**National University of Sciences and Technology**

**NUST Balochistan Campus (NBC)**

**Computer Organization and Assembly Language**

**Semester Project Report**

**“Travelling Salesman Problem Implementation in Assembly”**

**Instructors:**

**Mr. Najeebullah**

**Engr. Mr. Abid Hussain**

**Submitted by:**

**Muhammad Osama (356270)**

**Zainab Zahid (343162)**

**Amna Akram (344396)**

**Submission Date: 14th March 2022**

**BS(CS) 3rd Semester**

**Travelling Salesman Problem:**

The travelling salesman problem (TSP) asks a very fundamental question: Given a

list of cities and the distances between each pair of cities, what is the shortest

possible route that visits each city exactly once and returns to the origin city?

**Narrowed Down Scope for Our Implementation in Assembly:**

We hope to implement TSP with respect to our university’s van(s) in terms of how

many times they can travel to a faculty’s place of stay (home/mess) in a shortest

possible route whilst visiting each place of stay exactly once and returns to the

place of origin (which in our case would either be the university itself or the place

wherever the van(s) is(are) parked).

**Our Goals/Objectives of This Project:**

● Understanding the proposed problem and coming up with an algorithm to

best solve it.

● Understanding and implementing it first using video memory in DosBox

followed by:

● Using graphics in DosBox to simulate the path taken by the van(s) using

Minimum Spanning Tree(MST).

● Ultimately implement the solution of this proposed problem using either

Kruskal’s Algorithm or Prim’s Algorithm.

● A presentation signifying the importance of how this problem is salient in the

field of Networking with respect to its implementation using MST, which is a

concept of Trees data structure.

**Motivation Behind This Project:**

The main reason behind picking this project is to better understand graph theory

and how it is understood by the computer on machine level. This will also be of

great help to our concepts of networking which we will eventually study in the

coming semesters. This will help lay a better foundation for it. Perhaps the biggest

motivation behind picking this project is to pass this course.

**Approach to Solve TSP in Assembly:**

We are giving the distance between cities in an array as the input parameter and we have recreated the TSP accordingly where our distance = cost of the TSP. Our problem is that we are trying to give a shortest route for the van that is operating for NBC. It obviously picks up different people in the staff and gets them to University.

Given this, we can run with the assumption that the van is already parked at the University, thus its starting and ending location is the University itself.

And that is the crux of the TSP in our implementation.

For our approach, we have identified different locations as different nodes and when we reach a certain node, it is marked as reached in the AFD by moving 1 to ax.

Now, the van will begin at the starting point and end at the same point whilst calculating the shortest route between the next two locations (nodes) that follow the starting point and at the end it will give us the starting point again because the van has to return to the University.

**The Algorithm:**

For solving this problem (or rather implementing it) in assembly, we’ve used two subroutines (functions) to keep the time complexity as low as possible, and it already helps our case that we’re working with registers, the fastest memory possible.

Thus, the time complexity of our algorithm is O(n) as it computes the result in linear time.

