

Computer Science and Communications Engineering Laboratory A Algorithms and Data Structures

Shinichi Honiden
honiden@nii.ac.jp

Objectives of experiments

- Develop the ability to devise judgement algorithms through game creation
- Develop the ability to solve problems by dividing them

Index

1. Details of the game
2. Outline of the experiment
3. Supplements

Details of the game

Connect4

- Two-player board game
- Place the colored pieces alternately
- A player wins when she lined up her four pieces vertically, horizontally or diagonally first



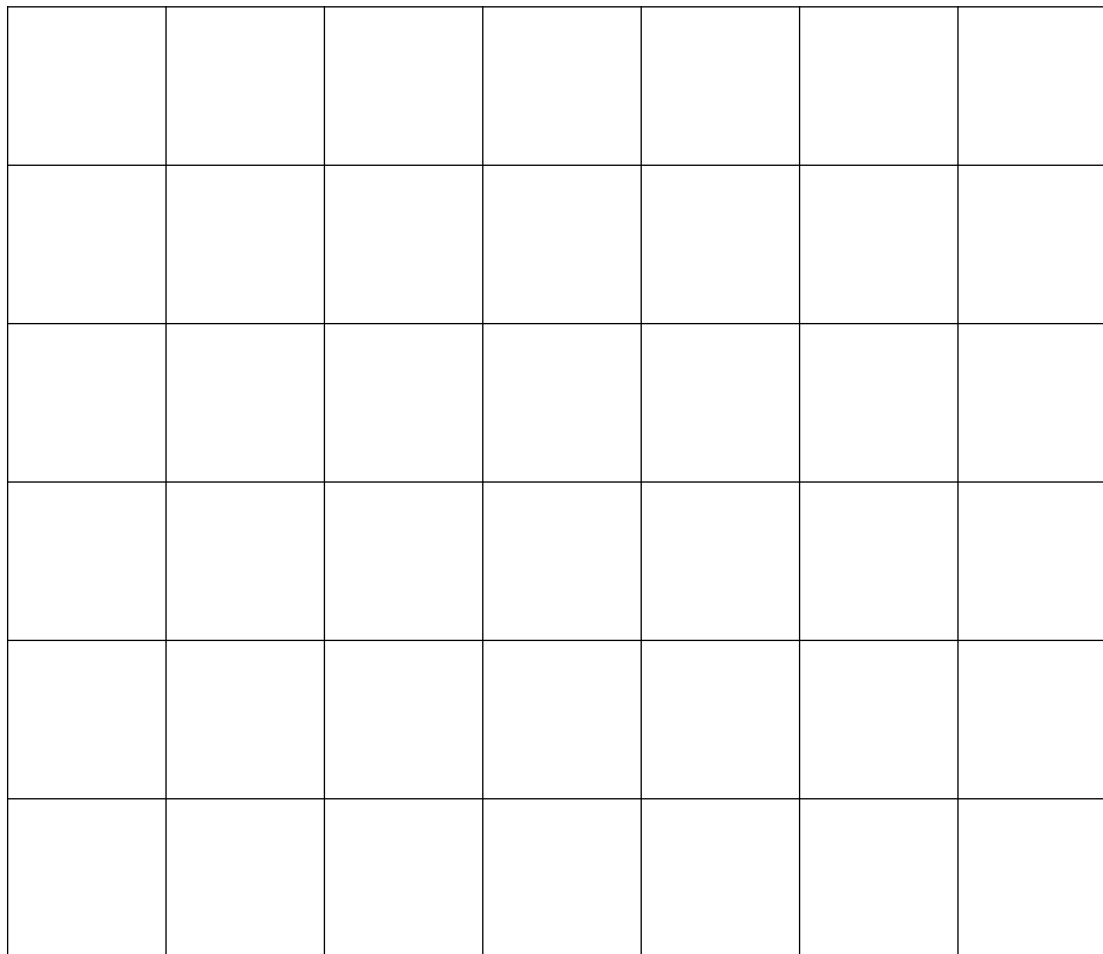
Connect4

- **Connect4** is a two-player connection board game, in which the players choose a color and then take turns dropping colored discs into a seven-column, six-row vertically suspended grid. The pieces fall straight down, occupying the lowest available space within the column. The objective of the game is to be the first to form a horizontal, vertical, or diagonal line of four of one's own discs.

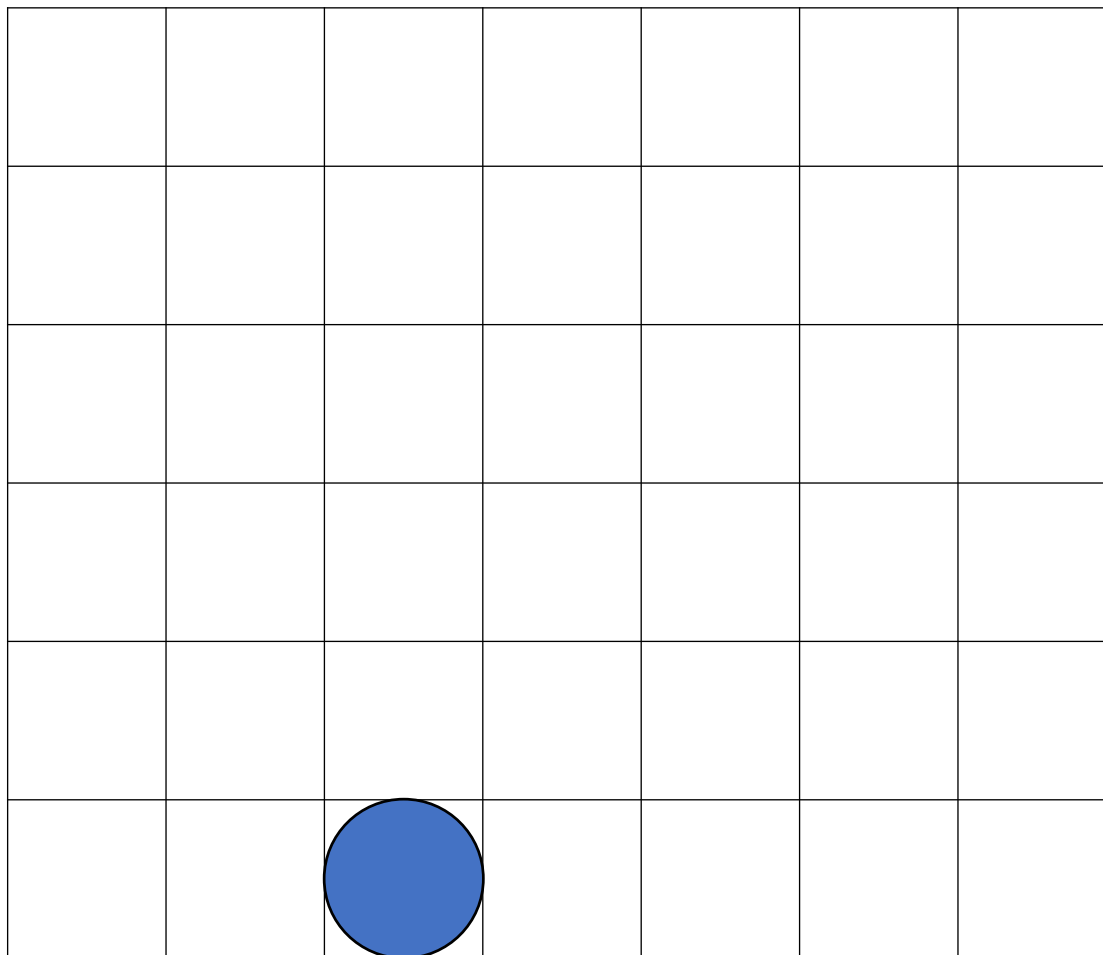
https://en.wikipedia.org/wiki/Connect_Four.



