DetectWins

When a player places three of his or her marks in a horizontal, vertical, or diagonal line, the player wins;

EnforceTurns

To play, one player marks a square in a 3 by 3 grid with X, then the other player marks a square with O, then it is X's turn again, and so on;

SquareTaken

Once a square is marked, it cannot be marked again;

DefaultOMoves

When other tactics are not applicable, player O should prefer the center square, then the corners, and mark an edge square only when there is no other choice;

StartOAtCenter

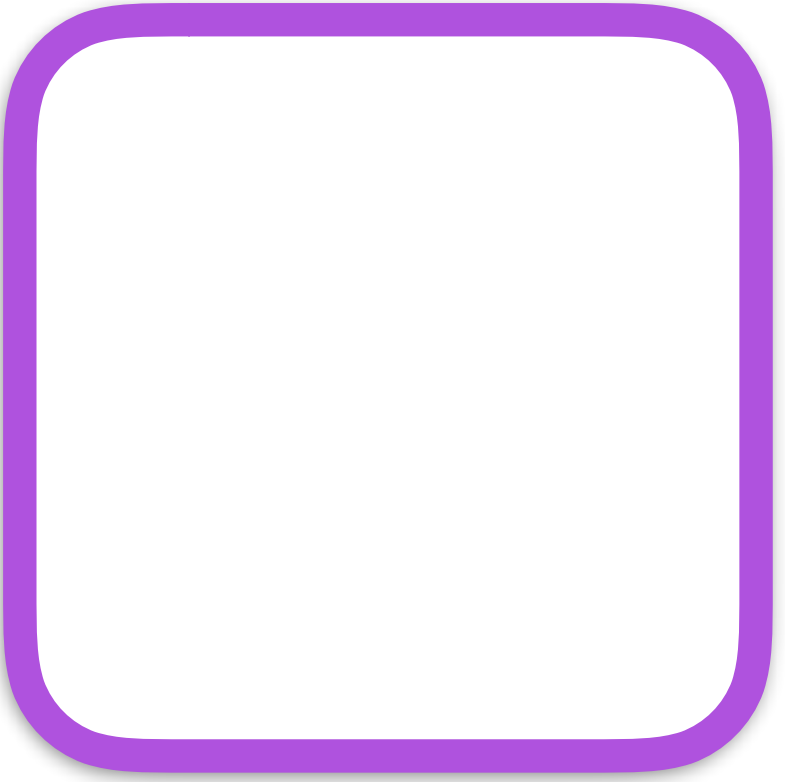
O should start playing at the center;

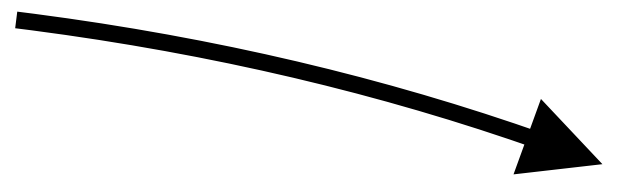
PreventThirdX

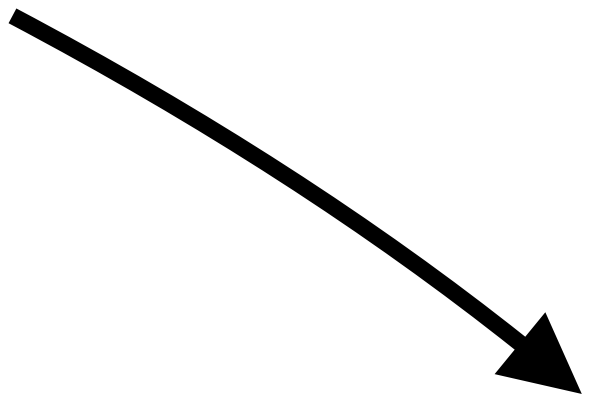
After the X player marks two squares in a line, the O player should try to mark the third square (to foil the attack);

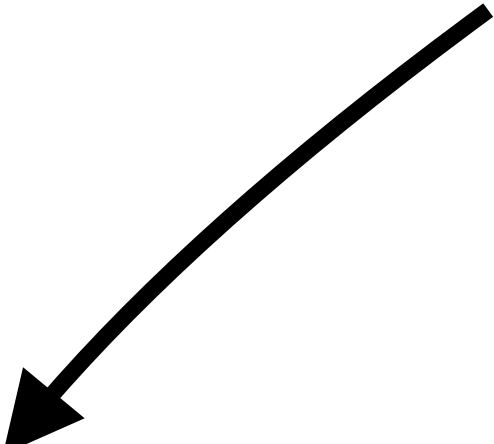
“requirements world”

