Task 2 Using State Space Search for Problem Solving

AI-lab 2

Student 1: Joachim Johnson
Student 2: Jonas Åsander
Course: Artificial Intelligence (**DT112G**)
franziska.klugl@oru.se.

Introduction:
The task in this exercise was to write a program that's find and compare restaurants.

There is a few feturers ask after like the comparison between an loaded restaurant in the program and an input som an user that want an good restaurant to eat and want an recommendation that the program should provide. The program should spit out an similarity value depends on how simulare the restaurant and present it in a readable way to a normal user.

The program should provide recommendation and to do this do the program need to have an database with restaurant. The exercise askt after to have it in a file and needed to load this file and even to be able to save to this file.

Recommended number of restaurants to test with is 20 restaurant that should be handed as cases in the system to compare the restaurants in the system to an input resturant case.

to clarify it should is “case” an container for the information about the resturant. information as the name itself priceclass quality nationality etc…..

Language is python.

Data structure design

The program have a few features there load save files add restaurants and delete restaurant and compare them.

Its is done through that it save it in a class in the program when it runs and have the permanent database in a file.

We did declarations to load and save over this file. when we save, we delete the old database and saves over. The database is automatically loaded in the program before the user is capable to save it over.

Generally you can say everything is centered around the class due to cases should be used to compare restaurants to each other and each attribute of the restaurant to take out the similarity value that's askt after.

Overall processes description:

Generally is the comparison function thats are ask after. All features are reachable from an main menu.

Why we made a meny is that it are easier to divide up the functions of the program codewise and for the user of the program.

When we are on the easy to use did we make it easy to add restaurants and delete specific returant from the program and replace the database with whats loaded in the program. this where ask after in the question with added delete function if one case got corrupt put in with wrong input. The question did **not** ask after an modifying function and an delete function where demet as more useful.

We made it so you could use the program to save everything stored in the program to the file and the program could only load in 1 time so it is impossible/very hard to save doublets of the restaurants and minimising the risk of misplaced headers.

Overall did we have more functions that were “needed” for complete the task. why we did include them where that we going to use this for an base for task 3 and we felt the functions where needed to prepare for next exercice and use the program more easily for user and manage stored data/data structure.

Results and Discussion:

That data structure we ended up with could of course be discussed if it were the most in all situations and program in “real life” in the way we divided it. There is more than one way to use it and for more than one way to integrate in an system thats are usually done.

How the program interface is another discussion that could be discussed. We made it quite simple and easy to use and show how it works and what functions it offer. Then if it should be in the real world and should use “in a form understandable for a user” should you probably build it differently. Build it for the kind of user that uses it.

The resulting program is a working program that can compare cases that are put into it that's also easily movable and to build upon for upcoming tasks.

We hope that we build the program in a good way to evolve this further in task 3. you could say we learn/get more experience with evolving code further on and how to build an good base(codevise) to build from. This experience going to be used in the future when we start to work in the real world.