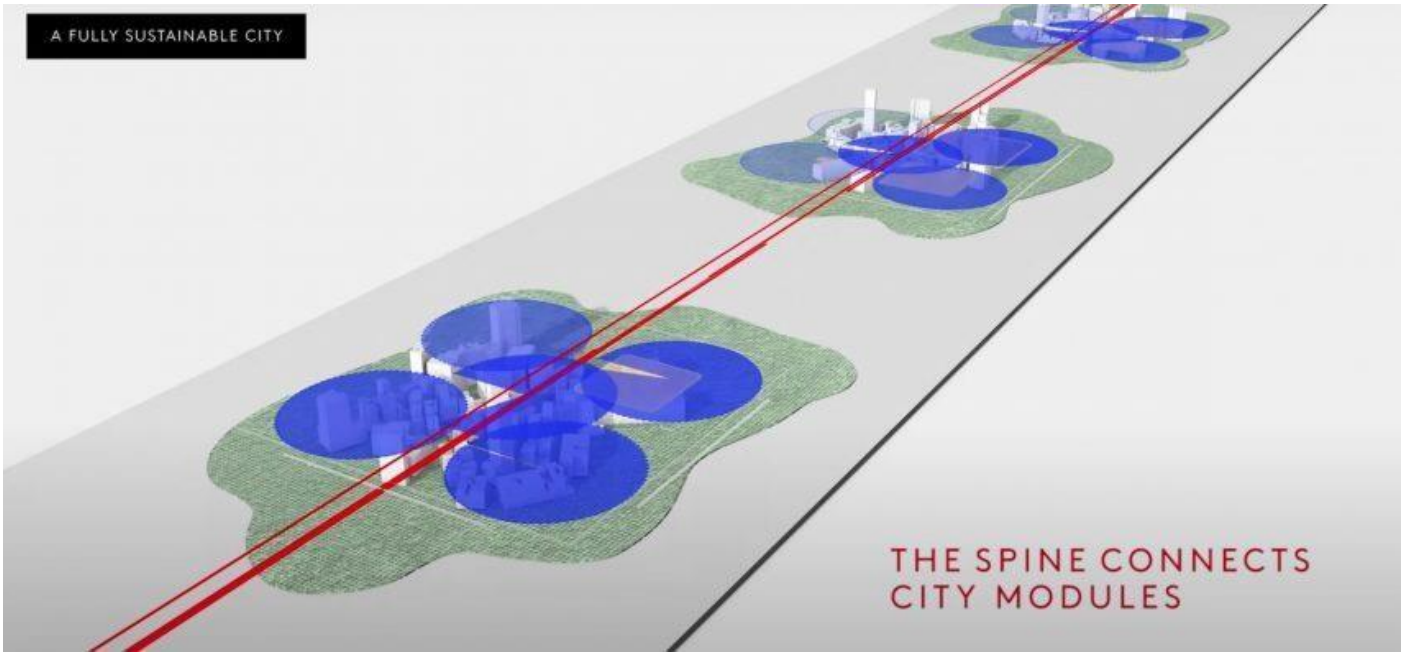
CS210 with Dr. Mohammed Akour **Term Project 1** **Weight 10%** **Due: End of Sunday February 21, 2021**

THE-LINE SPINE SIM TOOL

Neom was announced in 2017 and is part of Saudi Arabia's Vision 2030 drive to diversify its economy and become less reliant on oil. The area will be populated by more robots than people, and powered by solar panels and wind farms. Part of the NEOM project is the Line. The linear city will have no cars or streets, with all residents living within a five-minute walk of essential facilities. The 100-mile-long (170 kilometers) mega-city will consist of connected communities – which it calls "city modules" – and link the Red Sea coast with the north-west of Saudi Arabia. All communities will be connected by a Spine that runs through the 170 kilometers city. Drawings show vehicles driven by artificial intelligence (AI), a metro line and high-speed freight transportation located underground. The metro line may use hyperloop or high-speed underground trains that stops frequently at each module. Passengers can embark and disembark at metro stations. About the Line-NEOM Project: https://www.dezeen.com/2021/01/13/line-saudi-arabia-170-kilometres-long-city-neom/



In this project we will simulate this metro-train, by using Linked Lists as a suitable data-structure to store information about trains, passengers, service timings, cost of journey etc. You project implements three classes namely Node, List and Solution.

The following provides an API for your project.

