FROSTFIRE

This document is a step by step guide to Frostfire and mentions the basic game rules. The participants have to submit their codes in C/C++.

1 Terminologies

1.1 Game Board

The board consists of a hexagon of edge length 4 which will consist of 37 positions. 22 dragons are placed initially (11 per side). The two dragons being Tyragon and Abagon.Tyragon being the ICE dragon while Abagon being the FIRE dragon. There are 7 row markers on the board, spanning from A-G ,and 7 diagonal markers, spanning from 1-7. These markers are initialised such that each position has its own unique index (ROWDIA).

Initially each team will have its 11 dragons on either the top or bottom of the board (Tyragon on top). The 11 dragons of Tyragon will be placed as:-

- 4 Dragons in Row A
- 5 Dragons in Row B
- 2 in Row C at indicies C4 and C5.

where 'C' is Row Number and '4,5' are Diagonal Numbers.
The 11 dragons of Abagon will be placed as :-

- 4 Dragons in Row G
- 5 Dragons in Row F
- 2 Dragons in Row E at indicies E4 and E3.

where 'E' is Row Number and '3,4' are Diagonal Numbers.

A4 A5 A6 A7
B3 B4 B5 B6 B7
C2 C3 C4 C5 C6 C7
D1 D2 D3 D4 D5 D6 D7
E1 E2 E3 E4 E5 E6
F1 F2 F3 F4 F5
G1 G2 G3 G4

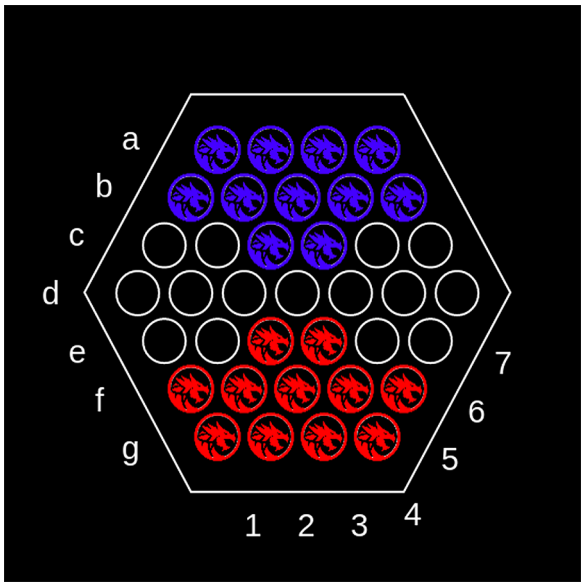

Figure 1

1.2 Initial Input:-

1.3 Input Format:-

The input specifies the opponents previous move. The input to your bot will recieve a 5 character string (including Spaces) of the format :-
"Ax By"
A=Row of First dragon
x=Diagonal of First dragon
B=Row of Second dragon
y=Diagonal of Second dragon

1.4 Output Format:-

The output specifies your bots move. The output by your bot must provide a 5 character string (including Spaces) of the format :-
"Ax By"
A=Row of First dragon
x=Diagonal of First dragon
B=Row of Second dragon
y=Diagonal of Second dragon

2 Gameplay

As the battle begins, each bot is either designated the number **One** or **Two**. The number represents the order in which the bots play their turns.

The objective of the game is to strategize the bot's moves such that, there remain:

- A minimum number of eight of its own dragons
- A maximum number of seven of the opponent's dragons

on the board.

This means that the battle ends when either of the bots have forfeited four of their dragons.

2.1 Dragon Moves

Moves are permitted in all directions, provided that the player's dragons continue to stay on board, that is, the player cannot push his own dragon out of the Board.

The Dragons can only move in pairs in the direction specified. For example, "Ax By" will have the dragon at index Ax, which is linearly adjacent to dragon at index By. The dragon must always move from Ax to By.

