Programming project 1

Due: see Canvas
Group work: You may work in groups of 1-3. Include all group members when submitting to Gradescope.

Write a Python class, `MissCannibalsVariant`, that defines the variant of the Missionaries & Cannibals puzzle **with boat capacity = 3** (same problem from HW #1). The puzzle can be set with an arbitrary number of Missionaries and Cannibals on the left bank. Your code should be usable by a search algorithm to solve the puzzle.

The class has two instance variables, `self.N1` and `self.N2`, which stores the total number of missionaries and the total number of cannibals starting from the left bank. These values will be set in the main function so that your class can be used to solve puzzles with different numbers of missionaries and cannibals. E.g., 3 missionaries and 2 cannibals.

Represent the state by a 3-tuple, two integers and a boolean: `(m, c, onLeft)`, which represents the number of missionaries on the left bank (note: the number of missionaries on the right bank is then `self.N1-m`), the number of cannibals on the left side, and if the boat is on the left respectively.

Represent actions using strings: 'M', 'C', 'MM', 'MC', 'CC', 'MMC', … It is not necessary to represent the direction of the boat as this will be clear from the state (e.g., if the boat is on the left, then the boat will cross to the right).

The class should be usable in the main function below (**this starter code is also on Canvas**) to print the sequence of actions to reach the goal state.

```
from search import *

class MissCannibalsVariant(Problem):

    def __init__(self, N1=4, N2=4, goal=(0, 0, False)):
        """ Define goal state and initialize a problem """
        initial = (N1, N2, True)
        self.N1 = N1
        self.N2 = N2
        super().__init__(initial, goal)

if __name__ == '__main__':
    mc = MissCannibalsVariant(4,4)
```

```
    #print(mc.actions((3, 3, True))) # Test your code as you develop! This should
return  ['MC', 'MMM']

    path = depth_first_graph_search(mc).solution()
    print(path)
    path = breadth_first_graph_search(mc).solution()
    print(path)
```

Your code should print something like:
`['MC', 'M', 'CC', 'C', 'MM', 'MC', 'MM', 'C', 'CC', 'M', 'MC']`
(Your sequence of actions will depend on your implementation as there are a few different solutions).

Hints:

- The class must be a subclass of class Problem in the search.py code.
- The class will be similar to class EightPuzzle (in structure only, the implementation will be VERY different) in search.py. Specifically, your class should have a
    1. constructor **(already completed)**
    2. method `goal_test(state)` **(default in the Problem superclass is sufficient)**
    3. method `result(state, action)` that returns the new state reached from the given `state` and `action`
    4. method `actions(state)` that returns a list of valid actions in the given `state`
- I recommended implementing the methods in the order above and to test each method individually.
- As the action is a string, the number of missionaries/cannibals crossing over can be calculated by just counting a 'M' or 'C'. E.g., `action.count('MC').`

Submit via Gradescope:

Submission is via Gradescope. Upload a single file with your class. The name of the file **must be** `misscannibalsvariant.py`. The name of the class **must be** `MissCannibalsVariant`. Gradescope will run a few unit tests automatically and show you the results. Include all group members when submitting to Gradescope.