Node	Description

int ticketID;
int paxID;
int journeyDate;
int srcModule;
int destModule;
String paxName;
Node next;
	// a unique ticket id;
	// Passenger ID number;
	// journey date DDMMYYYY;
	// takes values between 1 and 255
	// takes values between 1 and 255
	// Name of the passenger
	// Node next

Node()
Node(, , ,)
 	
String toString()
	// default constructor
	//override constructor as necessary. You can edit the params as needed.
	//returns a String with the contents of the Node as follows:
	ticketID, paxID, journeyDate, srcModule, destModule, paxName.

The following is the API for the List class. This implements a Singly Linked List consisting of Nodes from the Node class.

List	Description
Node Head; int size;	// Anchors the Head of the List // Maintains the size of the List

List();
 	
insert(Node N)
 	
void remove(PaxID)
 	
int cost(PaxID)
 	
void print1(PaxID)
void print2(journeyDate)
 	
void print3(srcModule)
 	
void print3(destModule)
	// default constructor
	
	// inserts the node N in the correct position, ordered by the date. e.g. date1 is 14022021; date2 is 01032021. date1 appears before date2. Assume all months have exactly 30 days. There are 12 months in a year. If dates are equal (we assume inserting at the beginning).
	
	//removes the first node with matching PaxID.
	
	//Computes the cost of ALL journeys for a passenger with PaxID.
	Here is how this is computed.
	
	Journey cost = Math.abs(srcModule – destModule) * 5.
	//example srcModule=3, destModule=5. Journey Cost = 10.
	
	//searches for PaxID in the list. Prints ALL nodes with PaxID.
	//searches for journeyDate in the list. Prints ALL nodes with journeyDate.
	//searches for srcModule in the list. Prints ALL nodes with srcModule.
	//searches for destModule in the list. Prints ALL nodes with destModule.
	
	NOTE: print methods call Node.toString() as necessary.

The following is the API for the Solution class. This class implements the main method, makes appropriate data Input/Output calls and implements data structures and all necessary method calls.

Solution	Description
//Attributes NA	Not applicable
main()	Implements the main method; reads data from console, processes information and make appropriate calls as necessary.

Sample Input

Here is a sample input

```
1 1225 101 10022021 3 5 Ahmed Baloshi

1 1226 101 11022021 5 8 Ahmed Baloshi

1 1227 101 12022021 8 3 Ahmed Baloshi

3 101

2 101

4 101
```

Sample Output

For the above sample input, the following would be printed on the console.

```
50

1226 101 11022021 5 8 Ahmed Baloshi

1227 101 12022021 8 3 Ahmed Baloshi
```

Explanation

```
1 1225 101 10022021 3 5 Ahmed Baloshi

1 1226 101 11022021 5 8 Ahmed Baloshi

1 1227 101 12022021 8 3 Ahmed Baloshi
```

```
The line starting with integer 1 indicates that insert would be called. No Output is
generated for the insert call.
```

```
3 101
```

```
The line starting with integer 3 indicates that cost method would be called. The program
computes the cost for all journeys taken by Passenger with paxID = 101. In the list, there
are 3 nodes with paxID = 101.

Node with ticketID=1225 has a journey cost = Math.abs(3-5) * 5 = 10
Node with ticketID=1226 has a journey cost = Math.abs(5-8) * 5 = 15
Node with ticketID=1227 has a journey cost = Math.abs(8-3) * 5 = 25

Hence total cost = 10+15+25=50
50 will be displayed on the console
```

```
2 101
```

```
The line starting with integer 2 indicates that the first node with paxID=101 will be
removed from the list. No output is generated. This removes the node that contains the
following information

1 1225 101 10022021 3 5 Ahmed Baloshi
```

```
4 101
```

```
The line starting with integer 4 indicates that All passenger details for PaxID=101 will be
printed on console separate by new line character at the end of each line.

1226 101 11022021 5 8 Ahmed Baloshi

1227 101 12022021 8 3 Ahmed Baloshi
```

Similar to the above, your program implements the following commands:

```
Legend for the acceptable commands:

1. insert details for a new node

2. removes node from list

3. computes the Cost of ALL journeys for this PaxID and displays the total.

4. prints All nodes with PaxID.

5. prints ALL nodes with journeyDate.

6. prints ALL nodes with srcModule.

7. prints ALL nodes with destModule.
```

Evaluation:

You are allowed to work as a group with maximum 2 members in a group . Your work's evaluation would be based on Code inspection and successful execution of *k* number of test cases. The instructor reserves the right to determine the scores of each test case.