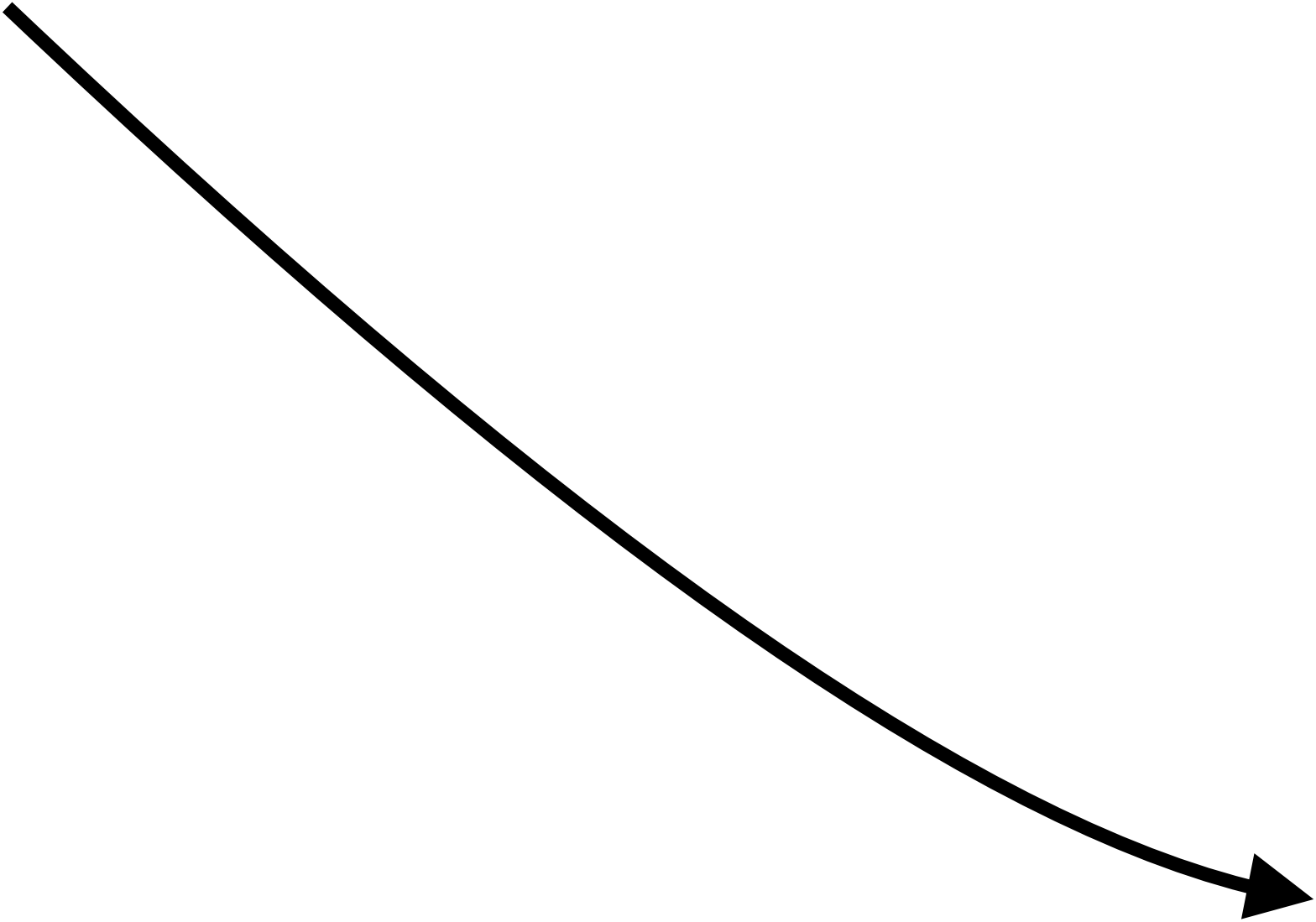
“implementation world”