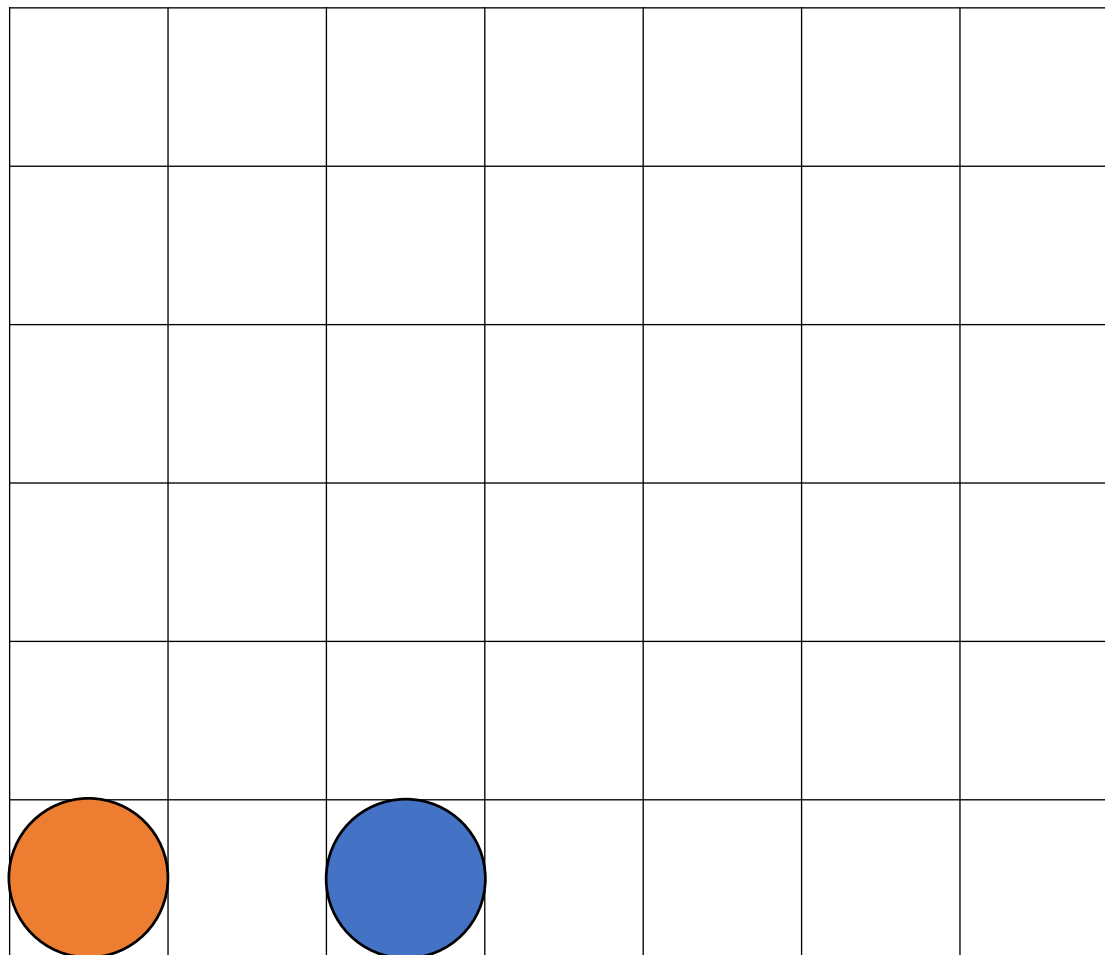
How to play 0/18



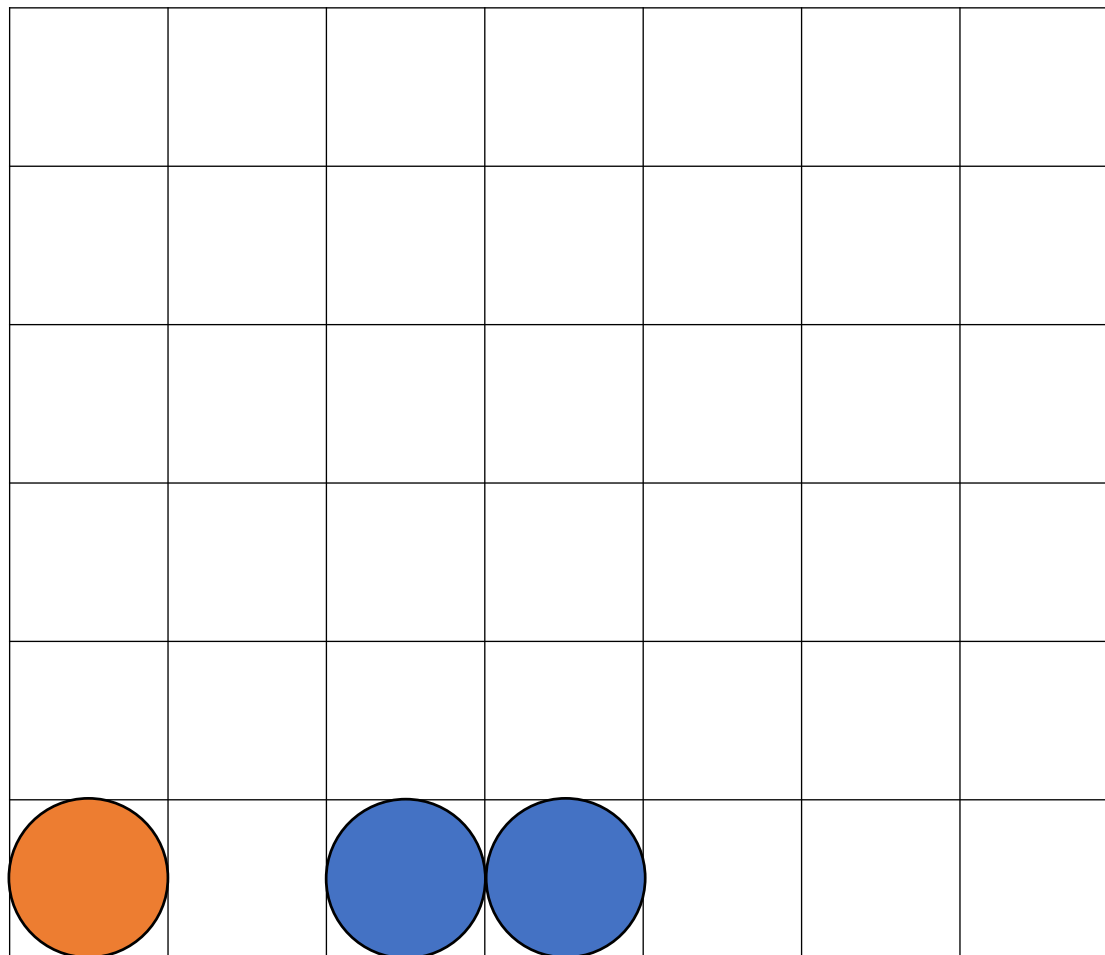
How to play 1/18



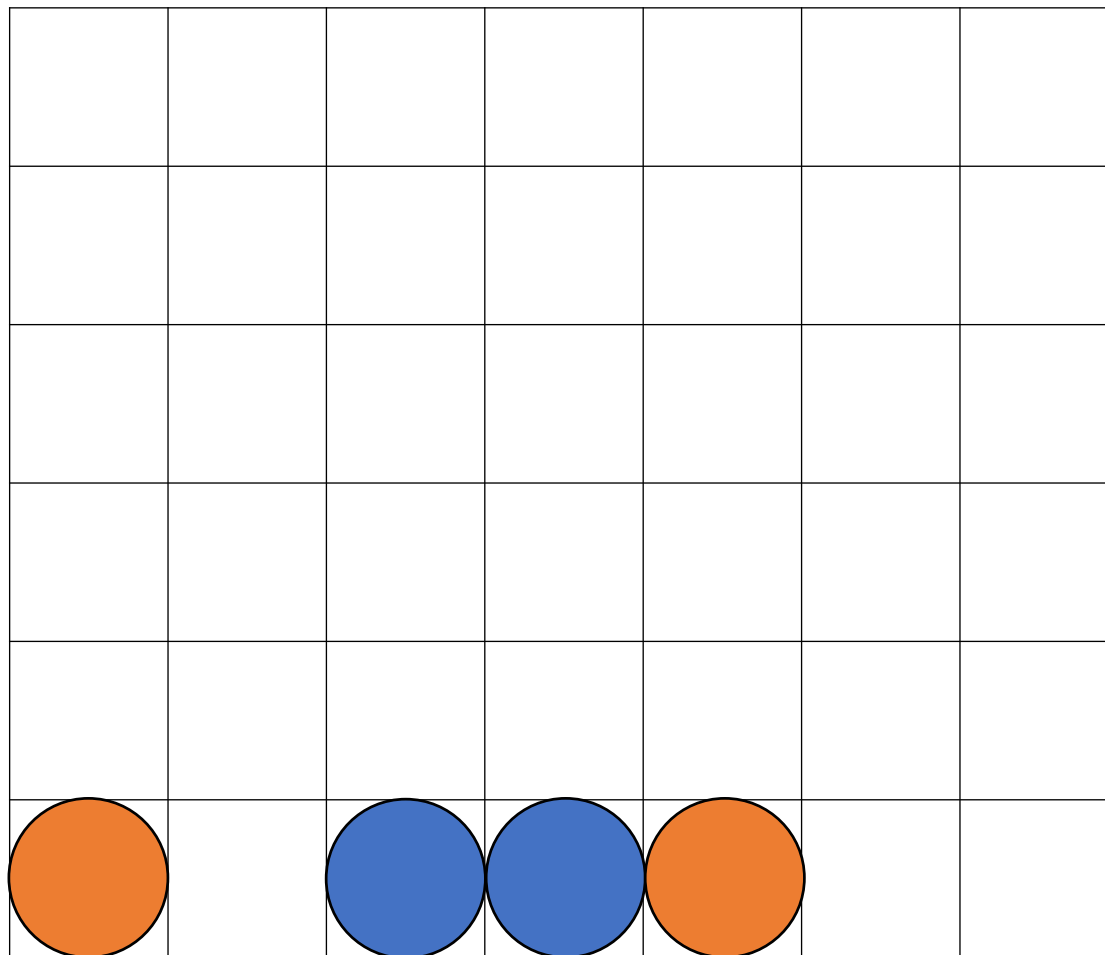
How to play 2/18



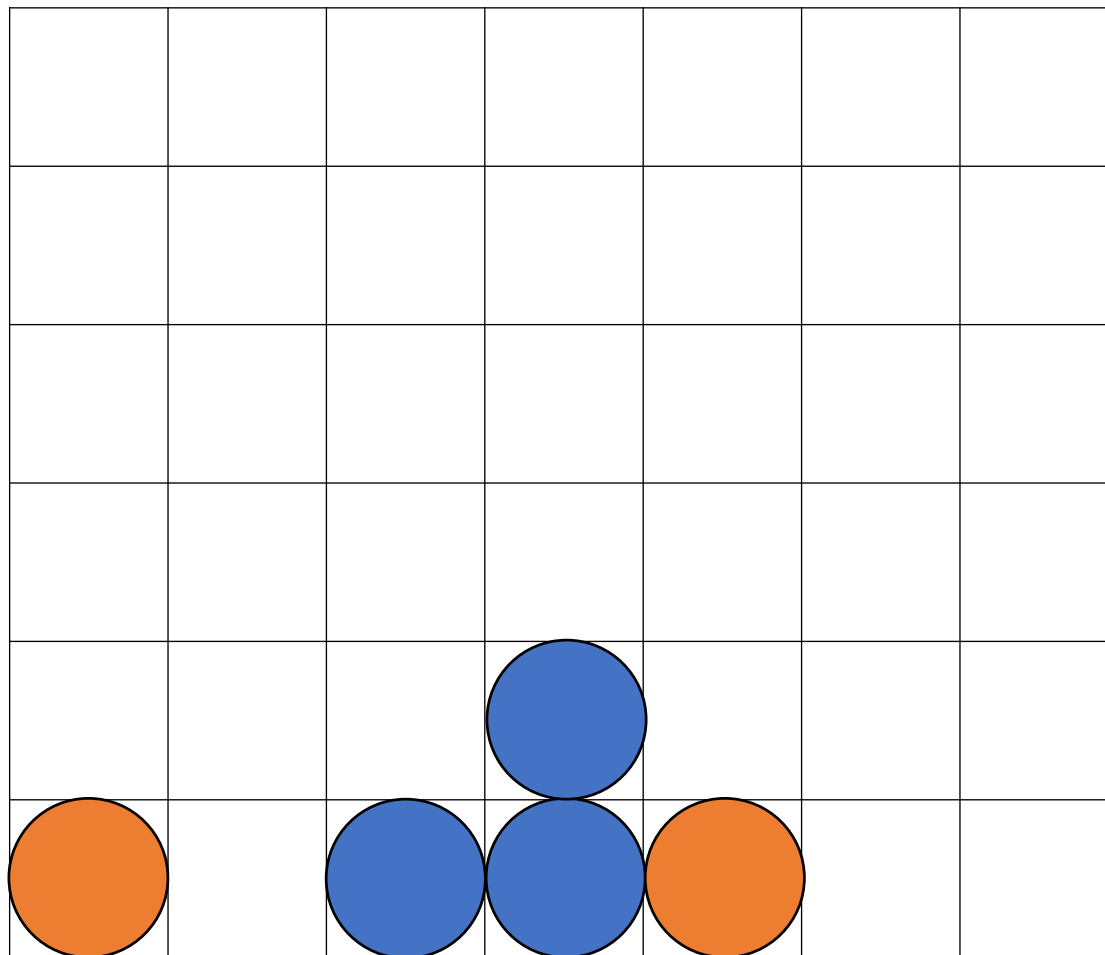
