Documentation for Puzzle.cpp

```
public:
   // Constructors
   Puzzle();
   Puzzle(vector< vector<int> > src);
   Puzzle(ifstream& file);

// Used to print out the width of the matrix
int width() const ;

// Used to print out the width of the matrix
int height() const;

// Checks if puzzle state is empty
bool empty();

// used to extract element at the given position
int at(int i, int j) const;

// Prints out the state
void printState() const;

// State clone;

vector<vector<int> > stateClone();

// Check completion of state;
bool checkStateCompletion();

// List valid moves individual pieces can make;
vector<char> moveList(int piece);

// Returns a list of all the moves from all the elements for a given state space
void allMoveList();
// Moves the given element in the selected direction
void applyMove(int piece, char dir);

// Normalizes the state
void normalize();
void swapIdx(int idx1,int idx2) ;
```

```
// Compares the object to another 2D vector
bool compare( vector< vector<int> > src )const;

// Compares two separate objects
bool compare(const Puzzle& p )const;

// Lists out all the elements in the state
vector<int> elementList();

// Overload compare operator for use with maps as key
bool operator<(const Puzzle& other) const;

// This methid was used to find the manhattan distance between the master block and the
// target celll
int manhattanMaster(Puzzle p);

// This method made use of both the h(n) and g(n), where h was the manhattan distance for
the master cell to the target cell. And a g(n) was added to compliment it where it added a value
of 1 for any non-zero neighboring blocks around the master block.
int heuristicVal(Puzzle p);
```