1. How does your program calculate the utility of each terminal state? Describe briefly.

For terminal states, I use, instead of infinity, a large number LARGE_NUM = 100000 to represent the utility of winning and losing. My program first checks if the current state is terminal by checking if its get_possible_moves returns nothing. If there is no possible moves, then whoever is playing the turn loses.

In addition, I took consideration of depth into my utility calculation that utility = LARGENUM // (depth+1) such that the terminal state with lowest depth will be chosen.

2. How does your program estimate the utility of each non-terminal state? Describe your evaluation function in a few sentences.

For non-terminated state, I use curr_score - opp_score, where score = 4*king + 2*normal_checker_at_center + 1*other_normal.

Normal_checker_at_center means the men pieces that are sitting inside and on (2,2) to (5,5) in a board of (0,0) to (7,7). This is because, according to https://www.ultraboardgames.com/checkers/tips.php, we should take control of the center board (2 scores for taking middle) and take as much king as possible (4 scores for kings).

3. Does your program perform other optimizations, such as node ordering or state caching? If so, describe each optimization in a few sentences.

Yes, I implemented caching for utility function. My cache dictionary contains hash(str(board)) as its key and utility function along with the turn as its value. So that whenever I want to calculate the utility of a state, I go check for the hash code of my board to see if there is a cache hit. If hit, check if the turn matches, matching turn returns the value and non-matching return negative of that value.

In addition, I implemented node ordering using sorted function with key argument of elem[1] where my elem[1] is the utility of that state. However my sorting optimization does not give much improvement under low depth limit (but with high depth limit the program is likely to timeout), therefore I discarded the node ordering implementation. But I kept the functions in my code if you want to check it.