second train line, and the arrival time at the destination, and the total number of fare zones.

Assignment 2a - Core

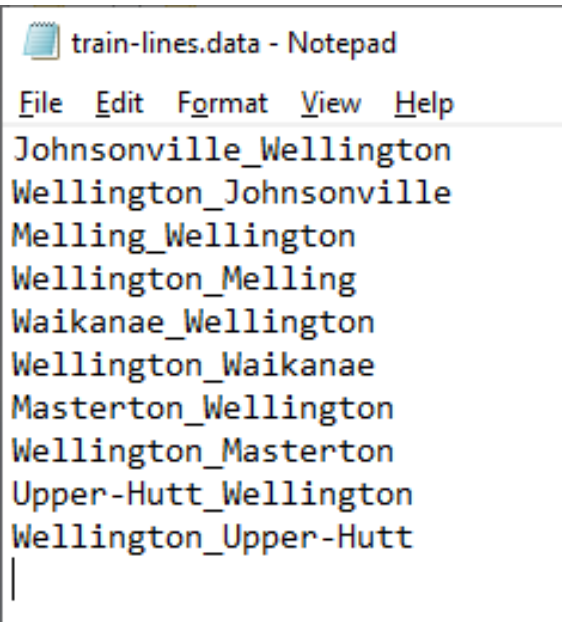
https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`
- List the stations along a given train line by writing a `listStationsOnLine` method that **only** prints each station using `UI.println(station)`

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

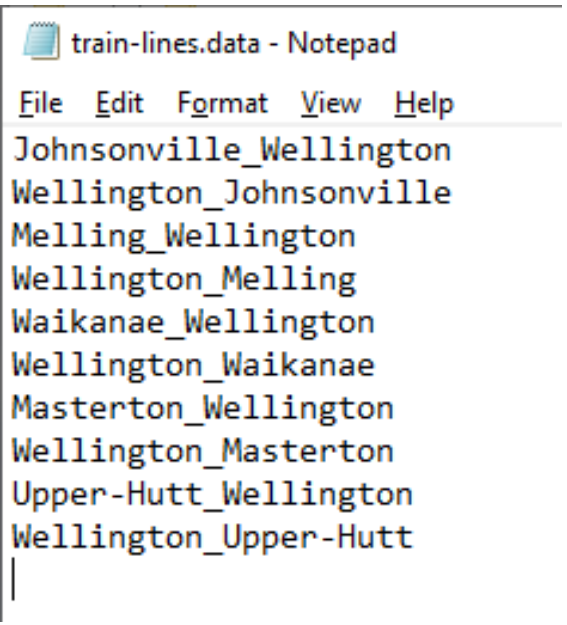
- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`
- List the stations along a given train line by writing a `listStationsOnLine` method that **only** prints each station using `UI.println(station)`