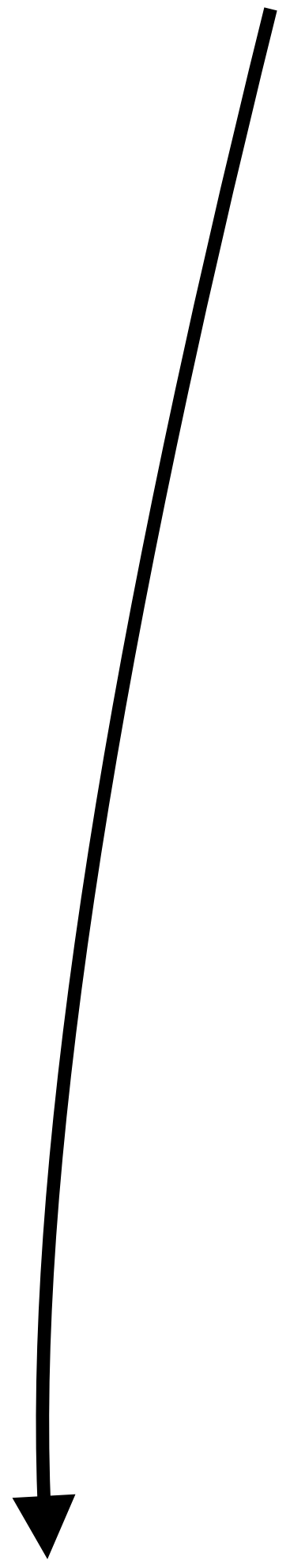


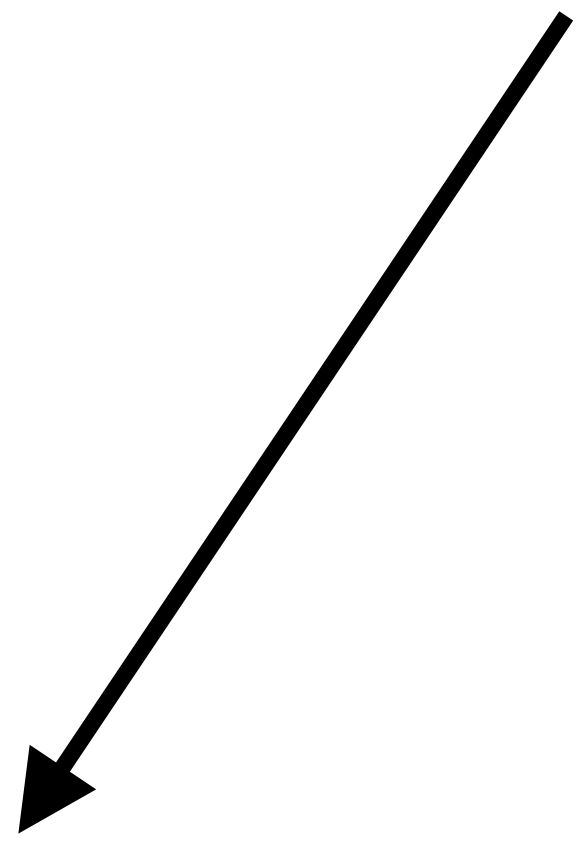


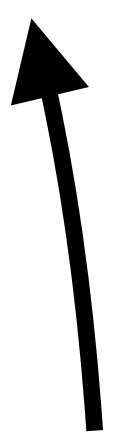


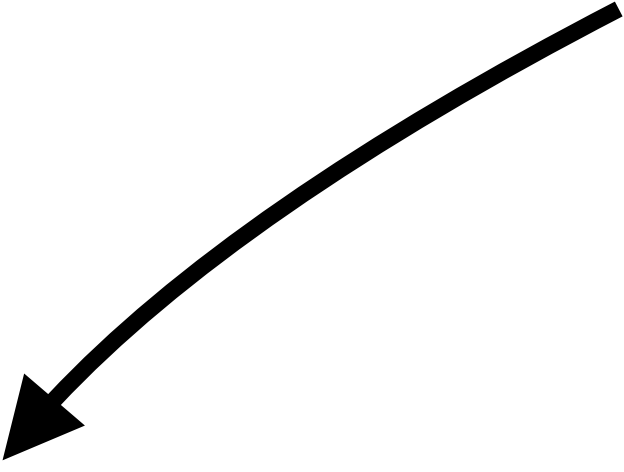


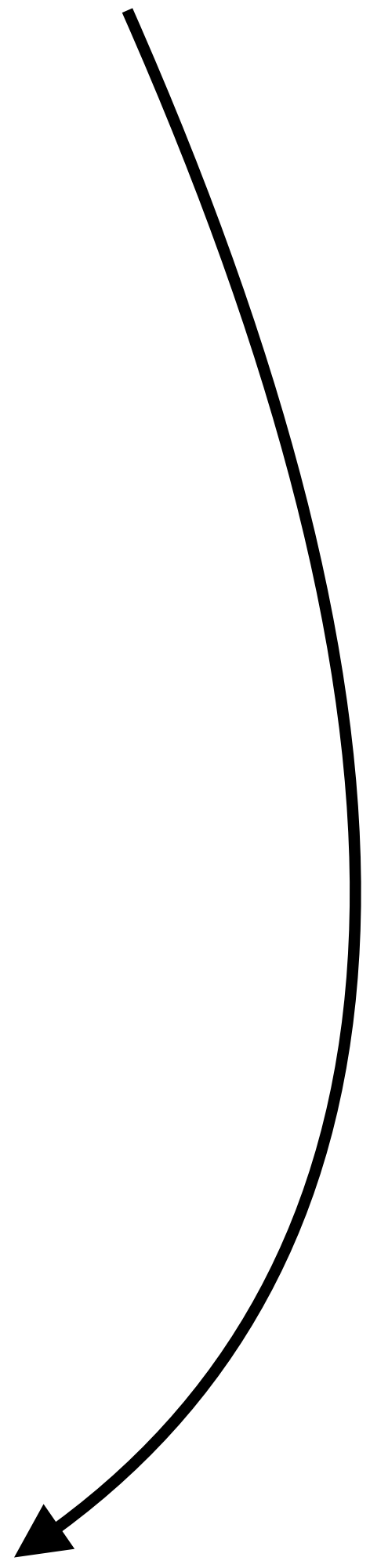