How to play 3/18



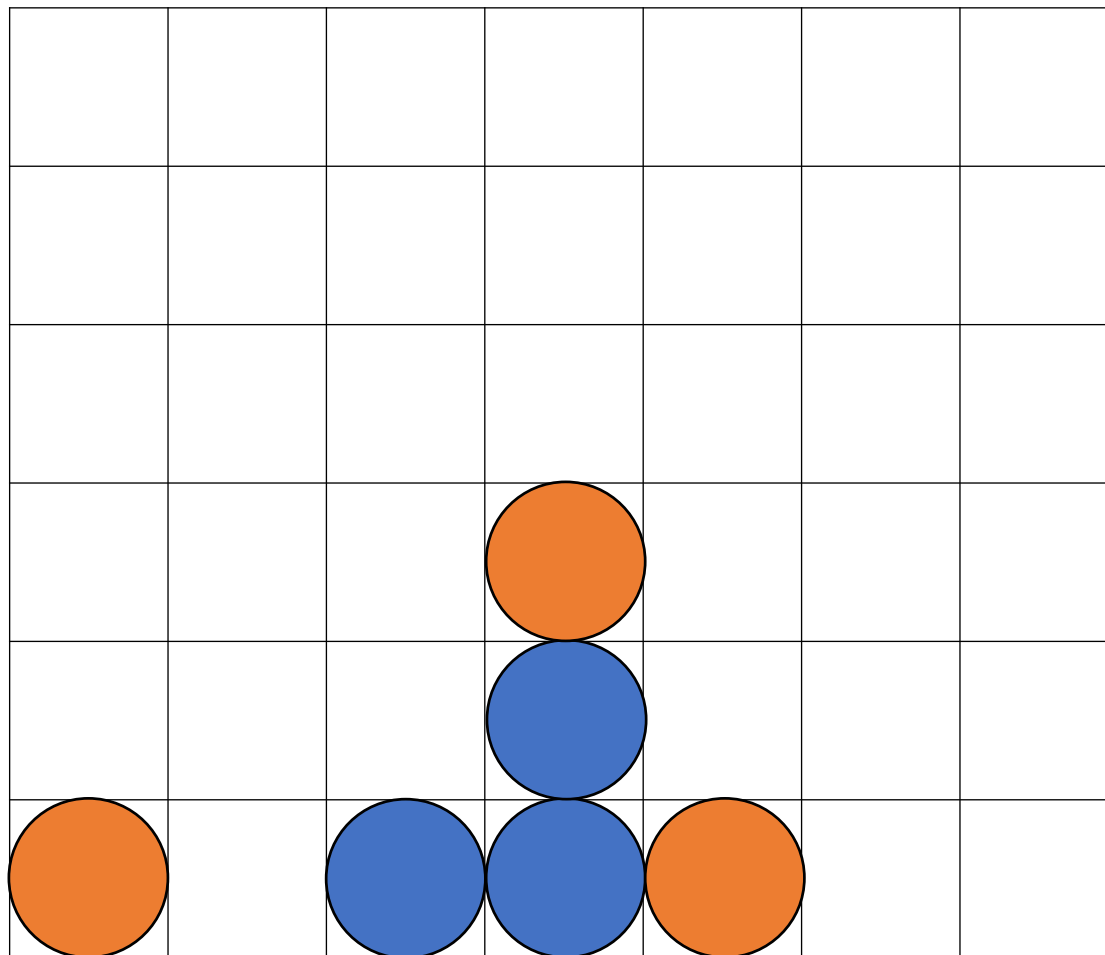
How to play 4/18



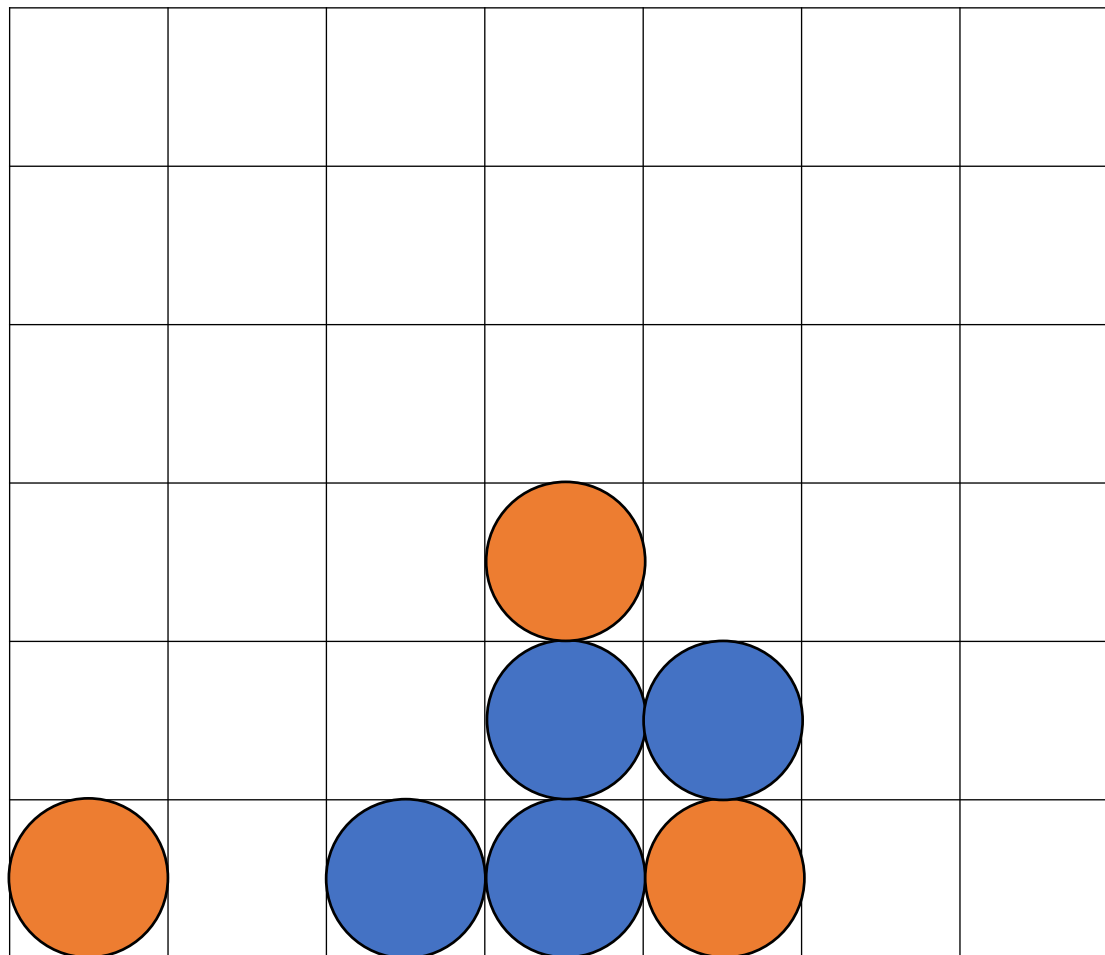
How to play 5/18



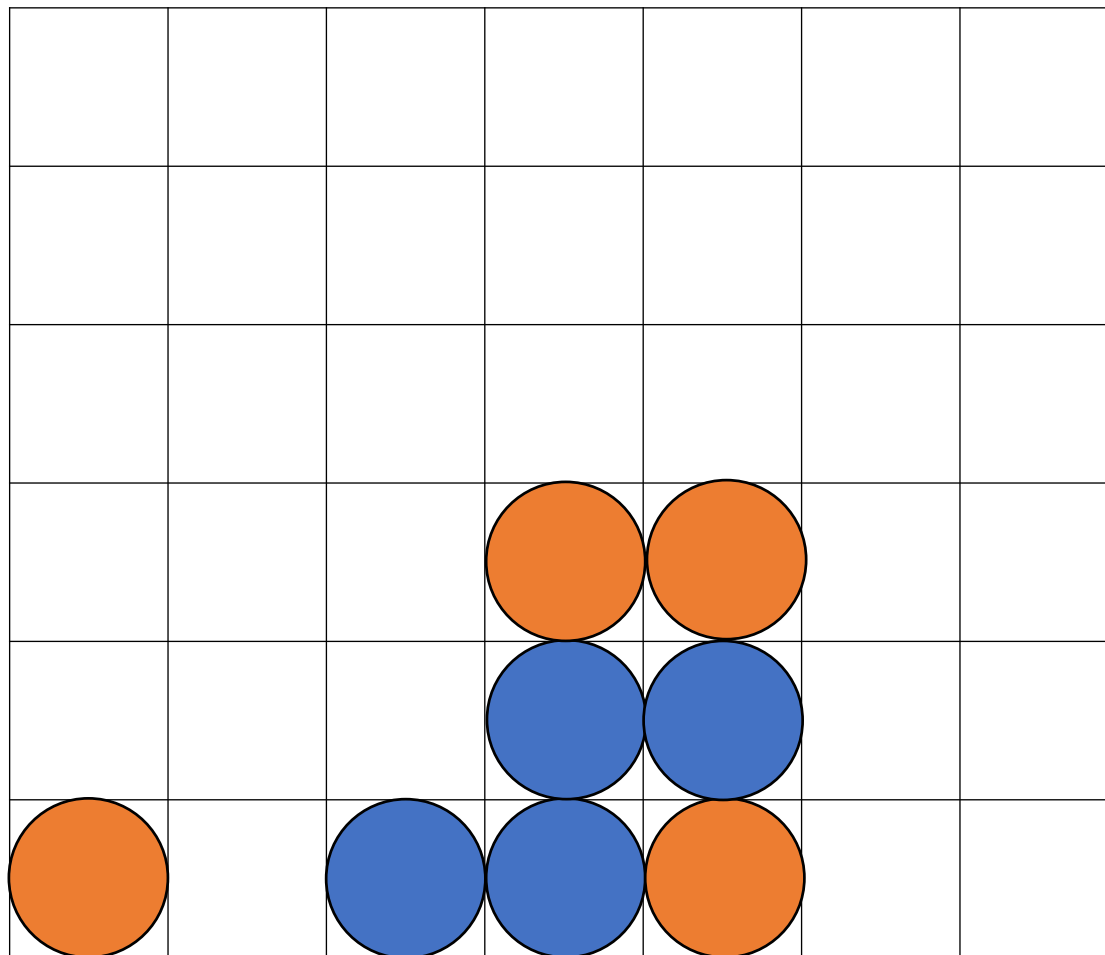
How to play 6/18