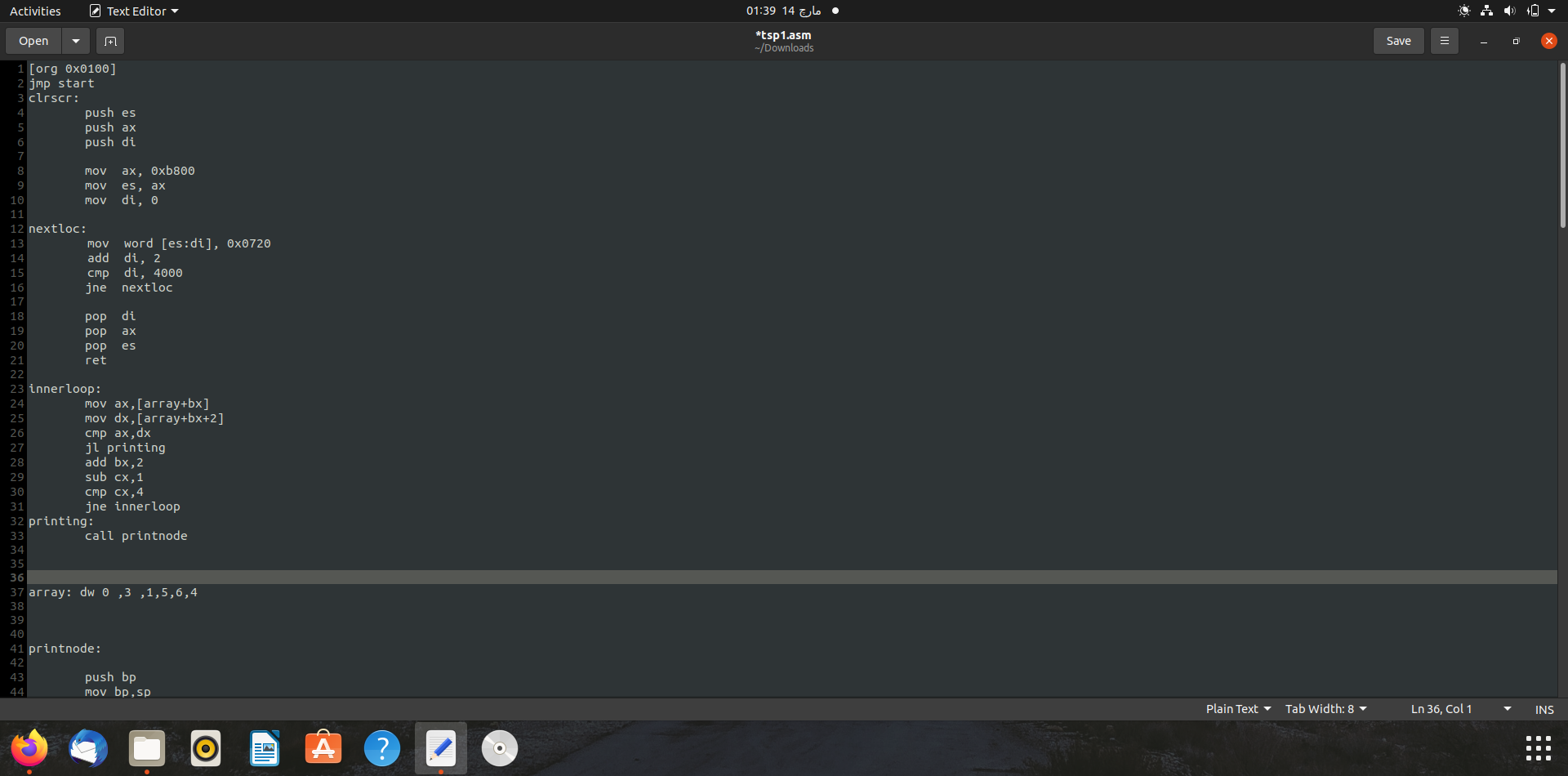
**The Actual Logic:**

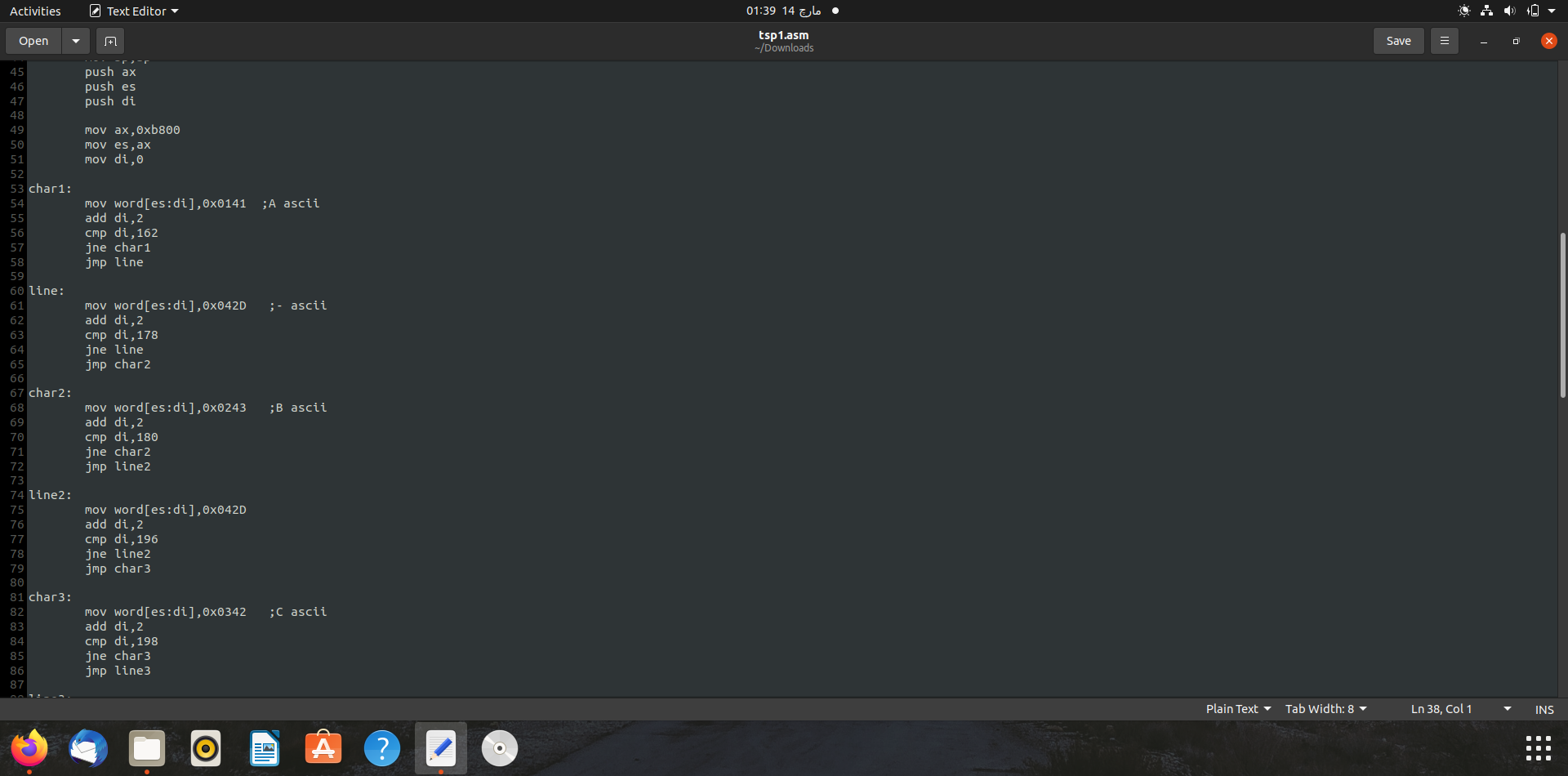
We started off by assigning a starting node to the values in our array, and then traversed the array by making comparisons with each of the “paths” or “routes” that seemed possible from each node. This way we were able to achieve all the routes from a node.