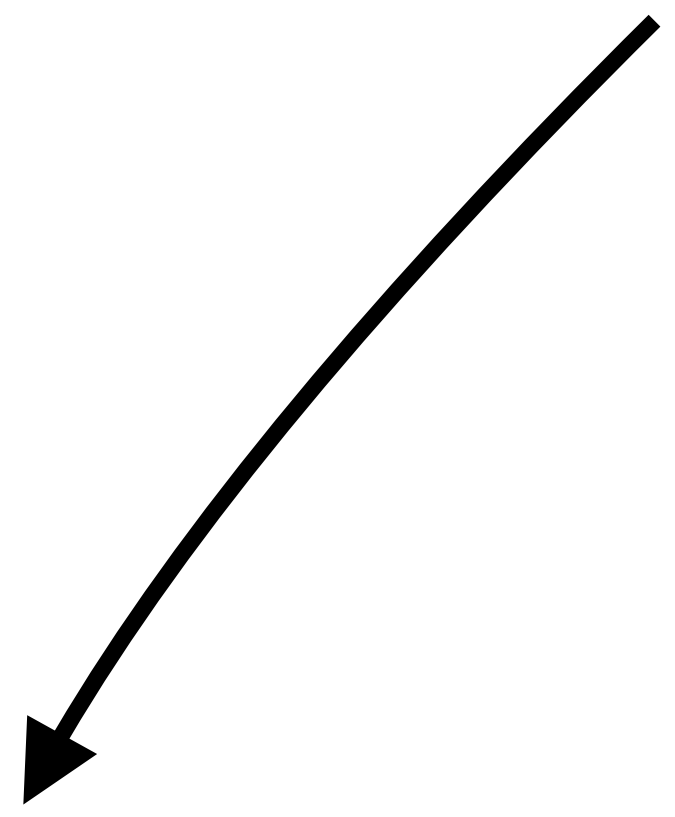


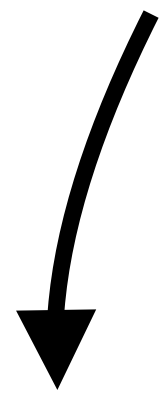


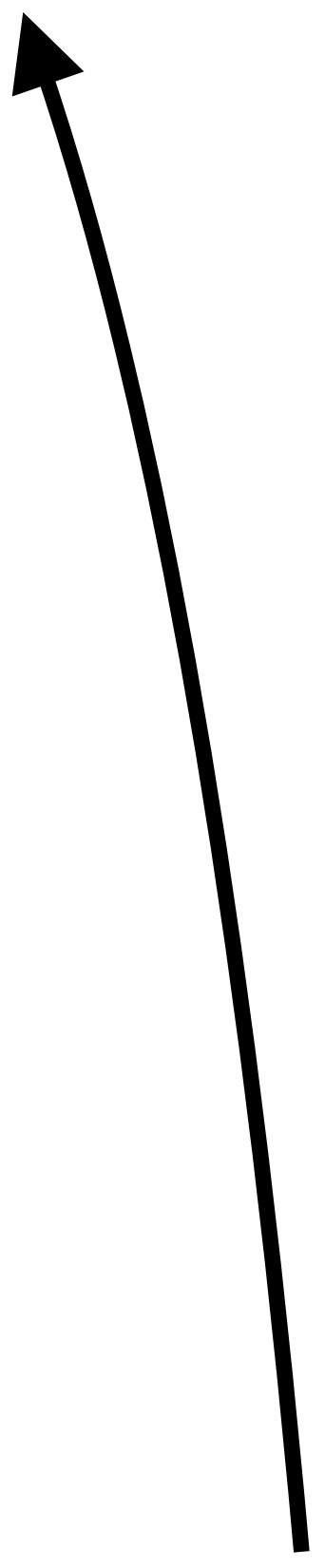


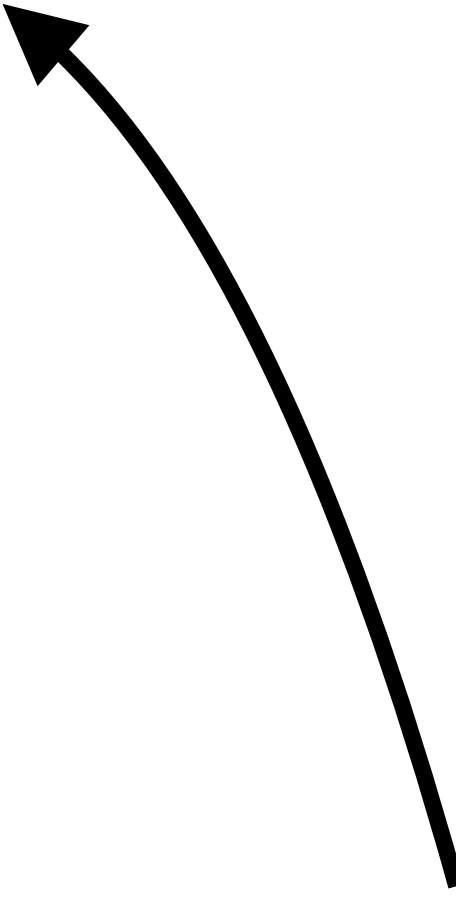


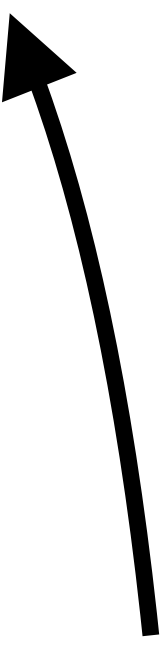


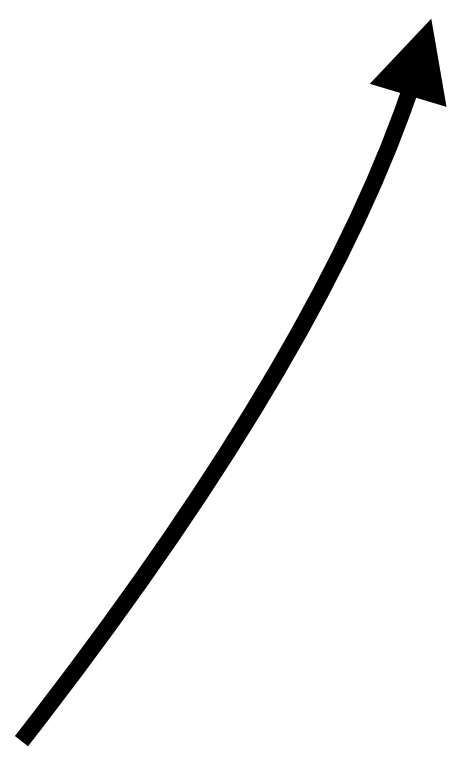