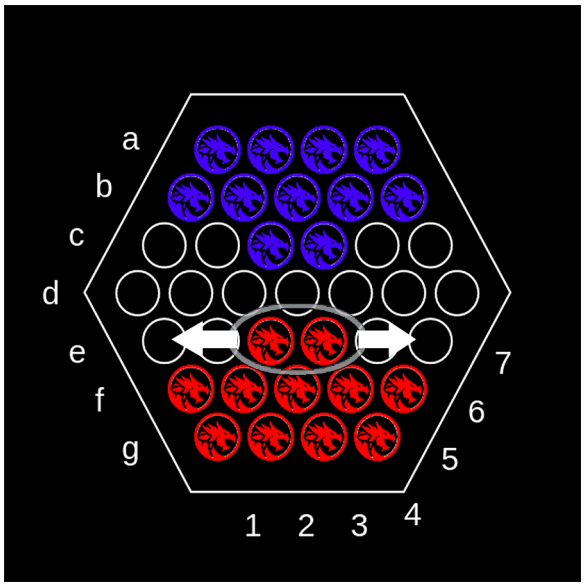

Figure 2

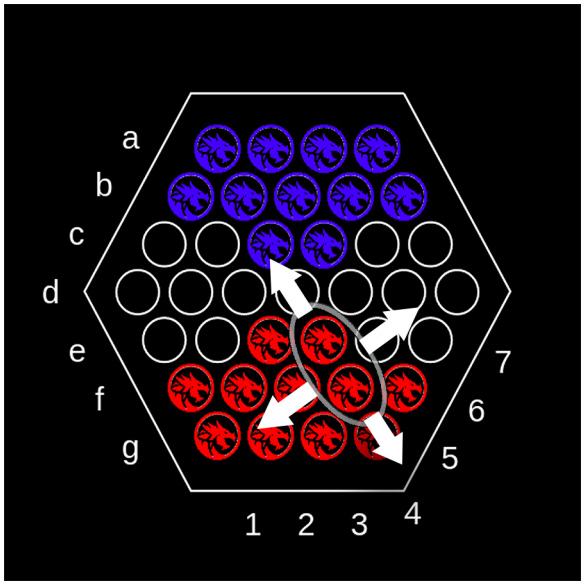

Figure 3

2.2 Pushing Criteria

The player's two **linearly moving** dragons can push only **one** opponent's Dragon **in the same direction.**

This means that, not more than two dragons can be used against an equal or greater number of the opponent's dragons.

Note: Linear moves indicate the movement along the rows or diagonals exclusively.

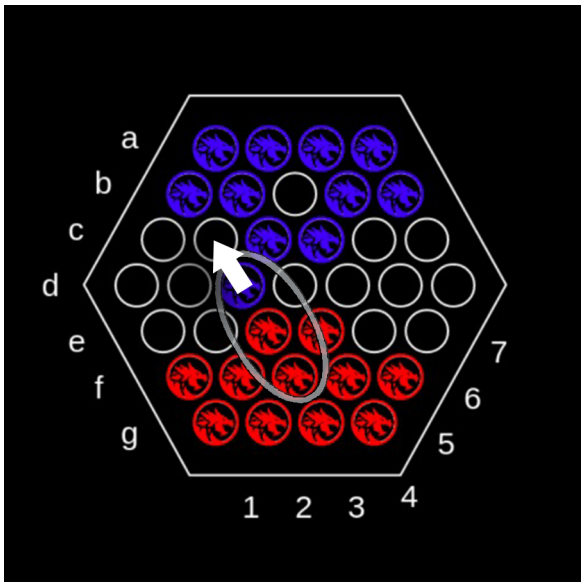

Figure 4

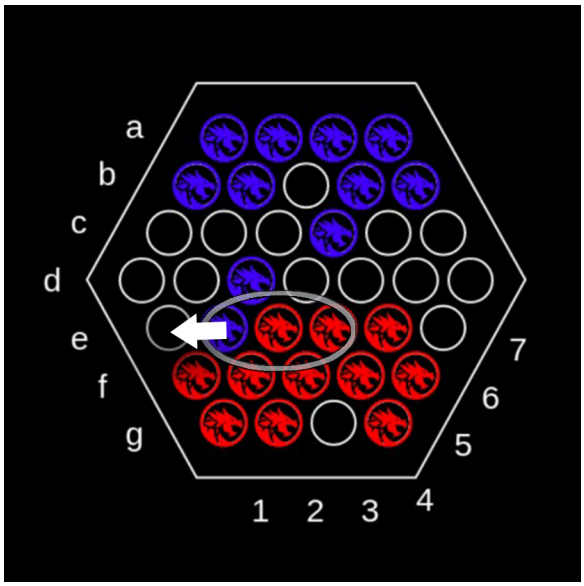

Figure 5

2.3 Invalid Moves

The following moves are deemed as **invalid**:

- The third dragon that is pushed belongs to the same team as the other two.