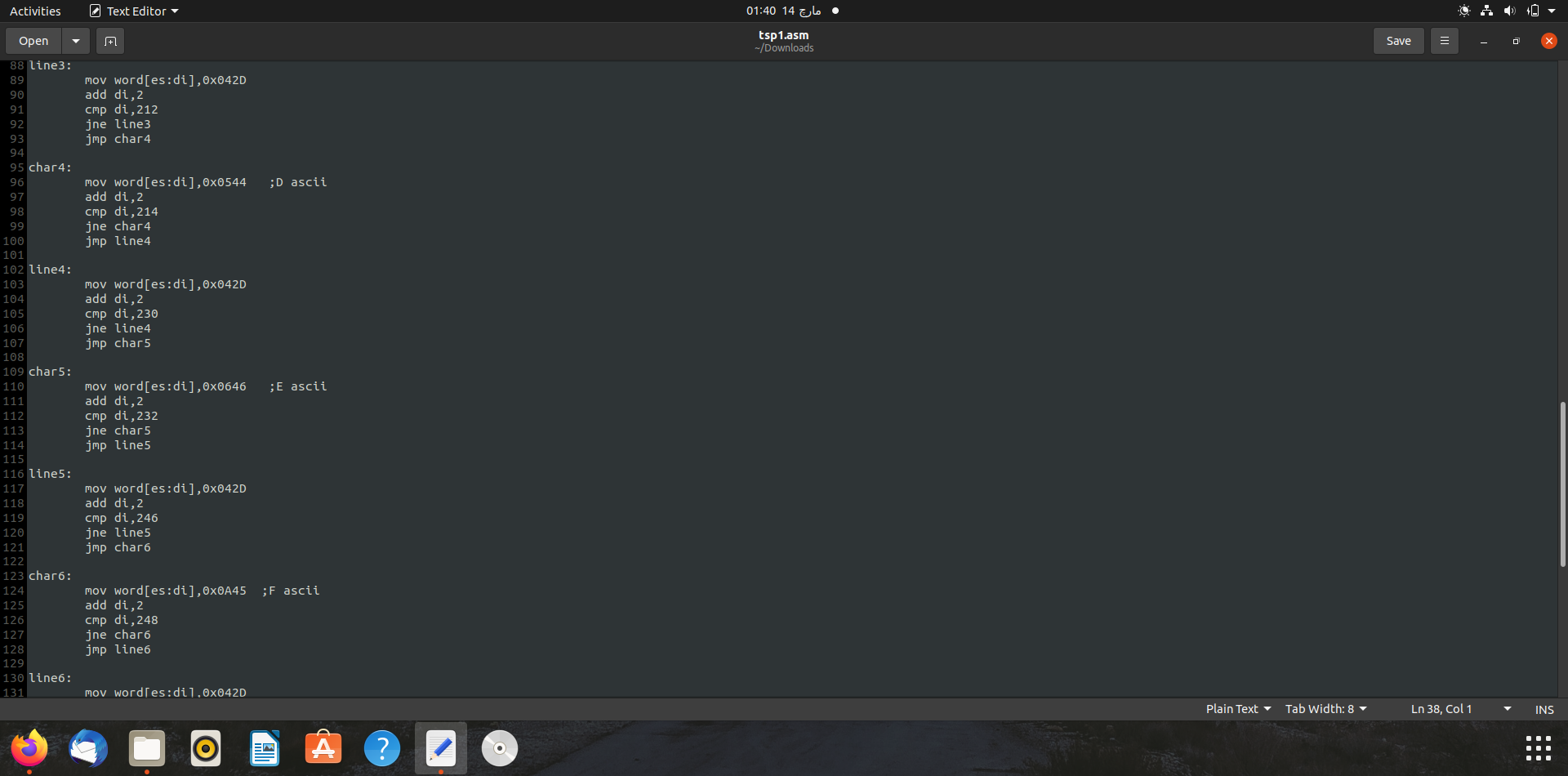
We compared the supposed distance of the nodes and if the distance of let’s say node B was 6 from the root node A and the distance of C was 5 from A, we will show a path from A to C instead of B.