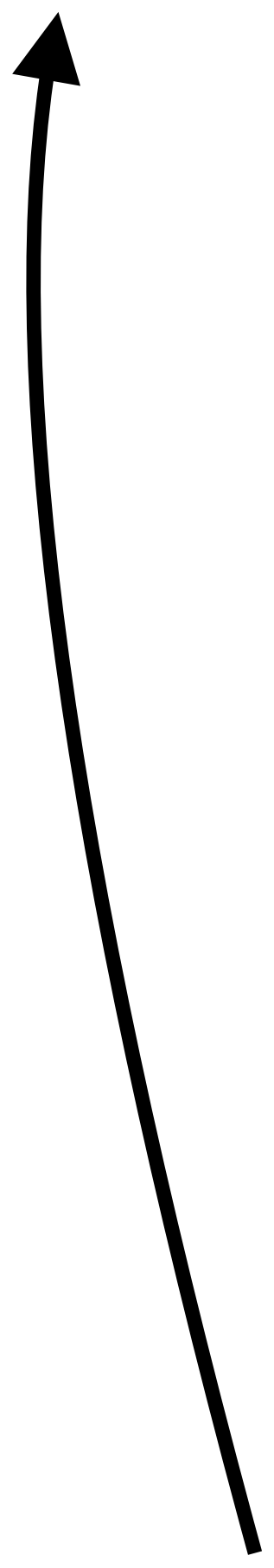


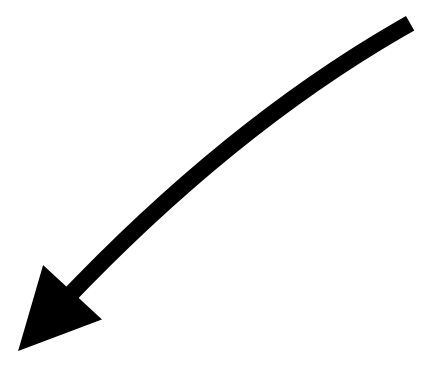


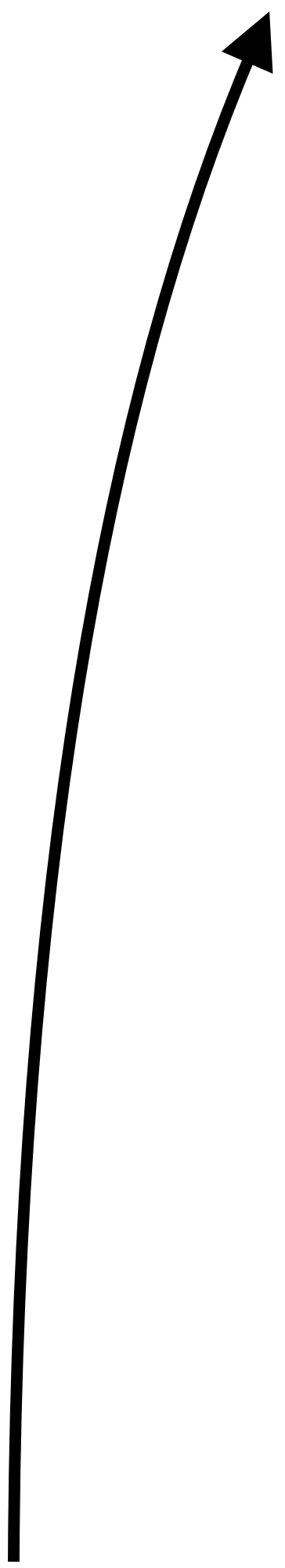


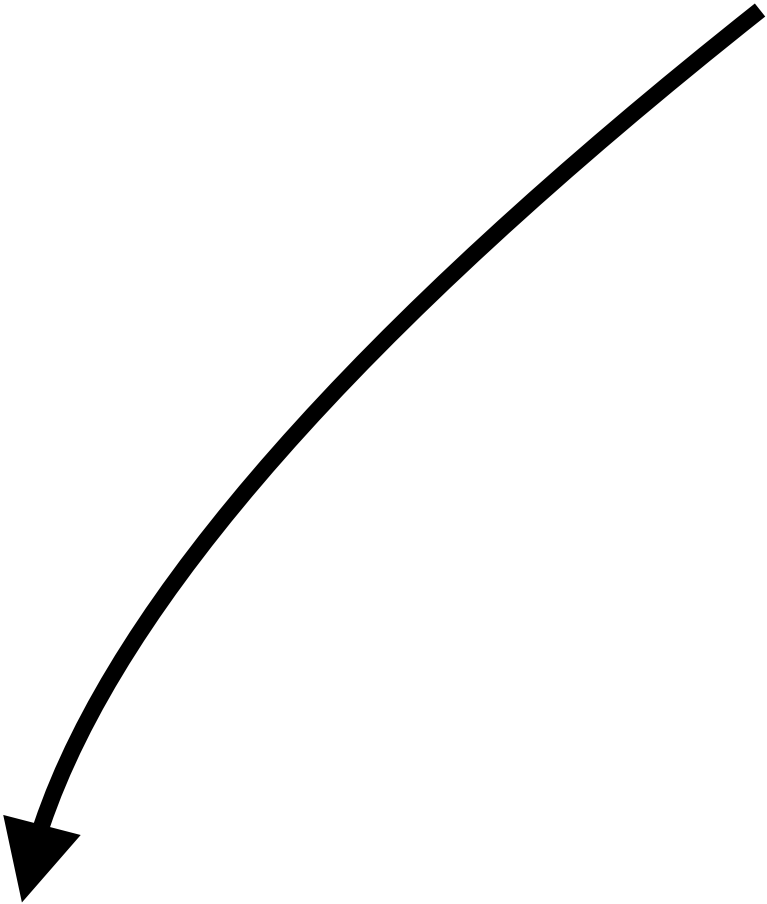


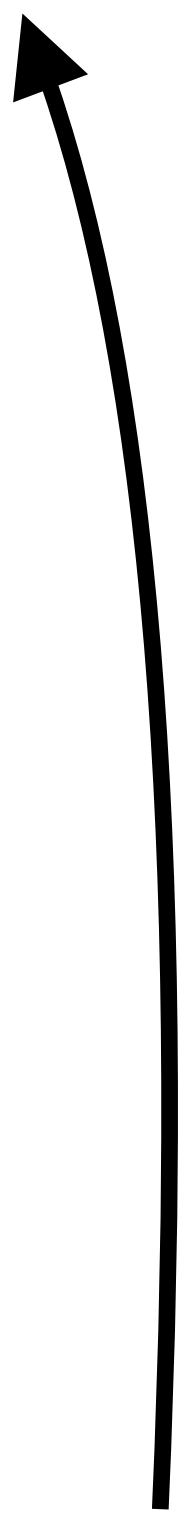


