CPSC 535 Advanced Algorithms

Project 2: It's a Small World

Instructor: Prof. Doina Bein

Submission date: 9th December 2022

Priya Keshri
[885191452]
priyakeshri@csu.fullerton.edu

Shrinivas Pravin Patil
[885212043]
pshrinivas264@csu.fullerton.edu

Summary:

Python3 version 3.10 is used to implement the project. Install the Python compiler and editor as 'PyCharm'. The code takes input from a text file named input.txt. First line of the text file represents the total number of test cases. From the next line each line contains a number of casts in each test case followed by casts on each line. The expected output is the shortest connection between cast[0] and cast[1] and common actor or cast.

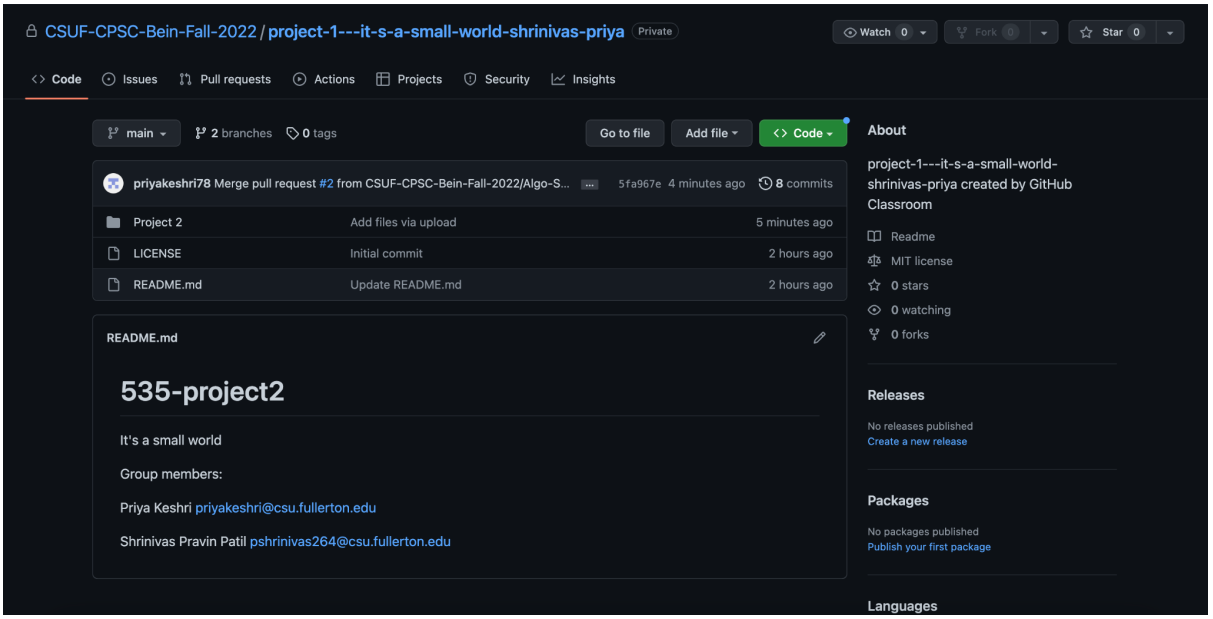
Pseudocode:

1) Read all the test cases from the text file.
2) Create list of casts named cast=[[]]*n
3) Store each cast of actors by preprocessing each line by splitting and removing new line and spaces.
4) if cast[0] & cast[1]:
        conn=1
        actor=list(set(cast[0] & cast[1]))
        return "shortest connection={},actor/actors={}".format(conn,cast[i])
   For i goes from 2 to n:
        If set(cast[i]) & set(cast[0]):
            If set(cast[i]) & set(cast[1]):
                conn=2
                Return "Shortest connection = {}, conn={}".format(conn,cast[i])
   Return "shortest connection > 2 or no connection"

Code:

