Our names are Peleg Ben Barak (id 207233560) and Adi Tal (id 211373493)

The code consists of functions applying the policy iteration algorithm for finding the optimal strategy.

We begin by applying a reward of -1, 0, or 1 for the result of an ‘episode’ (one blackjack game), which is given to the agent upon completing a game.

Our first step is to calculate the transition probability matrix in the function:

```
def transition_matrix(env, num_of_samples=num_samples):
```

which returns a dictionary with the probability of passing between each 2 states based on the action taken. The dictionary keys are state, action, next_state. We chose to represent using a dictionary and not a matrix for code clarity and readability. The code converges rather quickly to optimal values and therefore the runtime isn’t heavily affected by the possible added time complexity.

We then continue to answer 2B to find the best policy using

```
def policy_iteration():
```

which activates the algorithm taught in class – we iterate using greedy policy improvement, where for each policy we calculate the expected value using the formula.

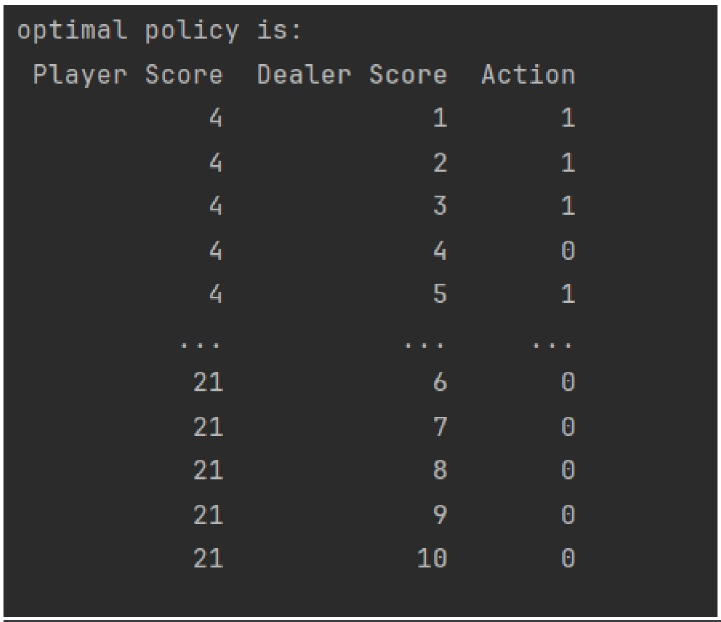
Our code includes a function

```
def apply_policy(policy):
```

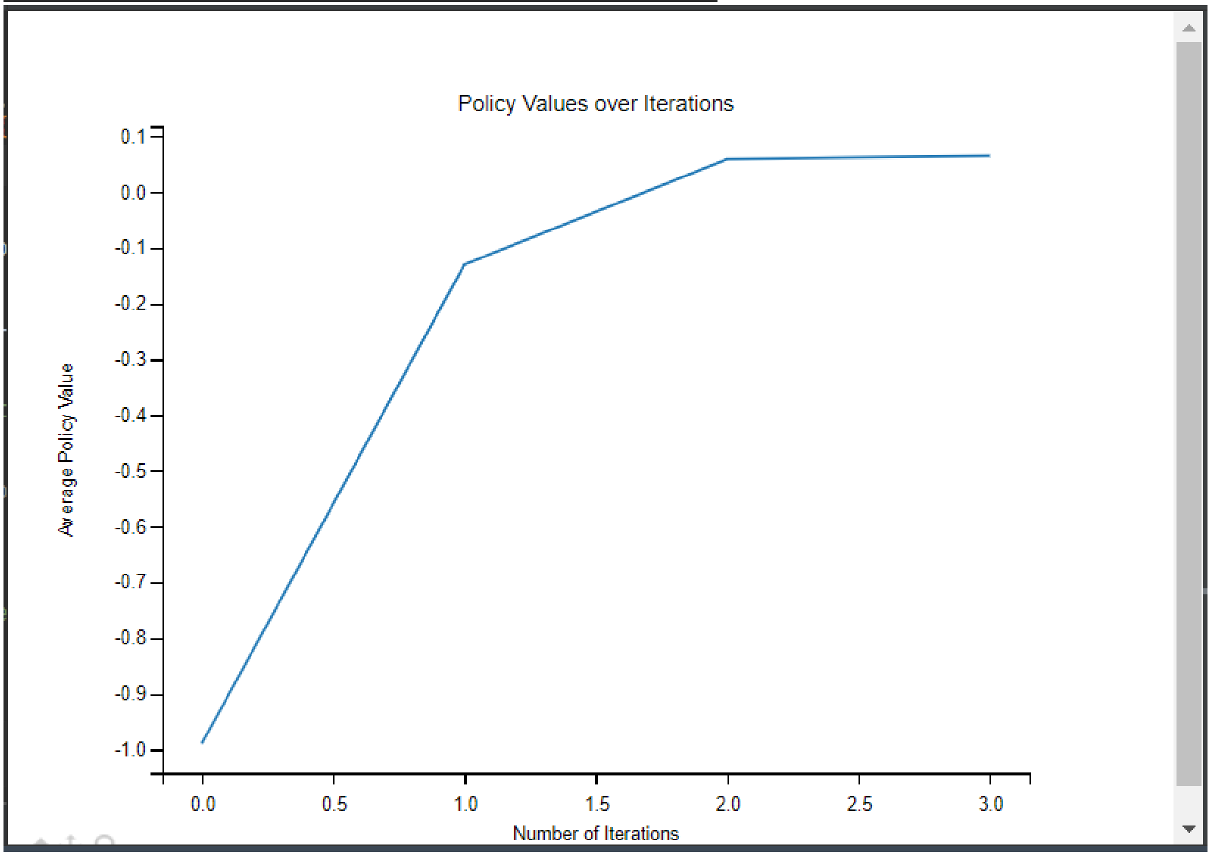
which received a policy dictionary and applies it to a single episode.

We have the method of `def value_function_q3(state, policy):` which returns the expected value of a given state under the given policy.

Finally we represent the policy iteration using a matplotlib pyplot (which normally terminated within 4 iterations), and the actions taken under the optimal policy as a pandas dataframe.



```
optimal policy is:
  Player Score  Dealer Score  Action
             4             1       1
             4             2       1
             4             3       1
             4             4       0
             4             5       1
           ...           ...     ...
            21             6       0
            21             7       0
            21             8       0
            21             9       0
            21            10       0
```


Policy Values over Iterations
Average Policy Value
Number of Iterations