Test-cases will be posted on hackerrank, students will have unlimited number of opportunities to post and test their project until the due date. The system will not take any submissions after the due date.

Code Inspection:

The code would be inspected by the instructor. The instructor would determine the score to be given for code inspection. Generally, a readable code (indentation, clear scope definition) is required. For this project, there are no limitations on time and memory usage. Hackerrank checks for plagiarism. If the similarity of your submission is more than 50%, you will be awarded a ZERO in the project.

Submission:

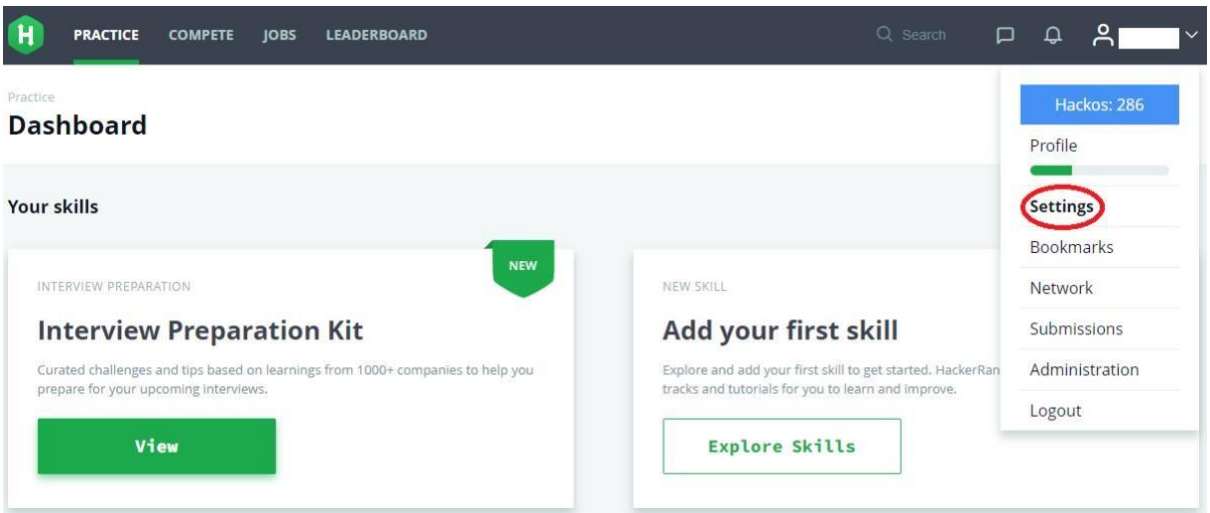
Submission on Hacker Rank

Step 1.

Register on https://www.hackerrank.com/

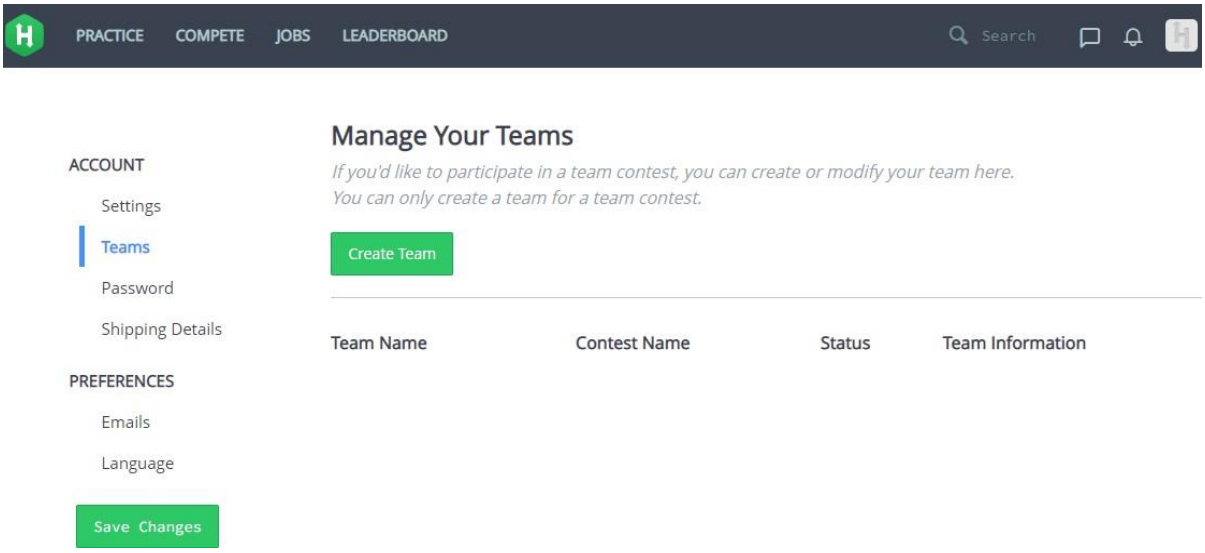
Step 2.