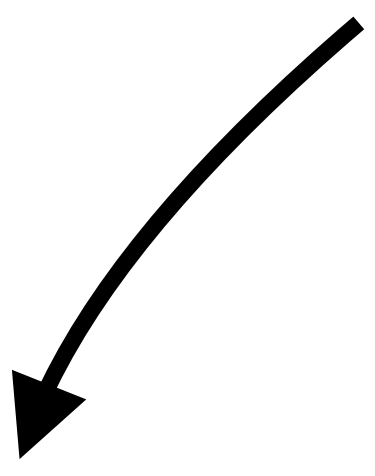


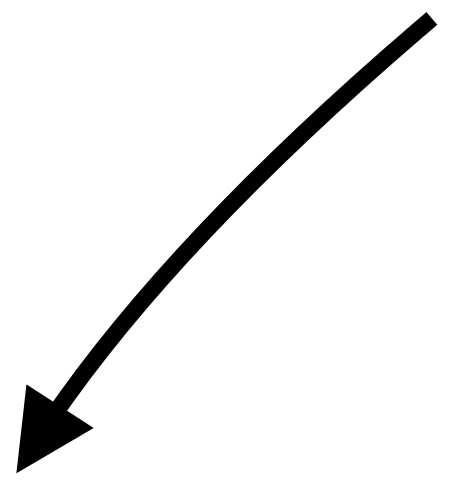


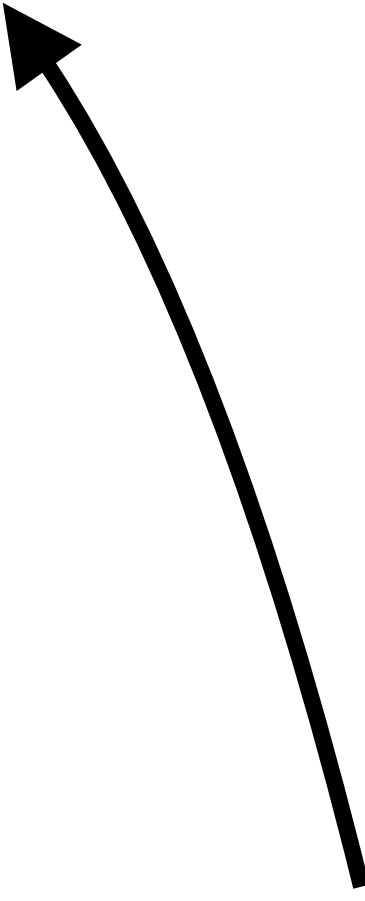












StopGameAfterWin

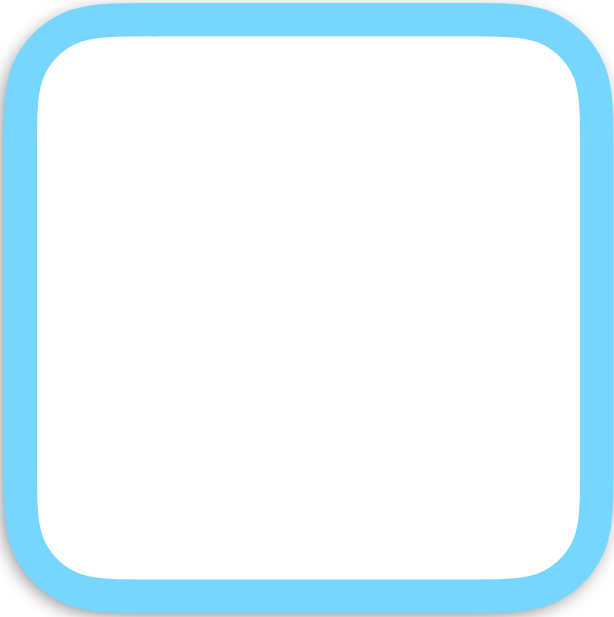
After a win is declared by either X or O stop the game.