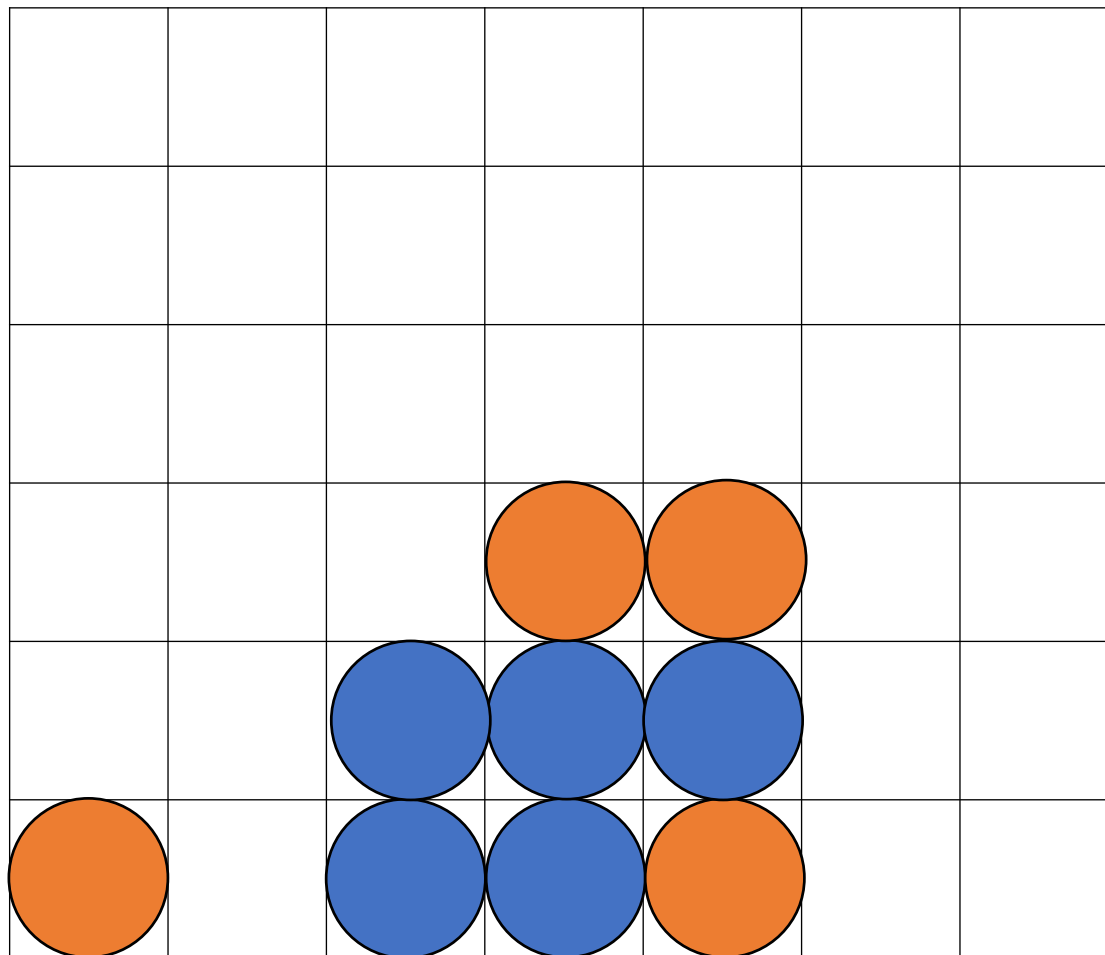
How to play 7/18



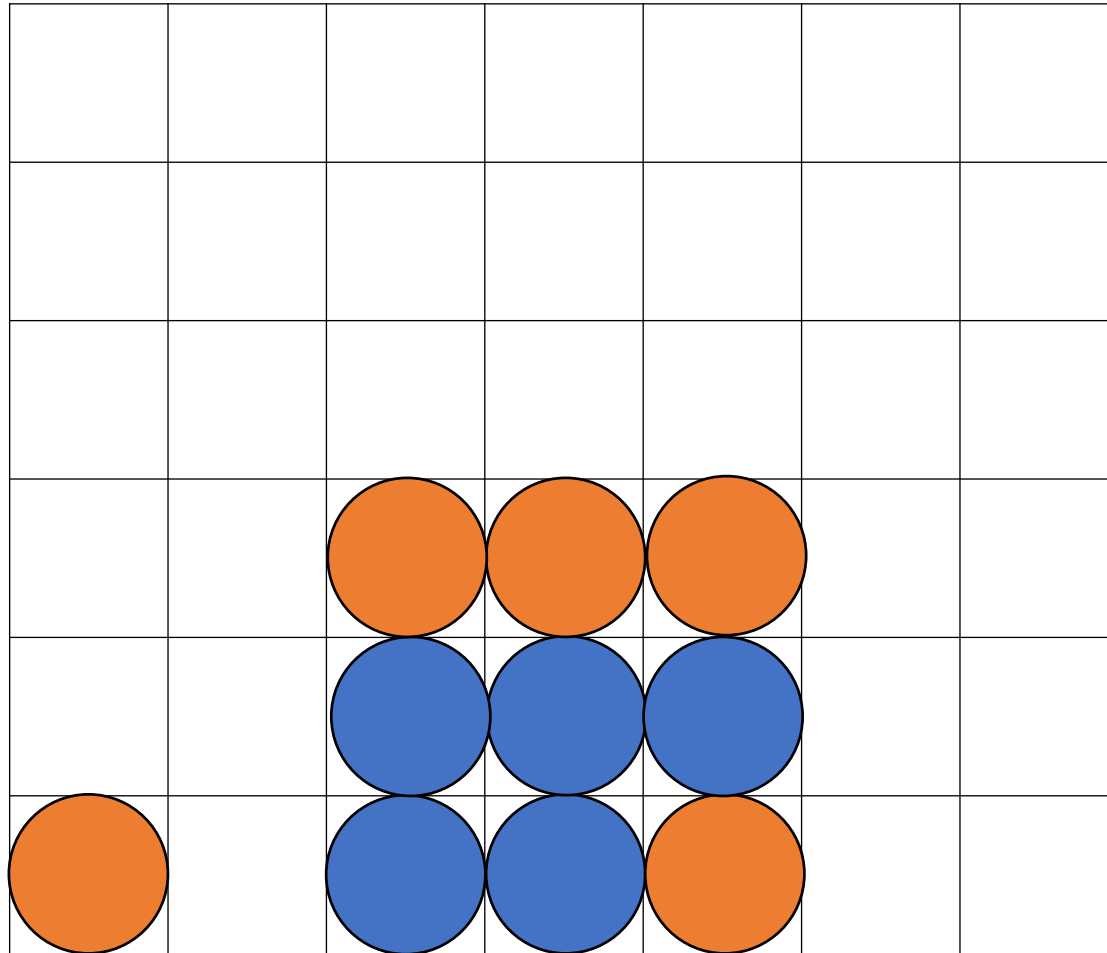
How to play 8/18



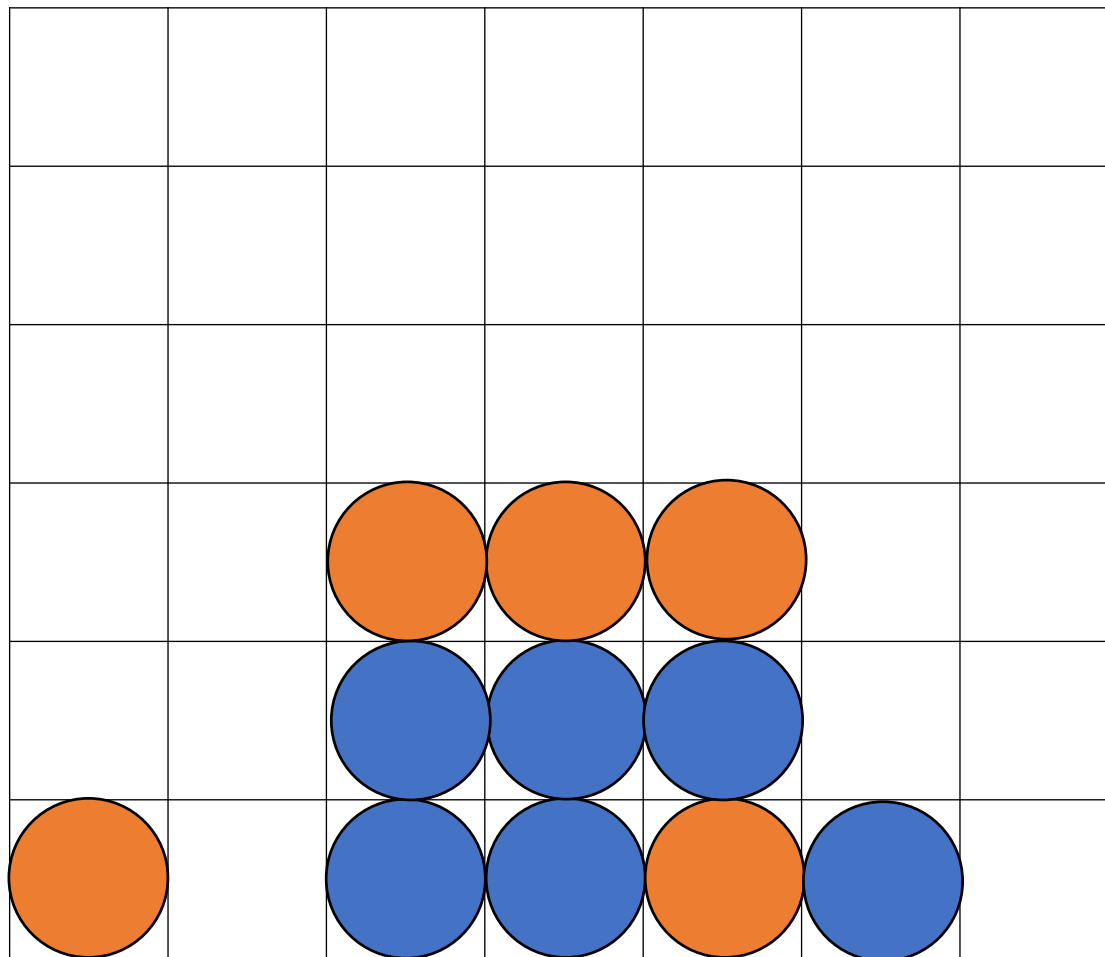
How to play 9/18



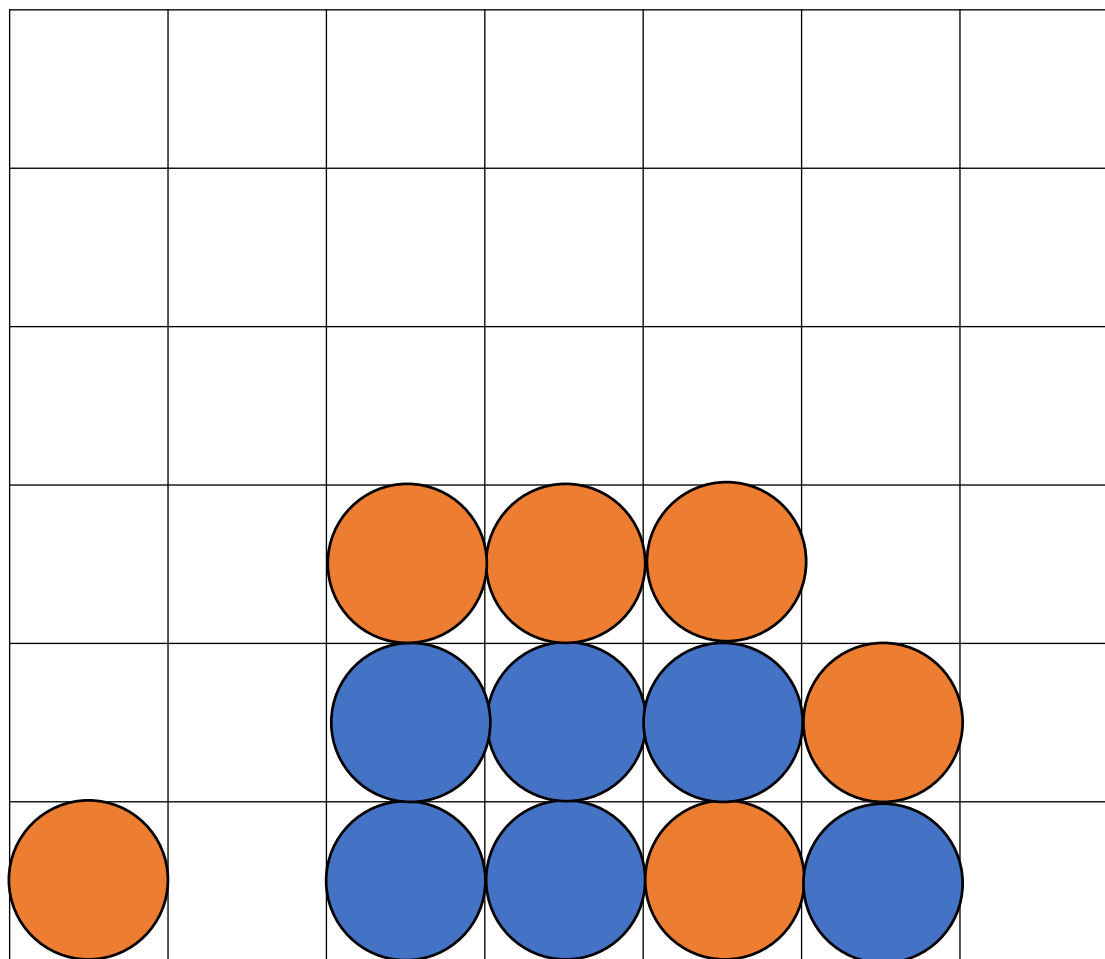
