1.
   a. Please see attached Python file
   b. Josiah did the porting of the tic-tac-toe logic into the Othello program, implemented the edge-play heuristic, and adapted the existing minimax/alphabeta code. Jason did background research into said edge-play heuristic, and general bug-squashing.
2.
   a. There were several strategies involved in trying to make our Othello player "good". Most importantly, our eval function is tuned such that corners are given high priority, followed by edges, whereas near-corner squares are avoided (unless the player also controls the corner). In addition, the program is given a large ply depth to search, increasing its effectiveness against short-sighted human players. As with Tic-Tac-Toe, alpha-beta pruning is used to decrease the search space and increase the calculations that can be performed, thus improving the accuracy of our moves.
   b. Our evaluation function works as follows: corner pieces are worth 50 points. Edge pieces next to the corner, which are dangerous and often lead to corner captures by the opponent, are worth -1 points. Edge pieces two from the corner are worth 5 points, as they are useful to have in capturing edge pieces. The inner corner pieces (one move away in both directions from the corner) are worth -10 points, as they are extremely difficult to hold. Finally, all other pieces are worth 1 point. The function sums up the points of a position by matching the available squares versus their point values. So, for any calculated flip, the evaluation function returns the point value of that board for that player at that move. The heuristic is compared against the raw point value (number of pieces) of that color on the board. If there is a lot of inner play going on, the heuristic will often be less than the raw points, so the raw score is used instead.
   c. In the alpha-beta scoring function, the time is calculated after every call. If this time minus the original, passed in time from the main play loop is at the 20-second limit, the code breaks, and returns the optimal search tree found so far. This avoids the additional overhead of iterative DFS, while still finding, in general, a superb solution that looks at least 6 plies in advance. While the best Othello players can look nearly 30 plies in advance, our range of 6-12 is often plenty to defeat a human opponent.
   d. Please see attached .txt file (with nicer formatting)