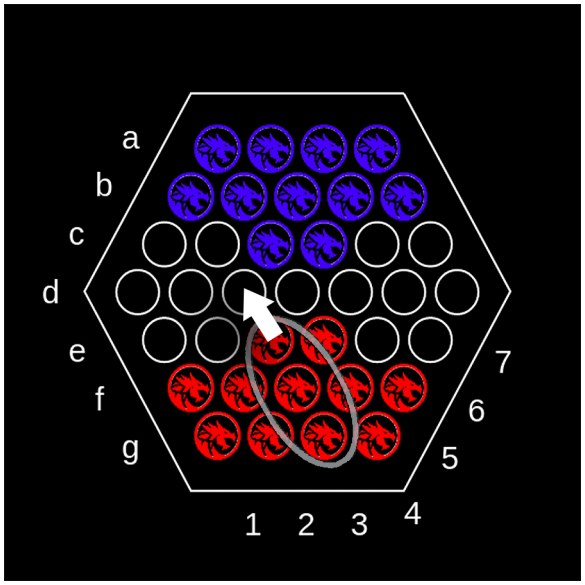

Figure 6

- **More than one** opponent's dragons are pushed at a time.

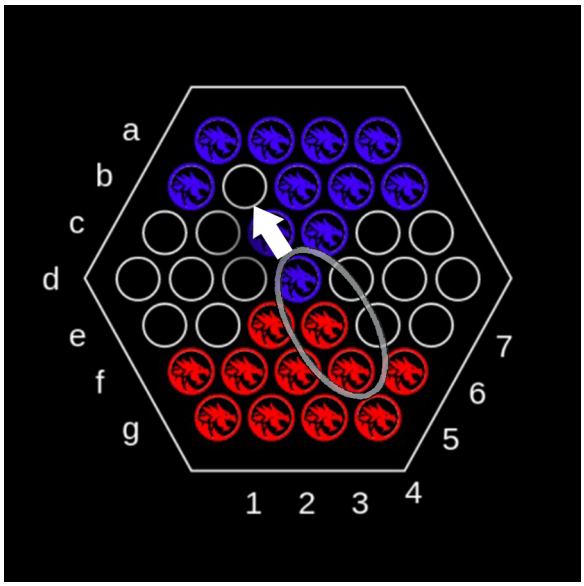

Figure 7

- The dragon is pushed off from the vertex from the board.

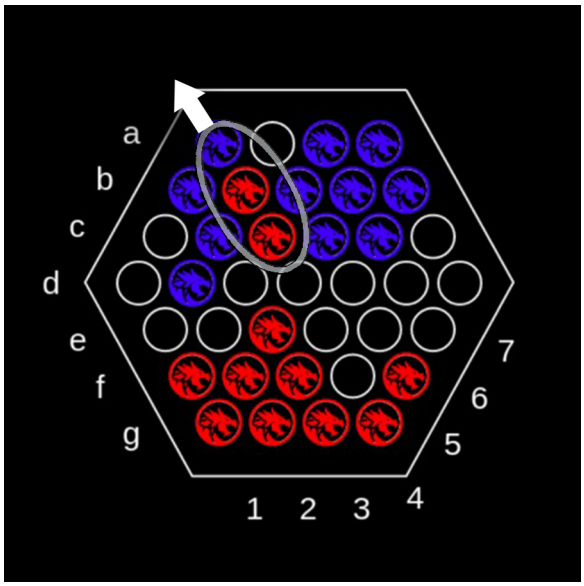

Figure 8

2.4 Elimination

A player's dragon is said to be eliminated when it lies on the edge and has been pushed off the board by the opponent.

