Следующая проблема, на которой я остановился, стало объяснение хода для компьютера.

Как объяснить компьютеру рамки, за пределы которых выходить нельзя?

Например, конь с поля B1 не может «скакать» назад. Но ведь во Вселенной компьютера такое возможно. Только тогда поле получит название “ ?-1 “

Для человека – это очевидный бред, но компьютер не видит поле.

Значит, есть два пути. Попытаться как-то задать поле, на котором будут фигуры, либо просто математически отсекать невозможные варианты.

Насчёт первого стоит и задуматься. Тогда это поле надо задать. Допустим, это будет таблица. Тогда нам нужна таблица, в которой переменные, (то есть фигурки) будут перемещаться по ней.

**И тут мне пришла в голову идея. А если не называть фигуры, а просто дать им значения? То есть ходить будет не пешка, а единица. Не слон, а 3.33. Тогда и стоимость фигуры будет известна, и то, как будет эта фигура двигаться, тоже известна – ведь у разных фигур разный «вес», а у одинаковых и ходы одинаковы. То есть теперь по таблице будут перемещаться числовые значения.**

Но, к сожалению, какие-нибудь нормальные таблицы в Питоне я не нашёл.

Так что идём ко второму варианту. Математически отсекать невозможные ходы. То есть заранее нужно установить, что значения могут быть в промежутке от 1 до 8 и от А до Н.

ИДЕЯ!

ХОДИТЬ МОГУТ КЛЕТКИ, В КОТОРЫЕ «ЗАШИТЫ» ФИГУРЫ! - подумать!

Хорошо, допустим, что мы научили компьютер понимать ход. Но что дальше? Он должен его куда-то записать, то есть запомнить. И как писать несколько ходов? Непонятно.

Остановимся на примере белой пешки.

def PIECE\_MOVES (piece, LETTER, NUMBER):

if (piece == 'WHITE\_PAWN'):

мы требуем у функции дать нам положение фигуры на поле – LETTER и NUMBER, а также определить тип фигуры. В данном случае, у нас пешка.

Теперь можно задать правило хода на две клетки вперёд. То есть:

If (NUMBER== 2):

New\_NUMBER\_1 = NUMBER + 2

else:

New\_NUMBER\_2 = NUMBER + 1

Таким образом мы объяснили, что если белая пешка стоит на второй горизонтали, то она может пойти на две клетки вперёд, а если нет, то только на одну. Есть только проблемка. Со второй горизонтали пешка может пойти на два поля вперёд, но может и только на одно.

То есть, идеал программы:

If (NUMBER== 2):

New\_NUMBER\_1 = NUMBER + 2

New\_NUMBER\_2 = NUMBER + 1

else:

New\_NUMBER\_1 = NUMBER + 1

Теперь, в комментарии можно ещё написать, как будет выглядеть перемещение этой пешки.

То есть мы заключили компьютеру гипотетическую возможность хода, но ход не выполнен. Но этим можно заняться позже.

Однако сразу возникает резон. А зачем нам писать правила того, как ходить так, чтобы не таранить свои же фигуры и не съедать фигуры противника?

Правильный ответ: задать их сразу. То есть сразу же написать, что пешка может идти вперёд, если там есть куда идти. Это делается просто: нужно просто проверить заранее, если клетка впереди пуста, то можно. То есть условие проверки пустоты передней клетки, а затем уже проверка положения пешки.

Следует также отметить, что пешки могут ещё и бить фигуры наискосок, а также en passant. Со вторым я разберусь позже, когда домыслю систему хода, потому что взятие на проходе напрямую зависит от хода противника, потому что по правилу наша (белая) пешка должна стоять на 5 горизонтали. А пешка противника, то есть только слева или справа, должна с седьмой горизонтали шагнуть на 2 клетки вперёд, тогда мы и побьём их фигуру.

А вот насчёт взятия всё и сложнее, и проще. С одной стороны, мы можем выполнить простой перебор того стоит ли там конкретно вражеская пешка, вражеская ладья, вражеский ферзь, вражеский конь, вражеский слон.

Но можно просто проверить свойство: если цвет фигуры вражеский, то есть чёрные в нашем случае. То есть тогда нам надо для каждой фигуры ввести четыре свойства:

1. Что это за фигура
2. Цвет
3. Положение фигуры по вертикали
4. Положение фигуры по горизонтали.

Тогда придётся переделывать весь код, но так будет меньше переборов. С другой стороны, проверка пешки не станет труднее! То есть мы будем всего лишь спрашивать два отдельных параметра. Но тут возникает проблемка. Если мы призываем фигуру к клетке то как тогда мы будем обозначать разные ладьи? Если мы будем задавать фигуру, то нам придётся класть в неё эти 4 значения. Можно ли это сделать через функции?

То есть нам желательно не разделять пешки по номерам. Так, нам надо привязать фигуру к клетке и при этом не создавать другие номера. Попробуем решить эту проблему. Если не получится, то ничего страшного, перейдём к более простому: к перебору.

a = ['PAWN', 'WHITE', 'A', '2']

**А что если задать фигуру как массив?!**

И в установленном порядке [фигура, цвет, вертикаль, горизонталь] обозначать все фигуры. А считывание фигуры превратить в «вытаскивание» из массива соответствующих элементов.