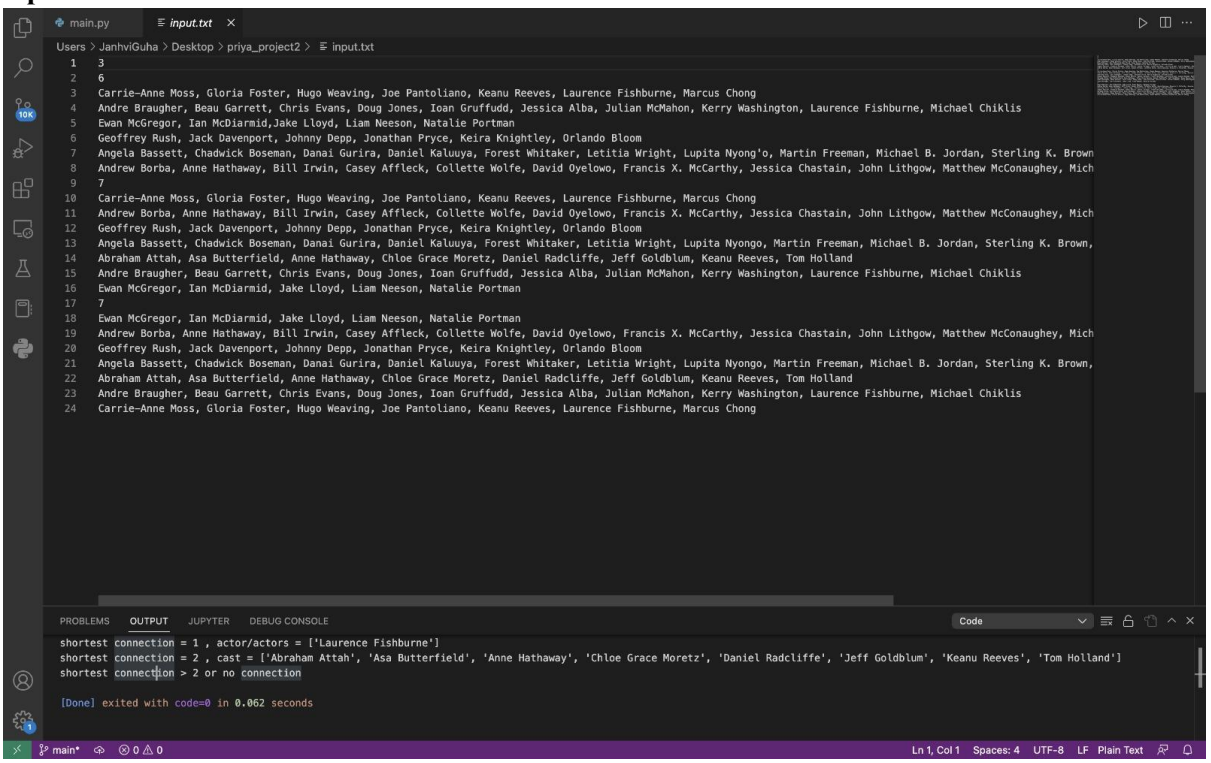
```
import os
import sys

def find_connect(n,cast):
   conn = 0
   try:
      if set(cast[0])&set(cast[1]):       # Convert the list into set and find common actor using
intersection
         conn = 1
         actor = list(set(cast[0])&set(cast[1]))
         return "shortest connection = {} , actor/actors = {}".format(conn,actor)
      for i in range(2,n):               # Check for cast 2 to cast n for shortest connection
         if set(cast[i])&set(cast[0]):
            if set(cast[i])&set(cast[1]):
               conn = 2
               return "shortest connection = {} , cast = {}".format(conn,cast[i])
      return "shortest connection > 2 or no connection"
   except:
      return "Please try it again!!"

if __name__ == '__main__':
   with open('input.txt','r') as fl:  # Reading Text file
      cases = int(fl.readline())      # Reading the first line which number of test cases
```

```
        while cases:
            n = int(fl.readline())      # Reading the length of list
            cast = [[]]*n               # Creating list of casts
            for i in range(n):
                cast[i] = list(map(str.strip,fl.readline().replace('\n','').split(','))) # Processing each line and
store in the list.
            #print(cast)
            print(find_connect(n,cast)) # Calling the function
            cases-=1
```

