Assignment 1: Search Methods

Deadline

Submission: 11:59pm, 16 April 2021 (Friday week 6, Sydney time).

This assignment is worth 12 marks = 12% of your final mark. It is an individual assignment.

Late submissions policy

Late submissions are allowed for up to 3 days late. A penalty of 5% per day late will apply. Assignments more than 3 days late will not be accepted (i.e. will get 0 marks). The day cut-off time is 11:59pm.

Programming languages

Your implementation can be written in Python, Java, C, C++ or MATLAB. The assignment will be tested using the language versions as described in the "How your program will be run" section below, so it is important that your program can be run in the specified versions.

Submission

Your assignment must be submitted using PASTA (https://comp3308.it.usyd.edu.au/PASTA).

If you are off-campus, you'll need to be connected to the university VPN in order to access the PASTA website. You can read this page to find out how to connect to the University Cisco VPN. The students in China should use the new (FortiClient) VPN, which is the special VPN for accessing university resources from China. If you are on-campus and connected to the University wireless network, you don't need a VPN to access PASTA.

Update: To access PASTA from China, it may be necessary to run both the FortiClient VPN and Cisco VPN; connect to FortiClient VPN first and then to Cisco VPN.

1. The 3-digit puzzle

In this assignment, you will implement a number of search algorithms to solve the 3-digit puzzle.

Given are two 3-digit numbers called S (start) and G (goal) and also a set of 3-digit numbers called $forbidden$. To solve the puzzle, we want to get from S to G in the smallest number of moves. A move is a transformation of one number into another number by adding or subtracting 1 to one of its digits. For example, a move can take you from 123 to 124 by adding 1 to the last digit or from 953 to 853 by subtracting 1 from the first digit. Moves must satisfy the following constraints:

1. You cannot add to the digit 9 or subtract from the digit 0;
2. You cannot make a move that transforms the current number into one of the forbidden numbers;
3. You cannot change the same digit twice in two successive moves.

Note that since the numbers have 3 digits, at the beginning there are at most 6 possible moves from S. After the first move, the branching factor is at most 4, due to the constraints on the moves and especially due to constraint 3.

For the purpose of this assignment numbers starting with 0, e.g. 018, are considered 3-digit numbers.

2. Tasks

1. Write a program to find a solution of the puzzle using the following 6 search strategies: BFS, DFS, IDS, Greedy, A* and Hill-climbing. Use the Manhattan heuristic as a heuristic for A* and Greedy and also as the evaluation function in Hill-climbing.
   The Manhattan heuristic for a move between two numbers A and B is the sum of the absolute differences of the corresponding digits of these numbers, e.g. $h(123, 492) = |1 - 4| + |2 - 9| + |3 - 2| = 11$.
2. Avoid cycles. When selecting a node from the fringe for expansion, if it hasn't been expanded yet, expand it, otherwise discard it. *Hint: It is not just the three digits that you need to consider when determining if you have expanded a node before. Two nodes are the same if a) they have the same 3 digits; and b) they have the same 'child' nodes.*
3. Generate the children in the following order:
   a. 1 is subtracted from the first digit
   b. 1 is added to the first digit
   c. 1 is subtracted from the second digit
   d. 1 is added to the second digit
   e. 1 is subtracted from the third digit
   f. 1 is added to the third digit

   Example: the order of the children of node 678 coming from parent 668 is 578, 778, 677, 679. Note that there are no children for the second digit as it already has been changed in the previous move (constraint 3).
4. For the heuristic search strategies: if there are nodes with the same priority for expansion, expand the last added node.
5. Set a limit of 1000 expanded nodes maximum, and when it is reached, stop the search and print a message that the limit has been reached (see section "Input and Output" for the exact message required).

3. Input and Output

As your program will be automatically tested, it is important that you adhere to these strict rules for program input and output.

Input

Your program should be called ThreeDigits, and will be run from the command line with the following arguments:

1. A single letter representing the algorithm to search with, out of B for BFS, D for DFS, I for IDS, G for Greedy, A for A*, H for Hill-climbing.
2. A filename of a file to open for the search details. This file will contain the following:

```
start-state
goal-state
forbidden1,forbidden2,forbidden3,…,forbiddenN (optional)
```

   For example, the file `puzzle.txt` might contain the following:

```
345
555
355,445,554
```

This file means that the search algorithm will start at state 345, and the goal is state 555. The search may not pass through any of 355, 445 or 554. Remember that the last line may not be present; i.e. there are no forbidden values.

The following examples show how the program would be run for each of the submission languages, assuming we want to run the A* search algorithm, and the input is in a file called `sample.txt`.