How to play 10/18



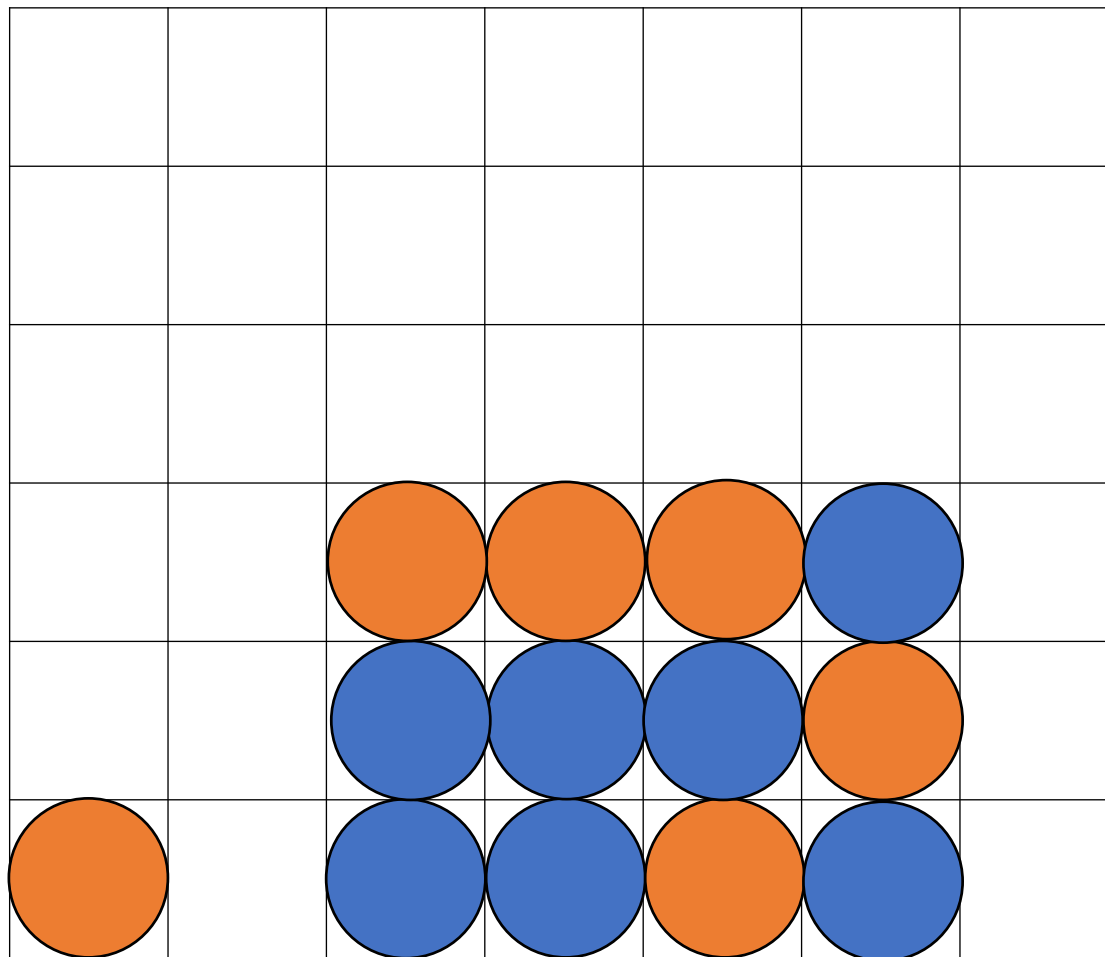
How to play 11/18



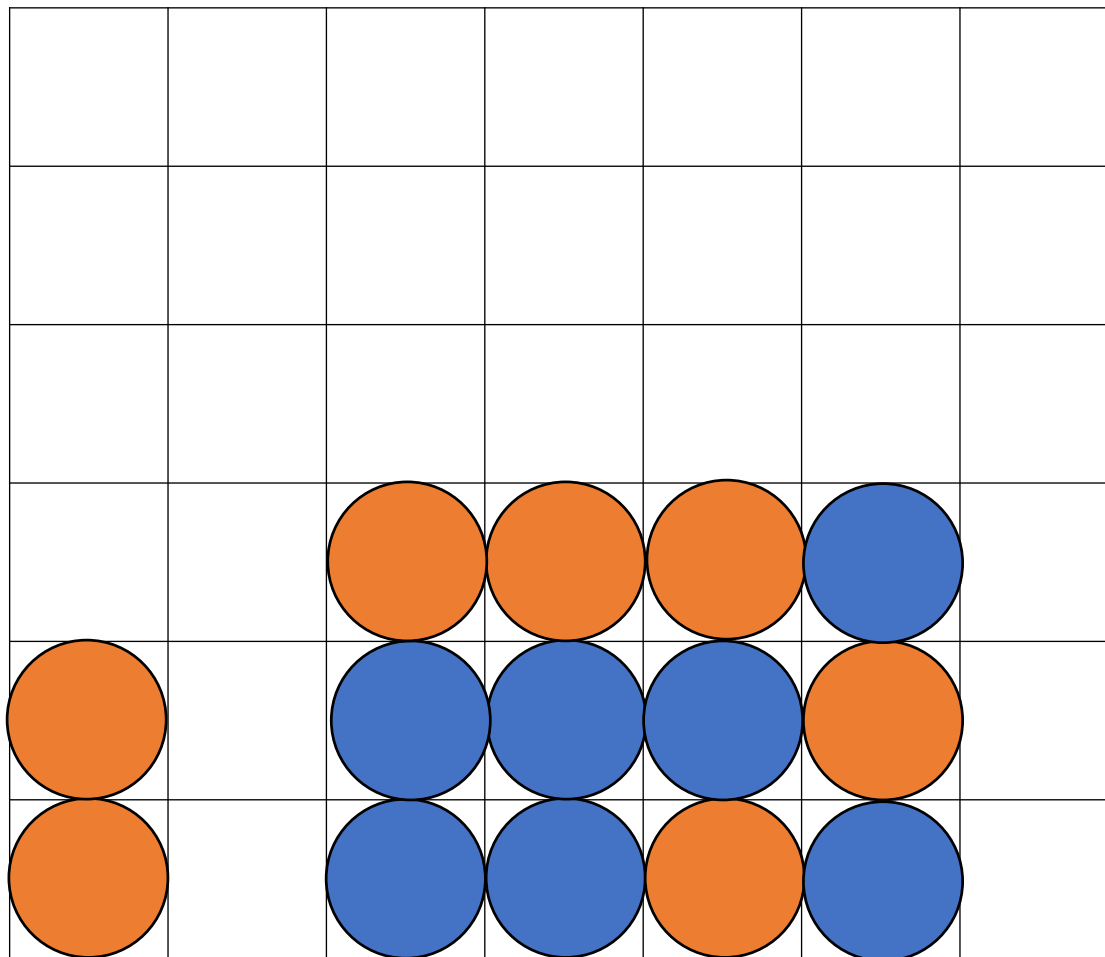
How to play 12/18



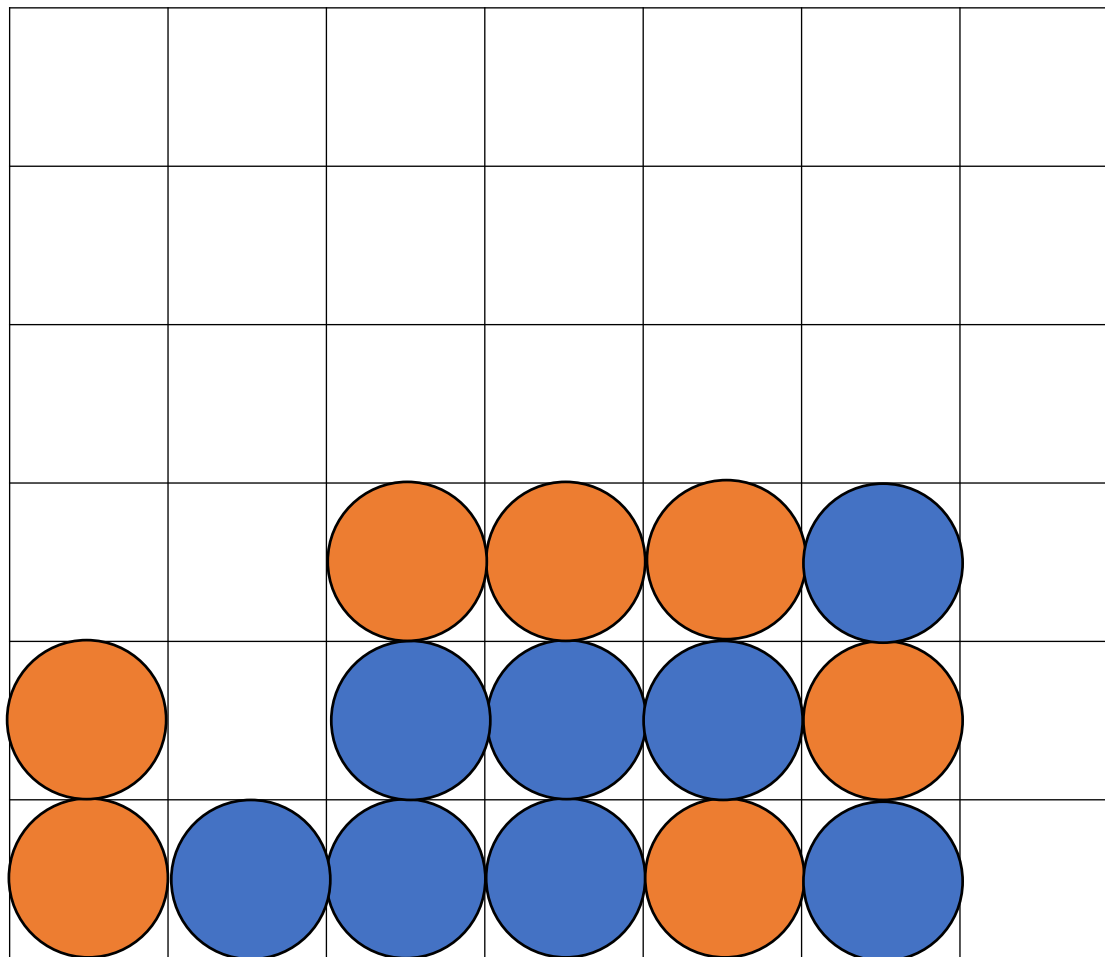
How to play 13/18