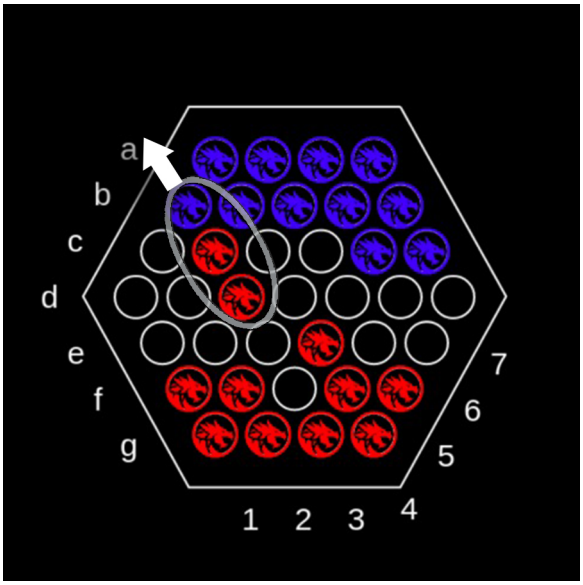

Figure 9

But, the move is said to be invalid if the dragon stands at a vertex of the board and is then pushed off the board.

2.5 End Of The Game

- The battle ends when a side loses **four** of its dragons. The opposite side is declared as the *winner*.
- The battle may also terminate if all dragons of a side are blocked or isolated. In such a situation, the Battle is declared as a **draw**.

Note: The bot program will be terminated automatically when game is over.

2.6 Bot Format

The player's bot(code) is run only once, with his/her logic executed in an unconditional while loop.
The format of the code should be as follows:

```
    Tyragon = 0
    Abagon = 1
    Unoccupied = -1
    boardPattern =
{
                                    0, 0, 0, 0,
                                 0, 0, 0, 0, 0,
                              -1, -1, 0, 0, -1, -1,
                           -1, -1, -1, -1, -1, -1, -1,
                              -1, -1, 1, 1, -1, -1,
                                 1, 1, 1, 1, 1,
                                    1, 1, 1, 1
};
    int main()
    {
      int botColor;
      frostfire board;
      setBoard(&board); //setting bot's board to the initial position
      acceptColor(&botColor);
      // bot object
      bot b(board, botColor);
      string move;
      string opponent_move;
      while (game has not ended)
      {
        // Tyragon starts
        if (bot is Tyragon)
        {
        // BOT'S TURN
          move = botMove(); // calculating the bot's move
          makeMove(move, &board); // making the move on the actual board
          print(move); //printing the bot's move
          activePlayer = Abagon;
          // OPPONENT'S TURN
          // input opponents move, assume that it's valid
          acceptInput(&opponent_move);
          makeMove(opponent_move,&board);
          activePlayer = Tyragon;
        }
        else // if bot is Abagon
```

```
        {
          // OPPONENT'S TURN
          // input opponents move, assume that it's valid
          acceptInput(&opponent_move);
          makeMove(opponent_move, &board);
          activePlayer = Abagon;
          setBoard(board); // setting bot's board to the current one
          // BOT'S TURN
          move = botMove(); // calculating the bot's move
          // validity of the move must be checked
          makeMove(move, &board); // making the move on the actual board
          print(move); //printing the bot's move
          activePlayer = Tyragon;
        }
      }
    }
```

Note:- No points will be awarded or retracted for draws or lost matches. We advise you to device your bots accordingly so that the bot plays a more agrressive game in order to win if necessary

3 Disqualification

Bot will be disqualified due to any of the following reasons:

- Bot does not respond within **2 seconds** (for C and C++) of its execution.
- Syntactical errors in the program.
- Bot returns an invalid move.
- Excessive resource usage (Bot should consume less than 20MB memory and max file size must be less than 1MB).
- Any malicious activity encountered in the code.

Note: The latest version of the bot would be taken into consideration.
Note: In case of any disputes, the decision of the XOdia team will be final.