Go to settings



Step 3.

Create a Team. For this project you may work alone (1 person per team) or up to 2 persons per team.



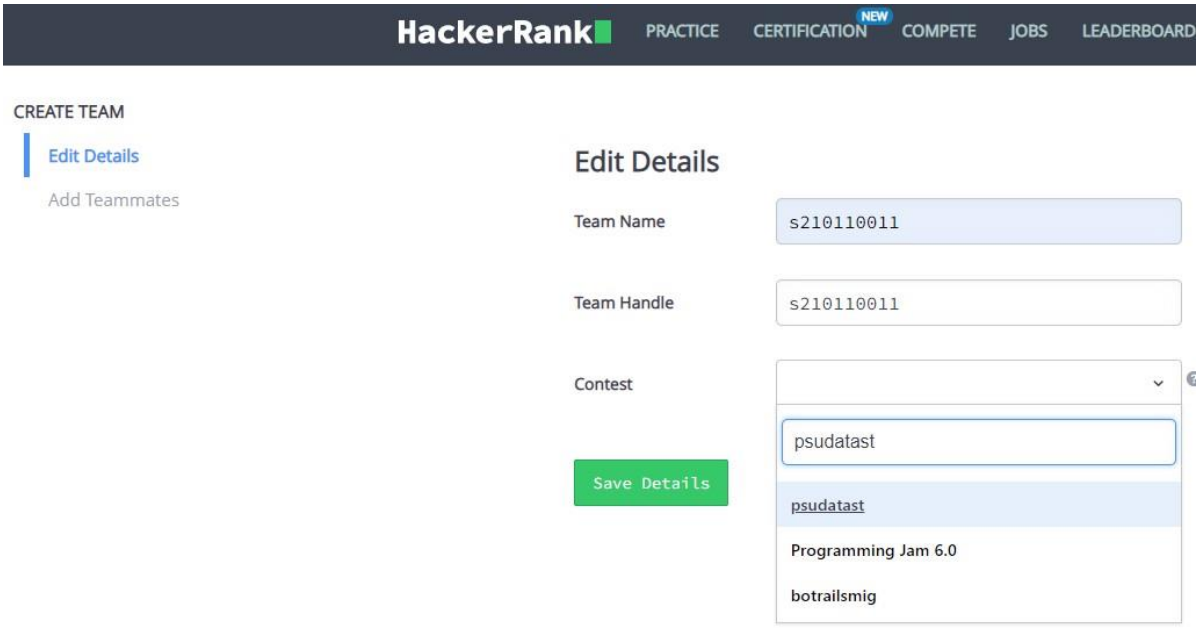
Step 4.

Edit Team details.

Important NOTE:

1. Team name must of "S" your Student ID. If it is 2 students per team, S210110123S210110124

2. Join contest "psudatast"



Step 5:

Go to https://www.hackerrank.com/psudatast

Start working on your project.

Submission Dead-Line:

The submission deadline is final. Late Submissions will be awarded ZERO points.

Plagiarism:

Be warned, all code submitted would be compared using hackerrank. Any similar code would result in ZERO earned for both group members as well as all other groups with similar code. NO EXCEPTIONS!

Important Notes:

- It is the student's responsibility to check/test/verify/debug the code before submission.
- It is the student's responsibility to check/test/verify all submitted work (including jar files)
- It is the student's responsibility to verify that all files have been uploaded to the LMS.
- For each project, instructor will provide 1-2 sample test-cases to verify the execution of your program.
- After an assignment/project has been graded, re-submission with an intention to improve an assignments scores will not be allowed.
- After the assignment/project has been graded, the instructor will post test-cases used for grading on the website.
- The Instructor has the right to share project execution reports that may have been auto-generated on the course website.