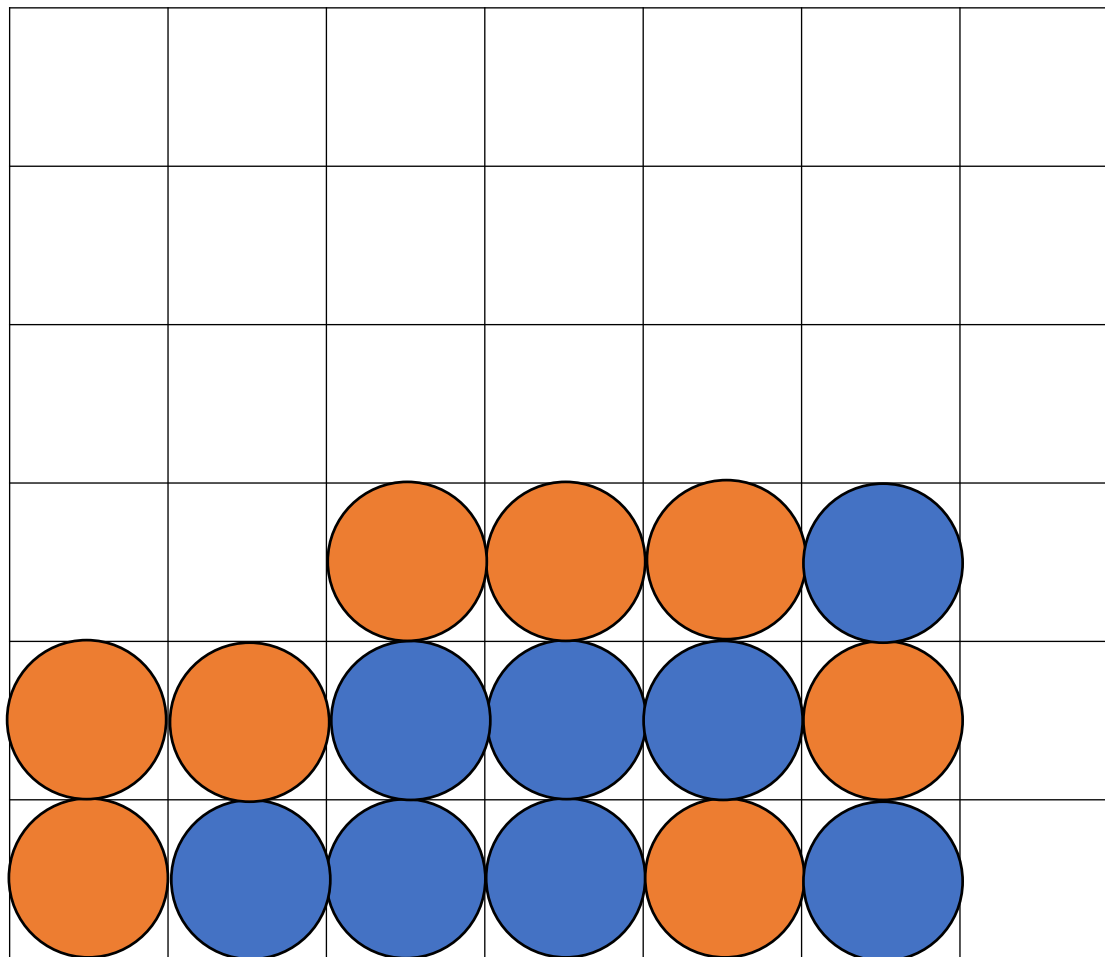
How to play 14/18



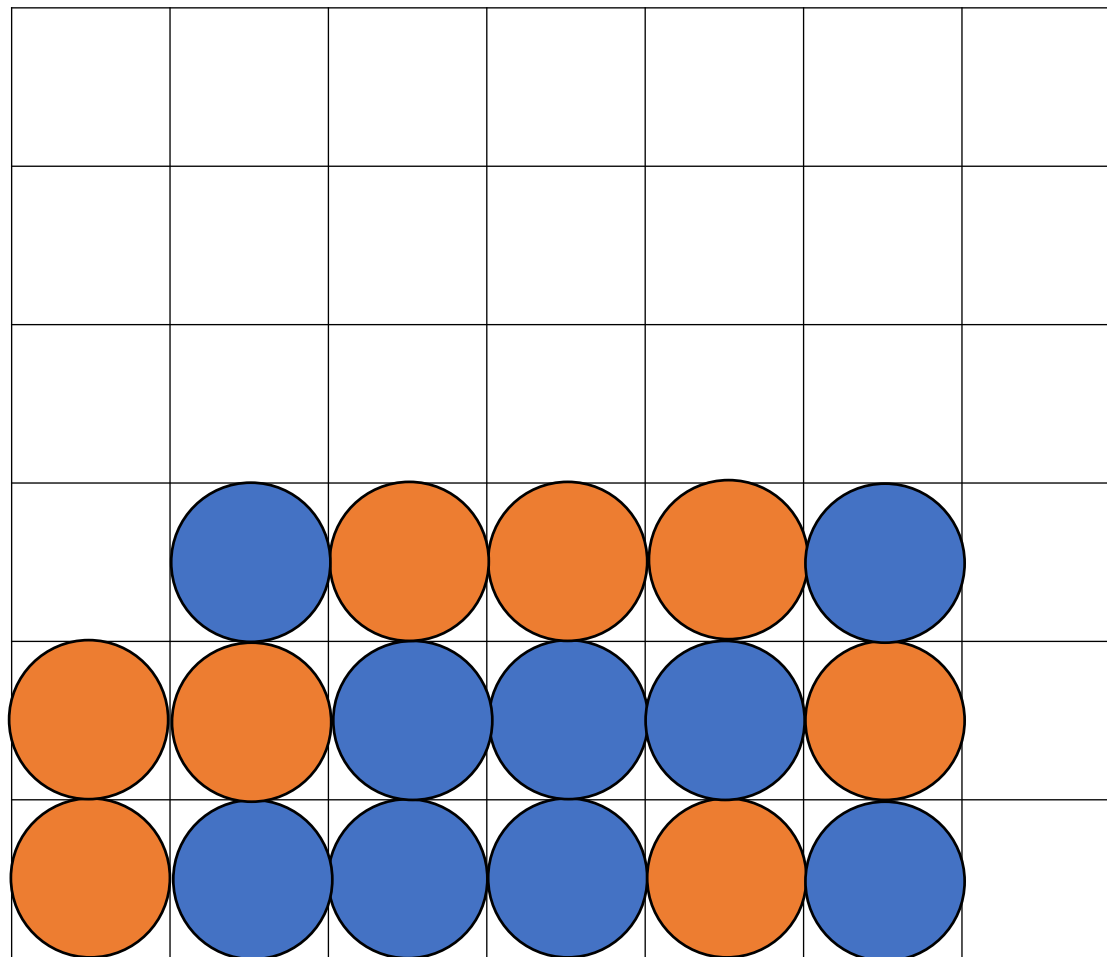
How to play 15/18



How to play 16/18

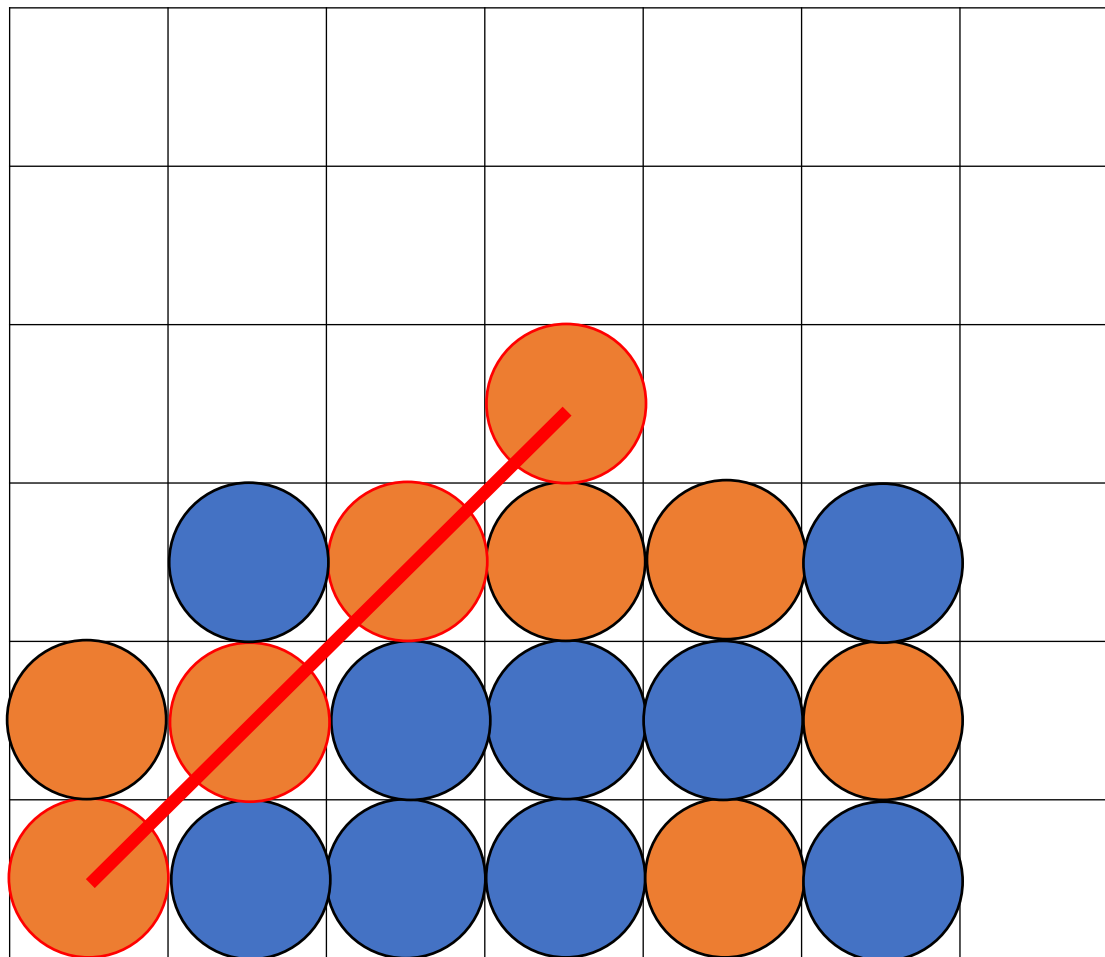


How to play 17/18



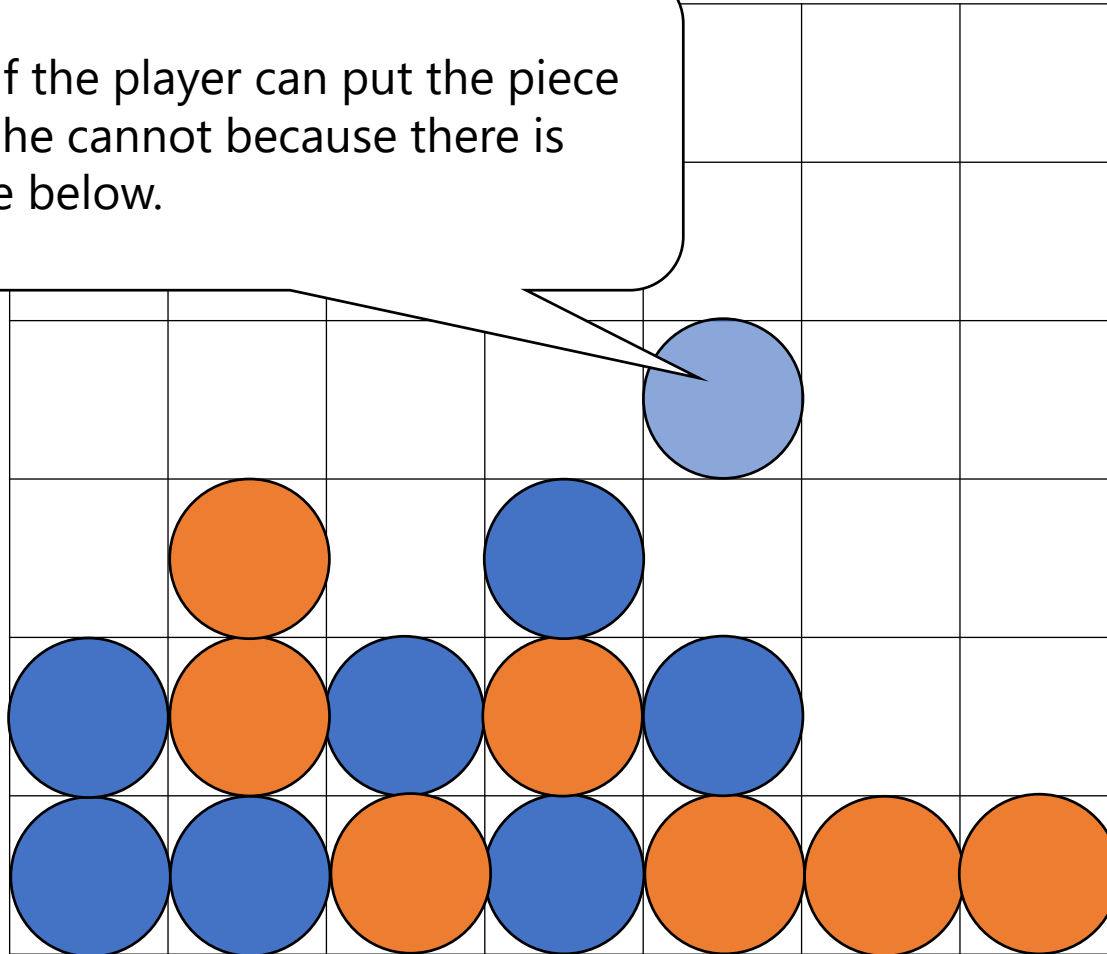