Python (version 3.7.0):

```
python ThreeDigits.py A sample.txt
```

Java (version 8):

```
javac ThreeDigits.java
java ThreeDigits A sample.txt
```

C (gcc version 6.3.0):

```
gcc –lm -w -std=c99 –o ThreeDigits ThreeDigits.c *.c
./ThreeDigits A sample.txt
```

C++ (gcc version 6.3.0):

```
g++ –o ThreeDigits ThreeDigits.cpp
./ThreeDigits A sample.txt
```

MATLAB (version R2018a):

```
mcc -m -o ThreeDigits -R -nodisplay -R -nojvm ThreeDigits
./run_ThreeDigits.sh <MATLAB_install_directory> A sample.txt
```

Note: MATLAB must be run this way (compiled first) to speed up MATLAB running submissions. The arguments are passed to your `ThreeDigits` function as strings. For example, the example above will be executed as a function call like this:

`ThreeDigits('A', 'sample.txt')`

Output

You program will output **two lines only**. The first line will contain the solution path found from the start node to the goal node (inclusive), in the form of states separated by commas. If no path can be found with the given parameters, then the first line should be "No solution found.".

The second line should be the order of nodes expanded during the search process, in the form of states separated by commas. If no path was found, this should still print the list of expanded nodes. Remember that this list should never exceed 1000 states (point 5 in the Tasks section).

Examples

`sample.txt:`

```
320
110
```

Note that the outputs of BFS and IDS here are quite long, and so the second line has wrapped. These outputs are still only two lines.

Search Method	Output
BFS	320,220,210,110 320,220,420,310,330,321,210,230,221,410,430,421,210,410, 311,230,430,331,221,421,311,331,110
DFS	320,220,210,110 320,220,210,110
IDS	320,220,210,110 320,320,220,420,310,330,321,320,220,210,230,221,420,410, 430,421,310,210,410,311,330,230,430,331,321,221,421,311, 331,320,220,210,110
Greedy	320,310,210,211,111,110 320,310,210,211,111,110
Hill-Climbing	No solution found. 320,310,210
A*	320,220,210,110 320,310,210,220,210,110

4. Submission Details

This assignment is to be submitted electronically via the PASTA submission system.

Your submission files should be zipped together in a single .zip file and include a main program called ThreeDigits. Valid extensions are .java, .py, .c, .cpp, .cc, and .m. Zip only the submission files, not the folder – when your zip file is unzipped there should be only submission files, not a folder with submission files. Only .zip format is accepted; do not use any other format, e.g. .rar or .7z. If your program contains only a single file, then you can just submit the file without zipping it.

Upload your submission on PASTA under **Assignment 1**.

Detailed submission instructions:

- Find "Assignment 1" and press "Submit" (the red icon on the right). Then press "Choose File" and attach the file, then press "I accept" after reading the University policy on academic honesty and agreeing with it.
- If you see a message indicating that your code is queued for testing, this means that your code has been uploaded successfully.
- If you see a red error box, you need to fix any problems indicated before your submission will be accepted.
- Once your program has been tested, the page will tell you to refresh for results, so refresh the page. You should see green and red boxes. Each box corresponds to a test; a red box indicates that your program has failed the respective test, a green box indicates that your program has passed the test and a grey box indicates that the test is hidden (you will see the result of the test after the due date).
- If you have red boxes, you can see which tests were not passed by clicking on "Assignment 1". Correct the errors (go back to your development environment, re-write your code and test it carefully) and then submit again in PASTA.

5. Academic Honesty

Please read the University policy on Academic Honesty very carefully:
https://sydney.edu.au/students/academic-integrity.html

Plagiarism (copying from another student, website or other sources), making your work available to another student to copy, engaging another person to complete the assignments instead of you (for payment or not) are all examples of **academic dishonesty**. Note that when there is copying between students, both students are penalised – the student who copies and the student who makes his/her work available for copying

The University penalties are severe and include: 1) a permanent record of academic dishonesty on your student file, 2) mark deduction, ranging from 0 for the assignment to Fail for the course and 3) expulsion from the University and cancelling of your student visa. In addition, the Australian Government passed a new legislation last year (Prohibiting Academic Cheating Services Bill) that makes it a **criminal offence** to provide or advertise academic cheating services - the provision or undertaking of work for students which forms a substantial part of a student's assessment task.

Do not confuse legitimate co-operation and cheating! You can discuss the assignment with another student, this is a legitimate collaboration, but you cannot complete the assignment together – this is an individual assignment and everyone must write their own code.

To detect code similarity in this assignment, we will use the system MOSS which is **extremely good**. If you cheat, the chances that you will be caught are very high. **Be smart and don't risk your future or break the law by engaging in plagiarism and academic dishonesty!**