OUR EXPECTATIONS:

1. We expect you to come up with a **simple console application** in the **language of your choice**. There is no need for a UI, or a web application.
2. With this exercise we are expecting to see how you write production ready code by focusing on:

    Simple and modular design.

    Clean code practices using OO / functional programming

    Unit test case coverage.

    Handling boundary conditions

    Code styles etc.

3. Please stay **within the boundaries** defined in the problem. Avoid over-thinking and over-engineering. Clearly state your assumptions wherever needed in the code or in the README file.
4. Submit your solution in a **public GitHub / Bitbucket repo.**

OTHER EXPECTATIONS (nice to have):

1. Please mention the **setup instructions** and **how to run the program** in the README file.
2. Divide the problem statement in smaller tasks / features and have small and atomic commits in your repo.

PROBLEM STATEMENT - CHESS:

You are required to create a program, which simulates an **empty chessboard** and the **movements of three types of chess pieces** on it.

Chessboard: The chessboard is an 8 x 8 grid with 64 cells in it. With 8 rows (1, 2, 3.... 8) and 8 columns (A, B, C.... H), each cell can be uniquely identified with its cell number. This has been illustrated below:

A8	B8	C8	D8	E8	F8	G8	H8
A7	B7	C7	D7	E7	F7	G7	H7
A6	B6	C6	D6	E6	F6	G6	H6
A5	B5	C5	D5	E5	F5	G5	H5
A4	B4	C4	D4	E4	F4	G4	H4
A3	B3	C3	D3	E3	F3	G3	H3
A2	B2	C2	D2	E2	F2	G2	H2
A1	B1	C1	D1	E1	F1	G1	H1

Chess pieces:

The game of chess has 6 unique types of pieces - King, Queen, Rook, Bishop, Horse and Pawn - each with their own unique types of movements.

What you have to do:

You have to simulate the movement of the following **three pieces** on an **empty** chessboard:

1. **Pawn** - It can only move 1 step at a time, in the vertical forward direction.
2. **King** - It can only move 1 step at a time, in all 8 directions (vertical, horizontal and diagonal)
3. **Queen** - It can move across the board in all 8 directions.

These movements will be illustrated in the Input and Output section below.

Inputs and Outputs to your program:

- **Input** - The input string to your program will be the **Type** of chess piece and it's **position** (cell number) on the chessboard. E.g. **Pawn, G1**
- **Output** - Once you execute the program, the output will be a string of all possible cells in which the chess piece can move from its current position. For the above input, the output would be **G2**.

Examples:

- Input - King, D5
- Output - C4, C5, C6, D4, D6, E4, E5, E6
- Illustration -

B7	C7	D7	E7	F7
B6	**C6**	**D6**	**E6**	F6
B5	**C5**	**D5**	**E5**	F5
B4	**C4**	**D4**	**E4**	F4
B3	C3	D3	E3	F3

- Input - Queen, E4
- Output - A4, B4, C4, D4, F4, G4, H4, E1, E2, E3, E5, E6, E7, E8, A8, B7, C6, D5, F3, G2, H1, B1, C2, D3, F5, G6, H7
- Illustration -

A8	B8	C8	D8	**E8**	F8	G8	H8
A7	**B7**	C7	D7	**E7**	F7	G7	**H7**
A6	B6	**C6**	D6	**E6**	F6	**G6**	H6
A5	B5	C5	**D5**	**E5**	**F5**	G5	H5
A4	**B4**	**C4**	**D4**	E4	**F4**	**G4**	**H4**
A3	B3	C3	**D3**	**E3**	**F3**	G3	H3
A2	B2	**C2**	D2	**E2**	F2	**G2**	H2
A1	**B1**	C1	D1	**E1**	F1	G1	**H1**