Append only

New requirements are ‘piled-atop’ it, with no component specific interface, connectivity, or ordering requirements; refining previously stated sentences. Without even seeing old requirements.

AddThirdO

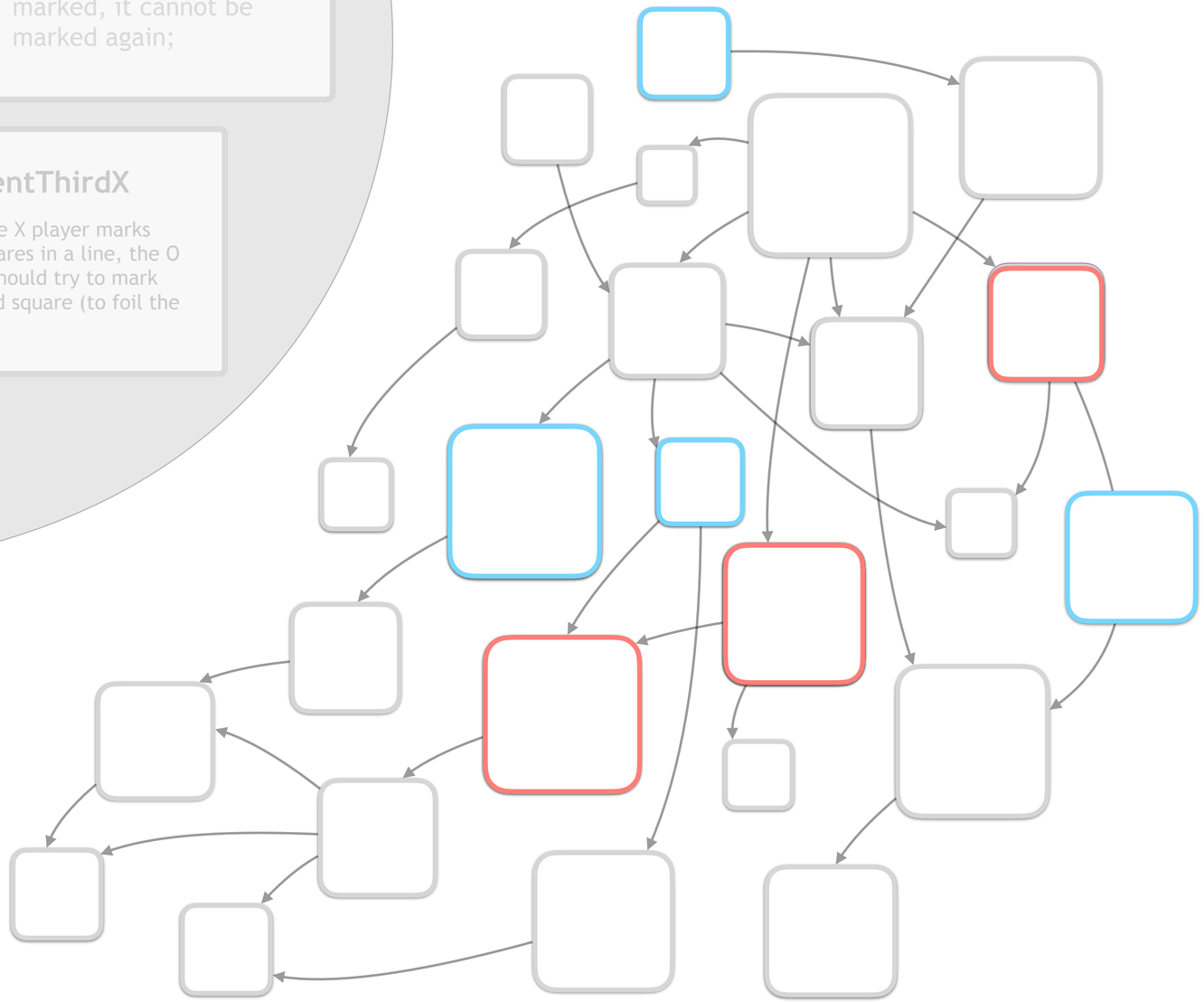
After placing two O marks in a line, the O player should try to mark the third square (to win the game);



Tic Tac Toe

Tic Tac Toe

“implementation world”



“requirements world”

DetectWins

When a [illegible] his or h[illegible] vertical[illegible] player [illegible]

EnforceTurns

To play, one player marks a square in [illegible]X, then the other [illegible]O, then it

SquareTaken

Once a square is marked, it cannot be marked again;

Sta[illegible]

Aft[illegible] eith[illegible] gam[illegible]