How to play 18/18

 win !



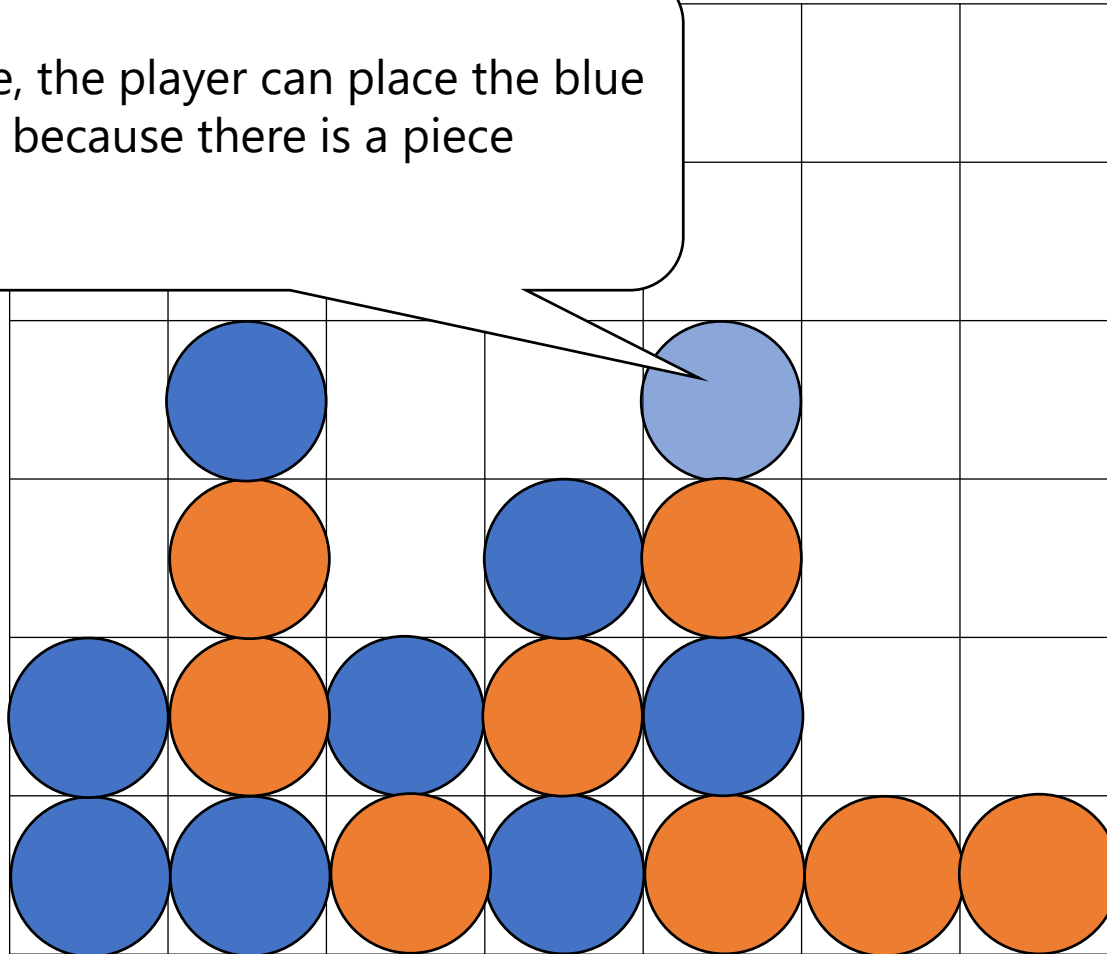
Remark on placing a piece

Blue wins if the player can put the piece here, but she cannot because there is not a piece below.