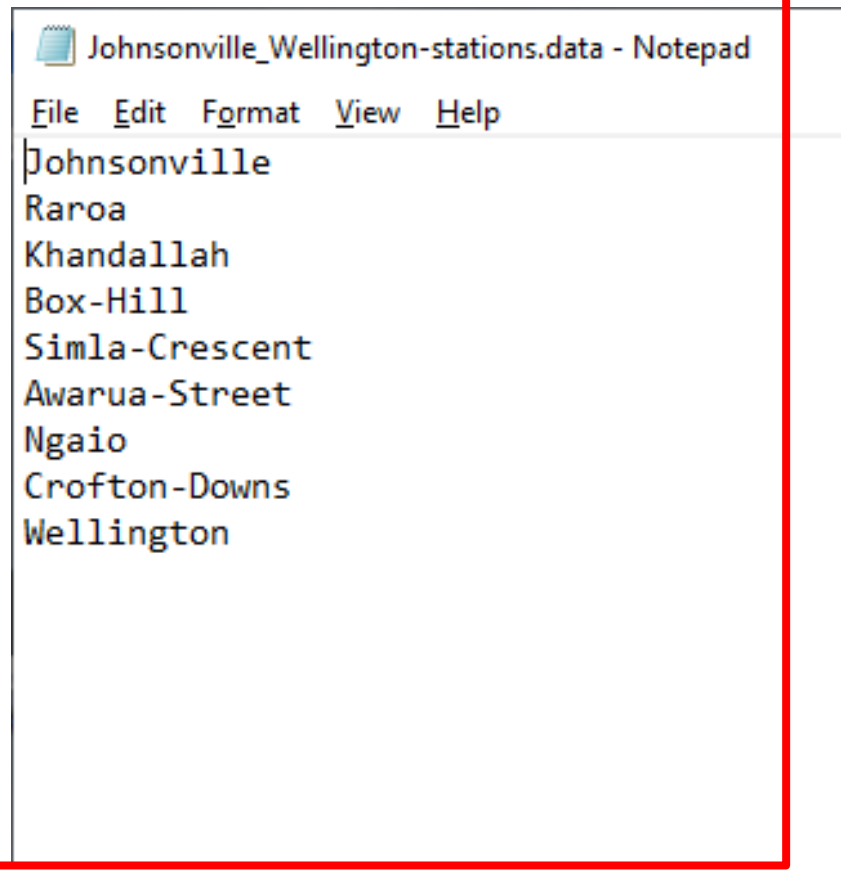


Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`
- List the stations along a given train line by writing a `listStationsOnLine` method that **only** prints each station using `UI.println(station)`





Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`
- List the stations along a given train line by writing a `listStationsOnLine` method that **only** prints each station using `UI.println(station)`

train-lines.data - Notepad

File Edit Format View Help

```
Johnsonville_Wellington
Wellington_Johnsonville
Melling_Wellington
Wellington_Melling
Waikanae_Wellington
Wellington_Waikanae
Masterton_Wellington
Wellington_Masterton
Upper-Hutt_Wellington
Wellington_Upper-Hutt
```

Johnsonville_Wellington-stations.data - Notepad

File Edit Format View Help

```
Johnsonville
Raroa
Khandallah
Box-Hill
Simla-Crescent
Awarua-Street
Ngaio
Crofton-Downs
Wellington
```

stations.data - Notepad

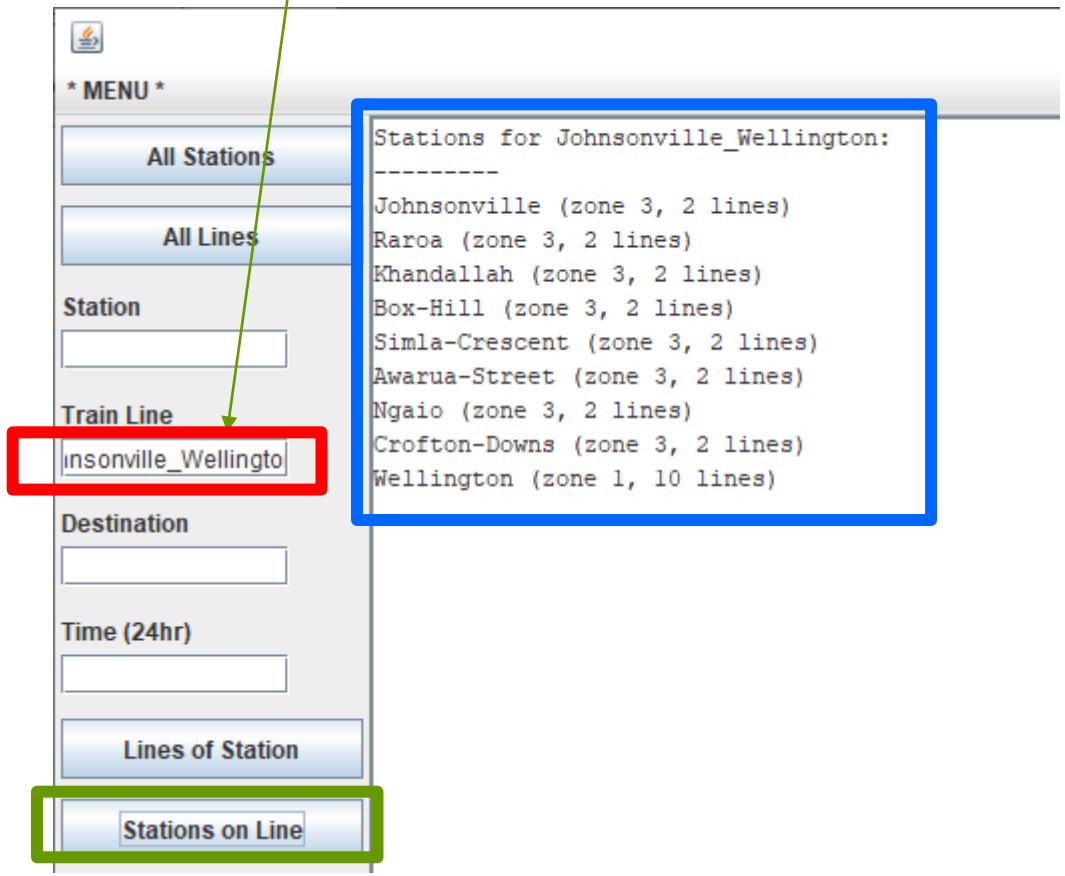
File Edit Format View Help

```
Wellington 1 0
Ngauranga 3 4.8
Crofton-Downs 3 4.9
Ngaio 3 5.2
Awarua-Street 3 6
Simla-Crescent 3 6.9
Box-Hill 3 7.2
Khandallah 3 8
Raroa 3 9.2
Johnsonville 3 10.5
Petone 4 10.5
Takapu-Road 4 11.9
Western-Hutt 4 11.9
Ava 4 12.5
Redwood 4 13.1
Melling 4 13.5
Tawa 4 13.8
Woburn 4 14.4
Linden 4 14.9
Waterloo 4 15.5
Kenepuru 5 16.2
Epuni 5 16.5
Porirua 5 17.7
Naenae 5 18.3
Wingate 5 19.5
Taita 5 20.6
Paremata 6 21.9
Pomare 5 22
Mana 6 23.2
Manor-Park 6 23.7
Plimmerton 6 24.5
Silverstream 6 26.8
Heretaunga 6 28.2
Trentham 6 29.4
Pukerua-Bay 7 30.4
```