AddThirdO

After placing two O marks in a line, the O player should try to mark the third square (to win the game);

Defa[illegible]

When other tactic[illegible] applicable, player[illegible] the center square[illegible] corners, and mark[illegible] only when there is no other choice;

PreventThirdX

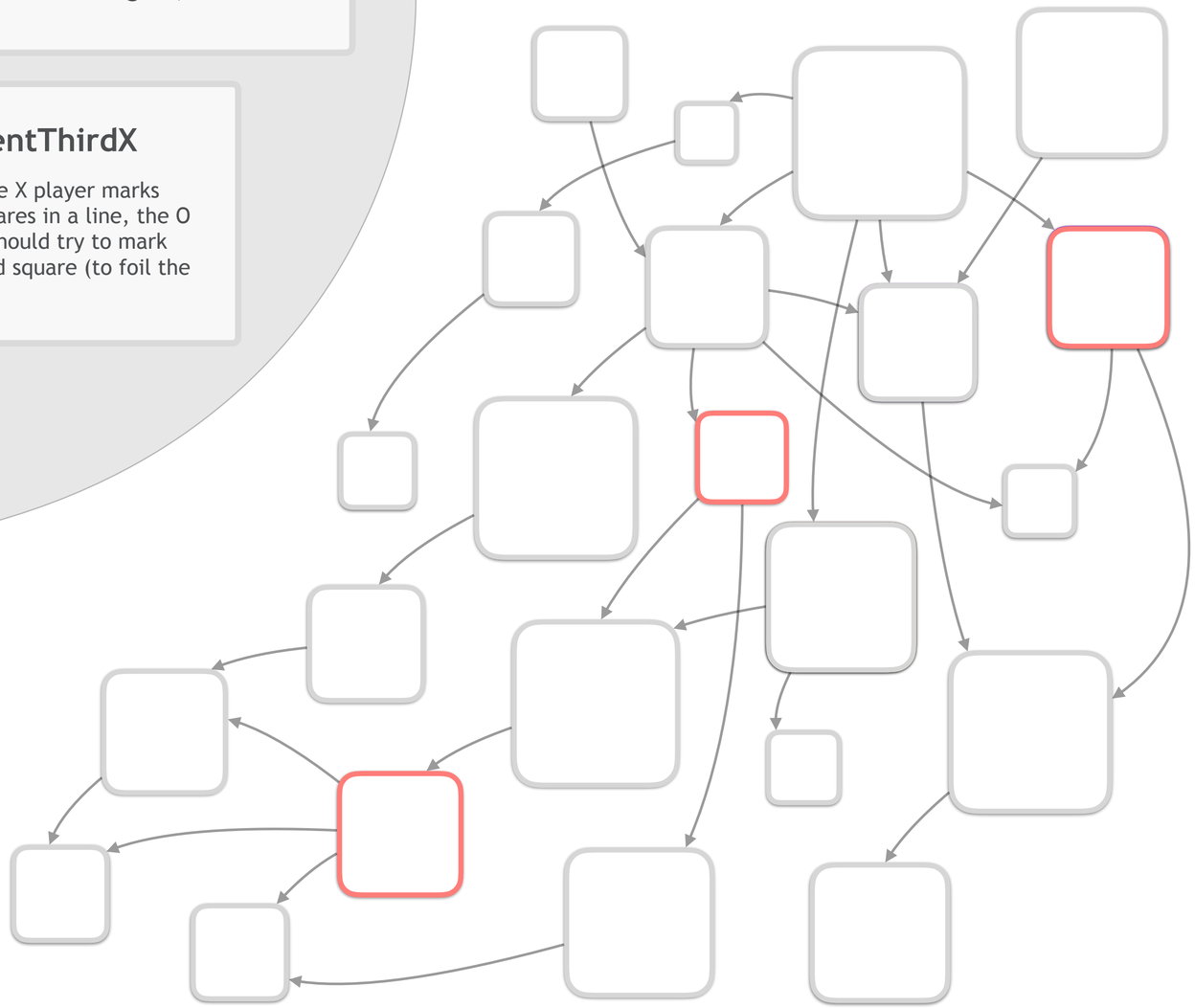
After the X player marks two squares in a line, the O player should try to mark the third square (to foil the attack);

Append only

New requirements are ‘piled-atop’ it, with no component specific interface, connectivity, or ordering requirements; refining previously stated sentences. Without even seeing old requirements.

Tic Tac Toe

“implementation world”



DetectWins

When a player places three of his or her marks in a horizontal, vertical, or diagonal line, the player wins;

EnforceTurns

To play, one player marks a square in a 3 by 3 grid with X, then the other

SquareTaken

Once a square is marked, it cannot be marked again;

StopGameAfterWin

After a win is declared by either X or O stop the game.

DefaultOMoves

When other tactics are not applicable, player O should prefer the center square, then the corners, and mark an edge square only when there is no other choice;

at the center;

PreventThirdX

After the X player marks two squares in a line, the O player should try to mark the third square (to foil the attack);

“requirements world”

Upfront negativity

How are the boxes in red built if they have nothing to connect them to? Imagine the learning potential and creative power of a human who is allowed to freely experiment with a variety of behaviors, except those that are forbidden (e.g., the illegal, expensive, or risky ones), figuring out if and when any of allowed actions produces valuable results