CHAPTER 3
RESEARCH METHODOLOGY

The program is created to implement Negascout algorithm in Othello games. Thus, the program is revolved around Negascout algorithm. To implement Negascout, data structure to save the data is created first. Afterwards, move generator is created and stored in tree nodes. State evaluator is created next to provide comparation value. Negascout algorithm implementation comes next. Finally, the user interface is created to provide visual representation of the board.

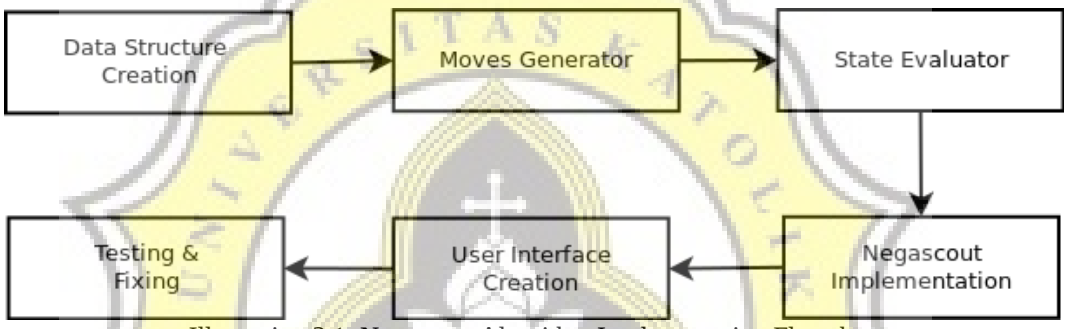

Illustration 3.1: Negascout Algorithm Implementation Flowchart

3.1 Data Structure Creation

The first step is designing the data structure. Tree data structure is used to store the possible moves. Array data structure will be used to store the board. The discs on the board square are stored by an integer value of 2 if it is a black piece, 1 if it is a white piece, and 0 if there is no piece on the square.

3.2 Moves Generator

The next step is to generate the possible moves. The possible moves are generated by generating all the square adjacent to enemy discs. Then, the square is checked if it is a valid move or not. The basic requirement is the square must be empty. The next requirement is the move will flip at least one of the enemy disc. Then the valid move is stored in a simple array.

3.3 State Evaluation

State evaluation is about evaluating the current state of the games. There are mobility factor, disc factor and weight factor. Mobility is evaluated by counting the possible moves for both player.s Disc is evaluated by the number of the disc owned by both players. At the early and mid-game stage, having fewer pieces on the board is considered favorable. Weight is evaluated by checking the square weight. The weight will be positive if the square is deemed favorable and negative if it is deemed unfavorable. For instance, the corner square will have a value of 100 and the adjacent square will have -25.

3.4 Tree Searching Algorithm

Negascout algorithm is the algorithm used for tree searching. After generating possible moves, Negascout will search for the nodes that have the maximum evaluation. The best move will be selected and done. Then the program will switch to next player turn.

3.5 User Interface

User interface is created after the algorithm and data structure is implemented. The program will have to represent the board to the player. The player simply has to input the board by clicking the desired square on the board.

3.6 Testing

The last step is testing. Testing will be done by running the program multiple times to check any issues regarding technical structure or graphical interface.