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`
- List the stations along a given train line by writing a `listStationsOnLine` method that **only** prints each station using `UI.println(station)`



Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`
- List the stations along a given train line by writing a `listStationsOnLine` method that **only** prints each station using `UI.println(station)`

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`
- List the stations along a given train line by writing a `listStationsOnLine` method that **only** prints each station using `UI.println(station)`

```
public void setupGUI(){
    UI.addButton("All Stations",    this::listAllStations);
    UI.addButton("All Lines",       this::listAllTrainLines);
    UI.addTextField("Station",      (String name) -> {this.stationName=name;});
    UI.addTextField("Train Line",   (String name) -> {this.lineName=name;});
    UI.addTextField("Destination", (String name) -> {this.destinationName=name;});
    UI.addTextField("Time (24hr)", (String time) ->
        {try{this.startTime=Integer.parseInt(time);}catch(Exception e){UI.println("Enter four
    UI.addButton("Lines of Station",    () -> {listLinesOfStation(this.stationName);});
    UI.addButton("Stations on Line",    () -> {listStationsOnLine(this.lineName);});
```

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`

```
public void listAllTrainLines() {
}

public void listLinesOfStation(String stName) {
}
public void listStationsOnLine(String lnName) {
}

public void checkConnected(String stName, String desName) {
}

public void findNextServices(String stName, int stTime) {
}

public void findTrip(String stName, String desName, int stTime) {
}
```

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`

```
public void listAllTrainLines() {
}

public void listLinesOfStation(String stName) {
}
public void listStationsOnLine(String lnName) {
}
```

