TITLE
Uninformed Search using Python.

OBJECTIVES
- Solving problems by search
- Discuss BFS, DFS.
- Learn how to formalize a problem as a search problem.

PREREQUISITES
- Python 3+.
- Online Python IDE
- Search Problem Solver Framework
- Small Game (The wolf, the goat and the cabbage)
- http://www.novelgames.com/en/missionaries/

RESOURCES
- Course Slides
- Python Tutorial
- Python Tuples

LAB

Use the archive provided for this laboratory. The archive contains all the Python code that is necessary for solving a problem using searching algorithms.
The code is well-commented so that you can understand what it actually does.

In order to solve a problem using this small search framework, you have to create a subclass of class Problem and implement all the required functions (read the comments of the class Problem).

Remove unused imports from the main file:
```
from eight_puzzle import EightPuzzle
from nqueen import NQueensProblem
```

The wolf, the cabbage and the goat
A farmer with his wolf, goat, and cabbage come to the edge of a river they wish to cross. There is a boat at the river's edge, but, of course, only the farmer can row. The boat can carry a maximum of the farmer and one other animal/vegetable. If the wolf is ever left alone with the goat, the wolf will eat the goat. Similarly, if the goat is left alone with the cabbage, the goat will

eat the cabbage. Devise a sequence of crossings of the river so that all four of them arrive safely on the other side of the river.

Formalization of WCG problem:
State Space: tuple (wl, cl, gl, m, wr, cr, gr) with 0 ≤ wl, cl, gl, m, wr, cr, gr ≤ 1
Initial State: (1, 1, 1, 'LEFT', 0, 0, 0)
Successor Function(Actions): from each state, we can move across the man with the goat, the man with the cabbage or the man with the wolf. Note that not all states are legal. State (1, 0, 1, 1, 0, 0) is illegal.
Goal State: (0, 0, 0, 'RIGHT', 1, 1, 1)
Path Costs: each action as a cost of 1

You will find the implementation for this problem in the provided code. The file is vcl.py. You can use this problem as a starting point for the next one.

Missionaries and Cannibals
Three missionaries and three cannibals are on one side of a river that they wish to cross. A boat is available that can hold at most two people and at least one. You must never leave a group of missionaries outnumbered by cannibals on the same bank.

State Space*: tuple(cl, ml, boat, cr, mr)*
Initial state*: (3, 3, 'L', 0, 0)*
Actions*:* from each state, we can move across two cannibals, a cannibal and a missionary or two missionaries. Note that not all states are legal. State (1, 2, 'RIGHT', 2, 1) is illegal.
Goal*:(0, 0, 'R', 3, 3)*

Implementation hint:
#Move 1M and 1C from L -> R

```
newState = (currentState[0] - 1, currentState[1] - 1, 'DREAPTA', currentState[3] + 1,
currentState[4] + 1)
if is_valid(newState):
   actions.append(newState)
```