CS302-Design and Analysis of Algorithms

FALL 2020

Project Report

Tooba Nazim 18K-1185

Rethek Kumar 18K-0242

Abstract

The purpose of this project is to use the array of different algorithms studied throughout our course of "Design and Analysis of Algorithms" and apply these concepts and their respective methods to create a simple yet useful interface for users.

Introduction

We have implemented all ten algorithms, namely:

a. Longest Common Subsequence

b. Shortest Common Supersequence

c. Levenshtein Distance (edit-distance)

d. Longest Increasing Subsequence

e. Matrix Chain Multiplication

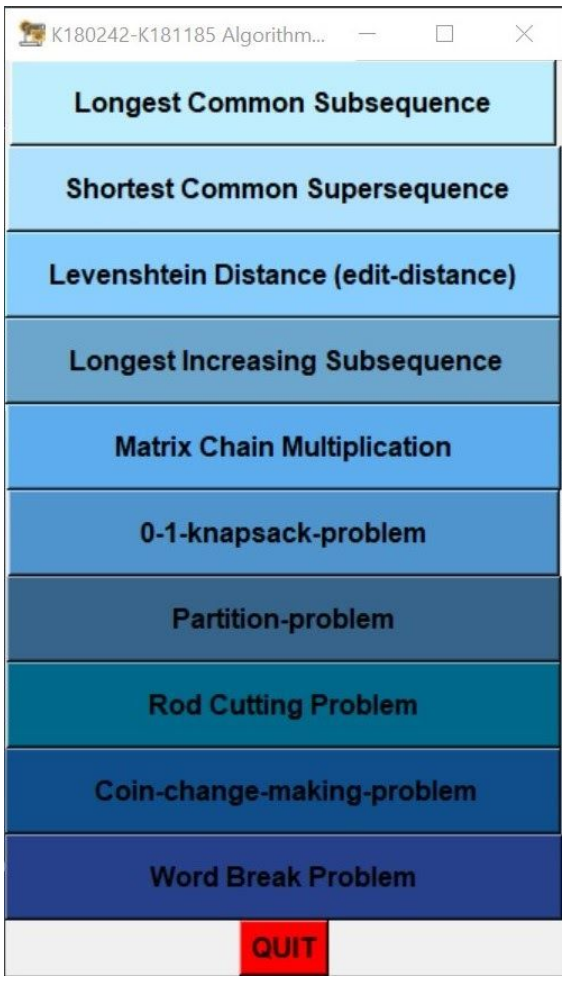
f. 0-1-knapsack-problem

g. Partition-problem

h. Rod Cutting Problem

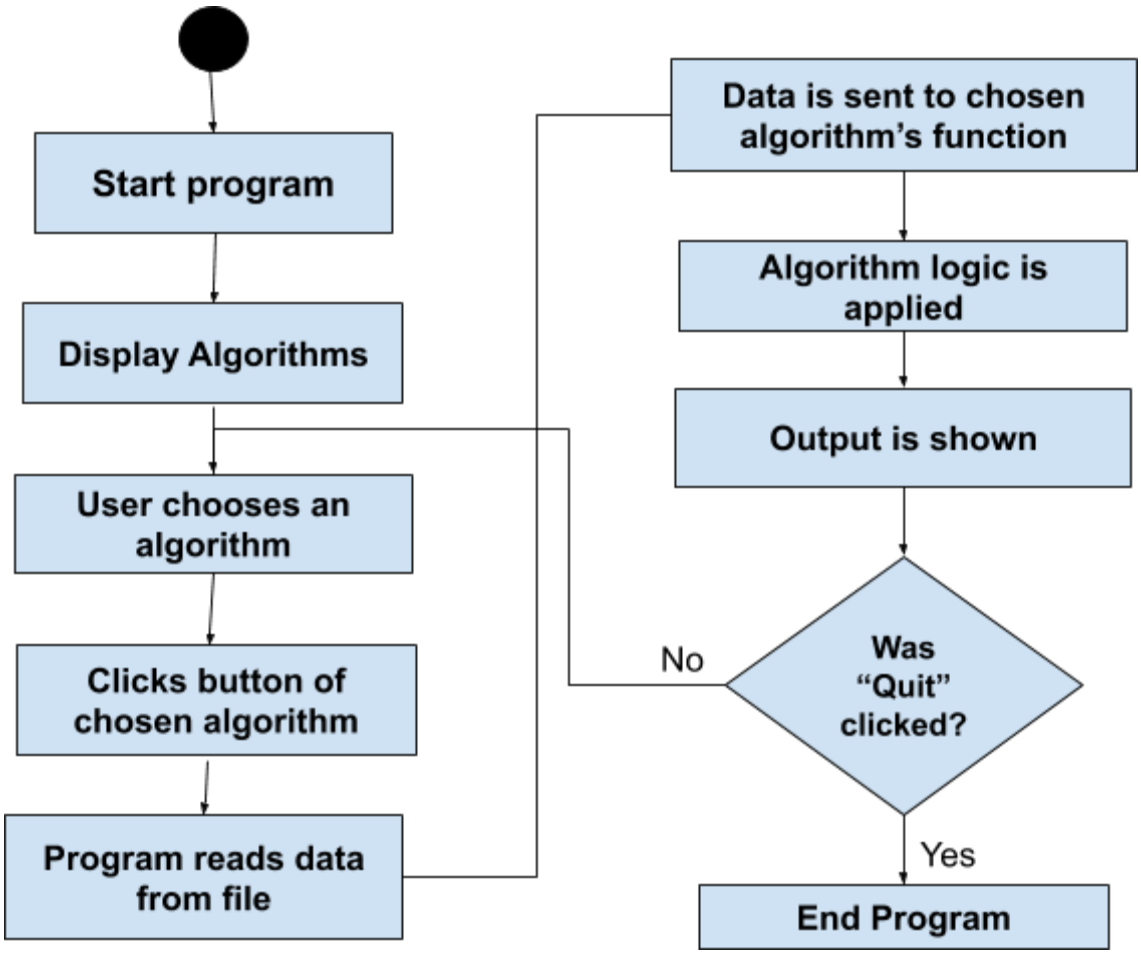
i. Coin-change-making-problem

j. Word Break Problem



Using Python as our coding language as well as Tkinter to implement our interface, we have created a well-functioning and useful system. It allows the user to choose any algorithm to implement, the user can choose more than one as well. The program then read the data from a single text file which includes sample inputs for all 10 Algorithms. The data is then manipulated according to the chosen method and the results are displayed. The colorful Tkinter interface includes, along with the ten available algorithms, a quit button. When the user is finished choosing as many options as they want, they can "quit" to finish the program.

Your proposed system



To discuss our system briefly, Our system starts off by displaying all the available Algorithms that the user can then choose from. Once the user clicks the button, the respective data is then read from the text file i.e algoproj.txt. The data is then sent to it's appropriate function to apply the algorithm logic on the data. The output is shown to the user. The user is allowed to continue clicking buttons and the process will be repeated until the quit button is selected.

Experimental Setup

The contents of our single file are shown below:

A

TOOOBAANAZZZIMMM

TOOOBAAANAZZZZZZIM