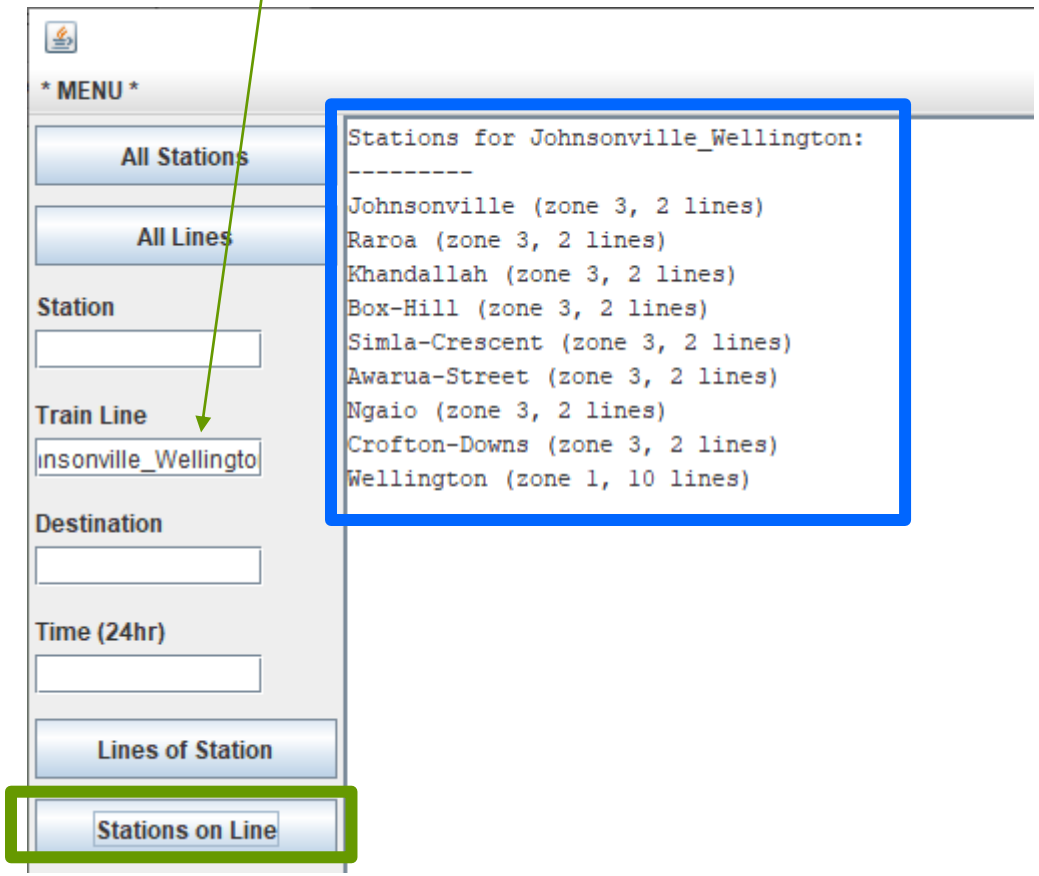
Study the following Java code given for the listLinesOfStation method and type-in similar code as required for the listStationsOnLine method

```
Station station = allStations.get(stName);
if (station == null){UI.println("Station "+ stName + " is unknown");return;}
UI.println("Train lines for "+ stName + " station:\n--------");
for (TrainLine line : station.getTrainLines()){
    UI.println(line);
}
```

Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

- List the train lines that go through a given station by writing a `listLinesOfStation` method that **only** prints each train line using `UI.println(line)`
- List the stations along a given train line by writing a `listStationsOnLine` method that **only** prints each station using `UI.println(station)`



Assignment 2a - Core

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment2PartA

Stations, Lines, and Services:

Your program has to deal with three kinds of information:

- Train Stations
- Train Lines: a sequence of stations. For example, the Wellington_Johnsonville line, which starts at Wellington station, and ends at Johnsonville station, with 7 stations in between. To make things easier, we treat the inbound and outbound as two separate train lines, so the Wellington_Melling line is different from the Melling_Wellington line. (They have the same stations, but the sequence in one is the reverse of the sequence in the other.)
- Train Service: a schedule/timetable for a particular train running on a given train line. For example, the 11:32 train on the Wellington_Johnsonville line that leaves Wellington station at 11:32, leaves Crofton-Downs at 11:40, ... gets in to Johnsonville at 11:55.
  A Train Service is specified by a sequence of times - the time that the train leaves the first station, followed by the times that the train gets to each of the remaining stations on the line. Note that some train services don't stop at every station on the line. If a train doesn't stop at a station, the corresponding time will be -1.

Queries

The basic queries the program should be able to handle are the following:

Hints provided in Lec6c & Lec7a

1. List all the stations in the region.
2. List all the train lines in the region
3. List the train lines that go through a given station
4. List the stations along a given train line
5. Print the name of a train line that goes from a station to a destination station. (The train line must go the correct direction.)
6. (Completion) Find the next train service for each line at a station after the specified time
7. (Completion) Find a trip between two stations (on the same line), after the specified time.
   Find the train line, the time that next service on that line will leave the first station, the time that the service will arrive at the destination station, and the number of fare zones the trip goes through.
8. (Challenge:) More complex trips involve going from the first station to an exchange station on one train line, then going from the exchange station to the second station on another train line. (In larger cities, trips might require more than 2 train lines, but the Wellington region has a central hub, so you never need more than two segments for a trip.)
   Find the best trip (earliest arrival) between a starting station and a destination station. If the trip requires two train lines, print out the starting time, the first train line, the exchange station, the arriving and departing times at the exchange station, the second train line, and the arrival time at the destination, and the total number of fare zones.