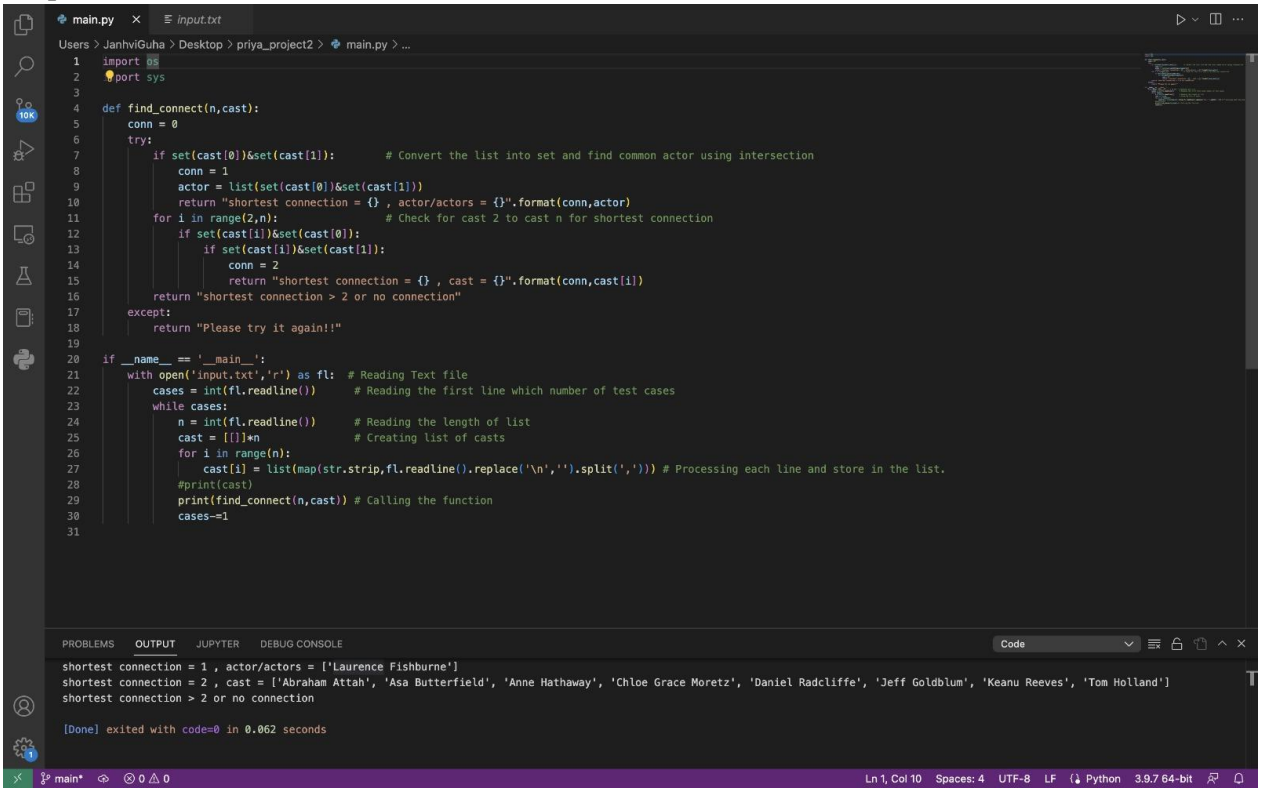


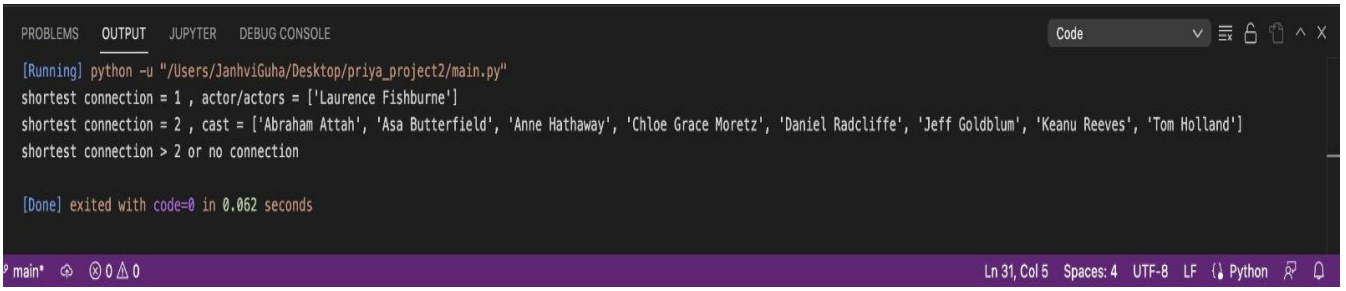
Test Case:

Input:



Output:





Time Complexity: O(n)

Reason: In this algorithm, whenever we call the find_connect(), the while loop will run for n-2 times in the worst case where n is the number of cities.

Space Complexity: O(n*m)

Reason: In this algorithm worst case space complexity will be n*m where n= total number of casts and m= Total number of actors in each cast.