B

TOOBBAAAAAAANN

TTTTOIIIMMMMM

C

REEEETTTTHEKKKKUUUUMMMMMAAAAAAAAAAAR

RRETHHHEEEKKKKUMMMMMAAAARRRRRRRRR

D

14,11,29,30,2,17,19,27,24,22,20,60,12,17,9

E

14,11,29,30,2,17,19,27,24,22,20,60,12,17,9

F

30,33,69,44,32,11,16,13

10,20,30,70,80,90,55,100

242

G

35,33,79,54,40,71,10,60,56,4,12,26,14,25,5,44,8,12,41,9,3,34,59,34,9,19,12,5,6,15,8,7,16,9,55,1

4,5,26,9

//rod length 185

H

1,2,3,4,6,7,8,9,10,11,12,13,14,15,16,17,18,19

//rod length 242

I

72,50,8,20,20,32,2,44,58,15,92,90,8,34,51,89,3,100,30,50,47,5,28,30,4,61,10,26

J

hello,ret,tre,re,r,e,t,h,e,k,ku,m,a,r,kum,ar,mar

rethekkumar

Results and Discussion

All of the code is working properly and implementing the algorithms to give accurate solutions. It is to be noted that option A and B is taking longer than others, due to the concept of recursion being followed. User's are also allowed the opportunity to choose as many algorithms as they want as well as the option to choose one repeatedly, Our system will continue giving correct solutions despite repeated selection until the program is quitted.

The outputs of selecting all options is attached in the Zip-File: (A-J)

Conclusion

As an overall conclusion to wrap up this report, we are of the opinion that our system works very well and implements all algorithms in terms of accuracy. However, timing constraints could be improved to make the system even more efficient.

References

1. Logo for Tkinter GUI taken from: **Flaticon**

www.flaticon.com

2. Color palette for the buttons referenced from:

http://www.science.smith.edu/dftwiki/index.php/Color_Charts_for_TKinter

3. Help for Python filing taken from: w3schools

https://www.w3schools.com/python/python_file_handling.asp

4. Diagram drawn using Google Drawing feature

5. Random numbers generated using:

https://stattrek.com/statistics/random-number-generator.aspx

6. For understanding binding of functions with buttons in Tkinter:

https://www.youtube.com/watch?v=qWnE-yp6wzU&ab_channel=thenewboston

7. For Error handling, StackOverflow was used