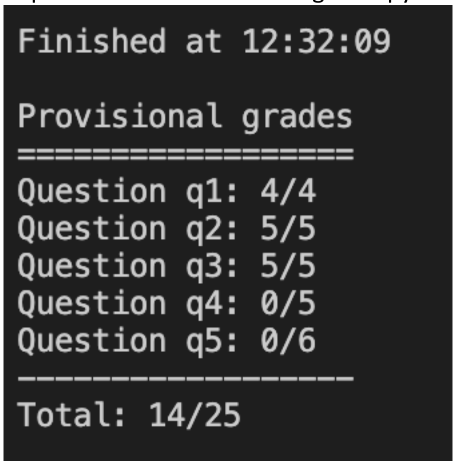
COSE361(03) - assignment #2
2022320006 최진혁

a. Capture of the result of autograder.py



```
Finished at 12:32:09

Provisional grades
==================
Question q1: 4/4
Question q2: 5/5
Question q3: 5/5
Question q4: 0/5
Question q5: 0/6
------------------
Total: 14/25
```

b. Three discussions when playing Pacman (2 points for each discussion. Total 6 points.)

1. How can alpha-beta pruning make it more efficient to explore the minimax tree?

   In minimax search tree, sorting the successors can be efficient. Sort successors in decreasing order in max agent, and increasing order in min agent, so that pruning can be done in first successor.

   Using dynamic programing is also a good option. We can construct the table of the (bestScore, bestAction) for each state, and use it when the game state is same as before.

2. Is there a situation where the Reflex agent performs better than the minimax or alpha-beta pruning algorithm?

   When the search tree is deep, performance of the minimax or alpha-beta algorithm can be bad. Even if we use limited depth search, the computation can be reduced, but the result can be not accurate, or not optimized.

   Also, if the evaluation function is not good, performance of the minimax or alpha-beta algorithm can be bad. Reflex agent use the evaluation function that consider the foods and the ghosts. However, minimax search and alpha-beta pruning algorithm use only scoreEvaluationFunction(), which is not better than Reflex agent's.

3. Ask yourself one question and answer.

   Q: Is there a situation where the minimax search performs better than the alpha-beta pruning algorithm?
   A:
   In the worst case, like the situation where the alpha-beta pruning never happened, minimax can perform better than alpha-beta pruning algorithm. Because there is no pruning done, so we should search all nodes as much as minimax, with more computation for pruning.