Intro to Dear AI
Spring 2019
Due: Sunday, March 17th, 11:59 pm

In Short
Write a program that receives an order of 4 pancakes and gives the best solution that DFS, UCS, Greedy and A* search find to go from the start state to the goal (ordered pancakes).

Description
In this programming homework, you will be working on the classic pancake problem that was discussed in the class. You need to write a program that receives an arbitrary order of 4 pancakes from the user plus the type of the search algorithm, and prints the steps that the specified algorithm needs to take to reach to the goal state.

Format
Similar to the format described in class, each of the pancakes has an ID number that is consistent with their size. This means that the largest pancake has an ID of 4, next largest 3, next 2, and the smallest has an ID of 1. For instance, Fig. 1 shows the IDs associated with each pancake on the left in a certain configuration.

Figure 1 – Pancakes' representation

Input
The input should consist of four-digit and one character (####X), where the first digit indicates the ID of the bottom pancake, second indicates the second lowest, etc. Figure 1 shows the four-digit representation of the left pancake-set on the right. The last character would be any of "d", "u", "g" and "a" characters, which refer to the Depth-First (DFS), Uniform-Cost (UCS), Greedy and A* search algorithms respectively.

Implementation
The cost associated with each flip is equal to the number of pancakes that are being flipped. For instance, the cost of one flip between pancake 4 and 1 from the state "4123" to "4321" is equal to 3. Use the same heuristic function that was discussed in the class: the ID of the largest pancake that is still out of place. For DFS, you don't need to consider a cost and heuristic. function.
Add as many comments as you can to your code, so that it's easy to understand your implementation, including the role of functions, variables, etc. Specifically, make it clear how your fringe is implemented and employed. Use an informative name for your fringe and add comments where you define that.

Tie Breaking
When needed for any of the four search algorithms, use the following tie-breaking mechanism:

"when there is a tie between two nodes, a node with a larger numerical ID will be chosen (imagine you read the IDs as a four-digit number)"
For instance, if there is a tie between 4321 and 3421, then 4321 will be chosen as 4321>3421. Also, between 3421 and 3412, 3421 will be chosen.

Output

Your program has to print the steps that the specified algorithm (e.g., DFS) finds to solve the problem. In other words, simply prints the solution that the algorithm finds. For each state (except the final state), use the character "|" to show where the flip to go to the next step happens. For all of the algorithms print the value for actual cost (function g, although DFS doesn't use it), and for the greedy and A* also print the value of the heuristic function (function h) in each step. The following is an example of an input and output of the program.

Input:

4132g

Output (possible):

41|32 g=0, h=3
4|123 g=2, h=3
4321 g=5, h=0

Note that the values for g and h correspond to the *current* state, and the character "|" denotes the location of the flip to go to the *next* state.

Programming Language

As previously discussed you can use any language that the TA approves (this includes Python and Java). There should be also a post on Piazza about this.

User Interface

You can also use a visual user interface (like textbox and buttons) for input and output. Make sure it's clear how the program operates and how the output can be interpreted. The steps taken to reach the goal have to be clear from the output.

Grading

Total points will be 100. For each of the four requested search algorithms, you will receive 25 points. 20 points will be for the code, and 5 for the correct output. Use a different function for each of the search algorithms, even if it needs copying a same piece of code. Use easily distinguishable names for your functions.

Final Note

Don't start late! It might take you longer than you expect to finish the homework.