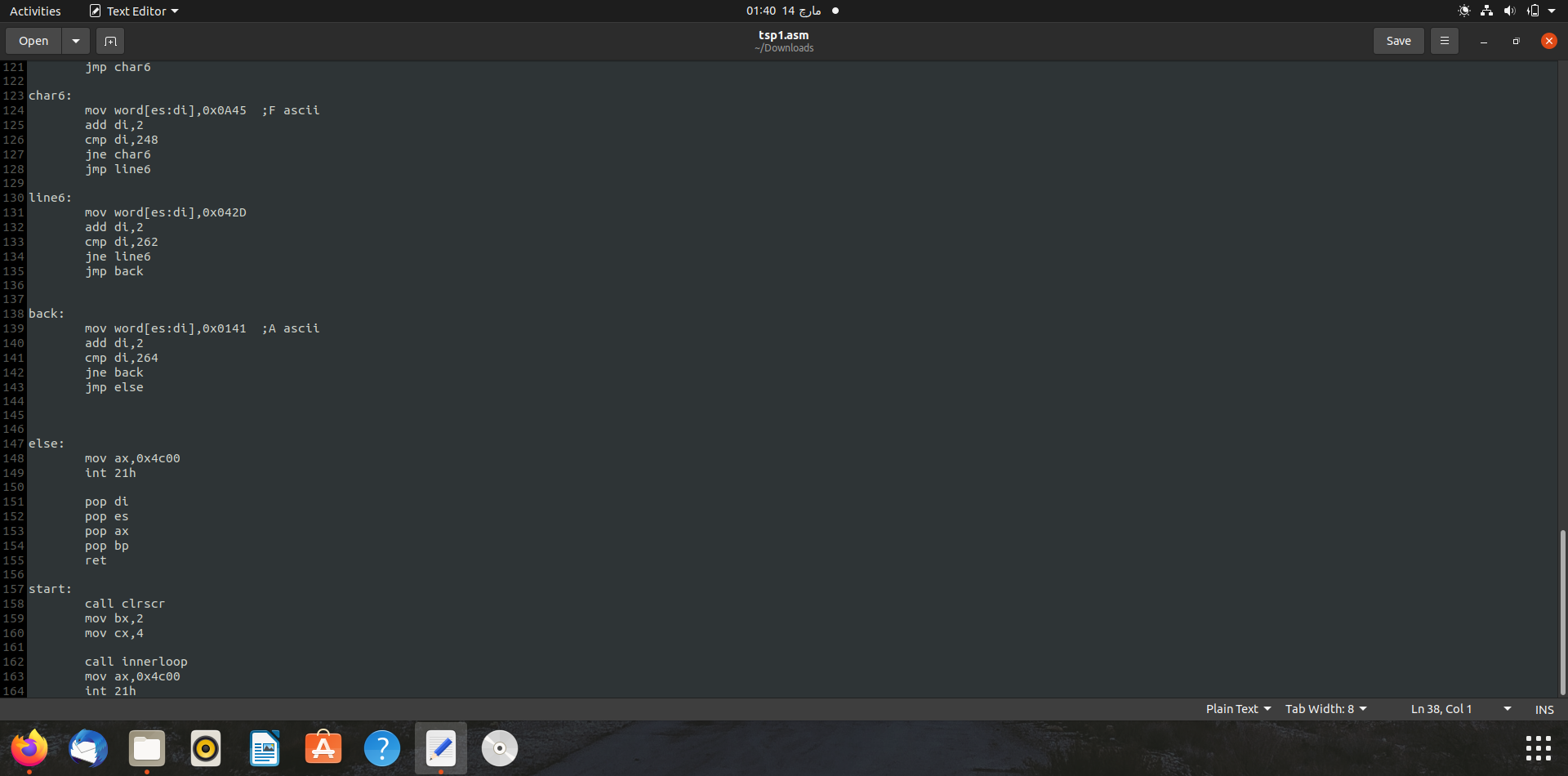
We have done this in order to obtain the least costly route i.e., the shortest possible route.

**Code:**









**Output:**