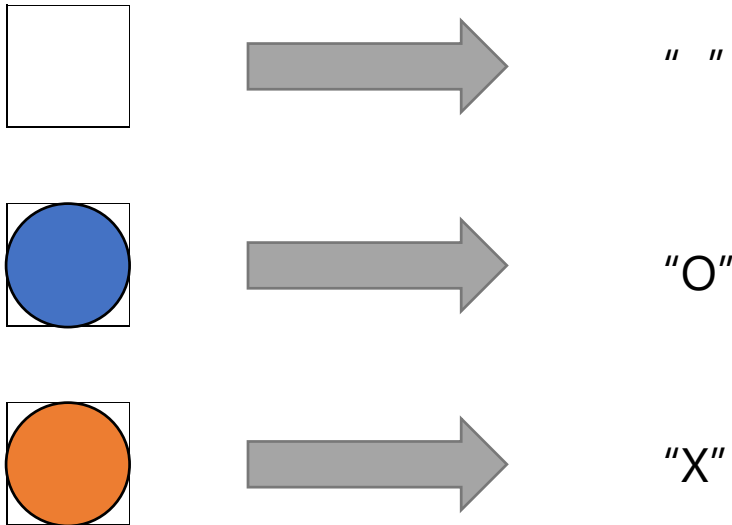
Remark on placing a piece

In this case, the player can place the blue piece here because there is a piece below.



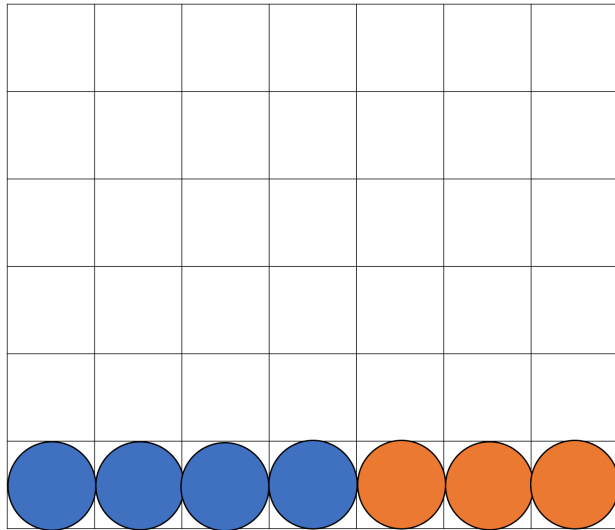
Example of implementation of the board 1/2

The board is often represented by a two-dimensional array in the source code. First, an empty space, a space with a blue piece, and a space with an orange piece are transformed as follows.



Example of implementation of the board 2/2

Using the transformation explained before, we can represent the entire board as follows and implement the program that process it.



```
Board = [
[" ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " "],
[" ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " "],
[" ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " "],
[" ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " "],
[" ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " "],
[" ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " ", " "],
["O", "O", "O", "O", "X", "X", "X"]]
```

```
for(i=0; i<6; i++){
  for(j=0; j<7; j++){
    //write the code for Board[i][j]
  }
}
```

Outline of the experiment

Outline of the experiment

- Implement the game step by step.
- Implementation is divided into four steps.
- Each step forms an exercise.
- You can use any programming language.

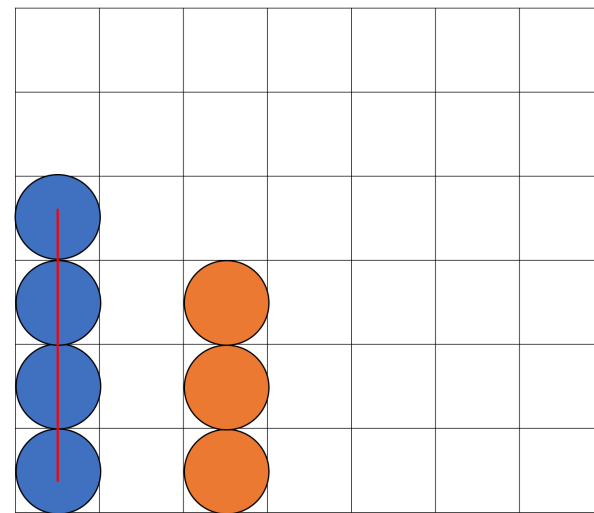
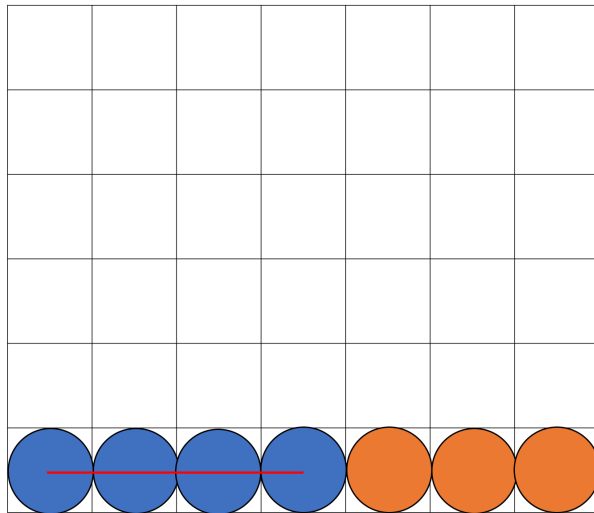
Steps of implementing the game

In this experiment, you implement the game in the following steps.

- 1: Judgement of winner (vertical/horizontal)
- 2: Judgement of winner (diagonal)
- 3: Functionality of placing a piece
- 4: Completion of implementation of the game

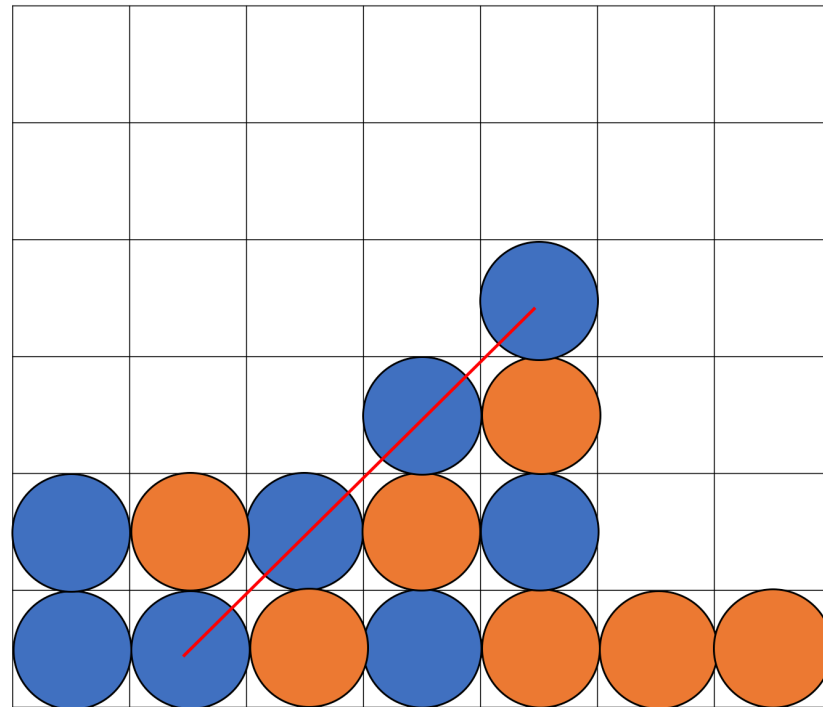
Exercise 1: Judgement of winner (vertical/horizontal)

As the first step, implement detection of a vertical or a horizontal line of four pieces of the same color to determine the end of the game



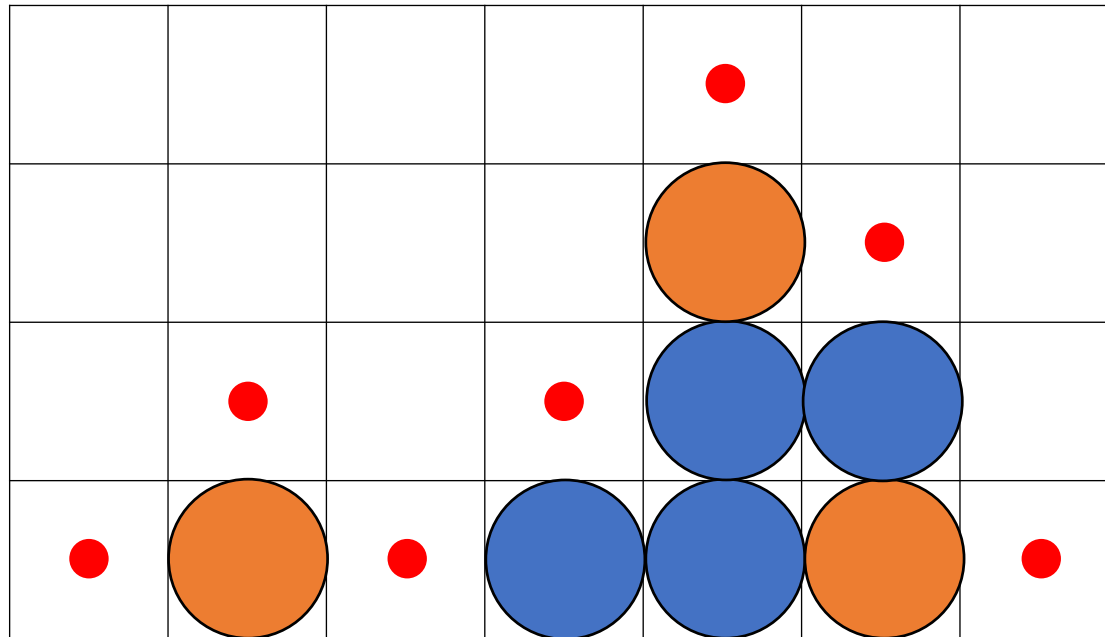
Exercise 2: Judgement of winner (diagonal)

Next, detect a diagonal line of four pieces of the same color.



Exercise 3: Functionality of placing a piece

In Exercise 3, implement the functionality for the players to place the pieces alternately. Note that they cannot place the pieces against gravity. (As for the example below, they can only place a piece in the space indicated by ●).



Exercise 3: Functionality of placing a piece

While you can implement in your own way, you can use the following example.

E.g. You can place a piece by inputting its coordinate using the keyboard.

```
X:3  
Y:4  
Placed the piece at (3,4)!
```

Exercise 4: Completion of implementation of the game

You will have implemented the functionality of judging the winner in Exercises 1 and 2, and the functionality of placing a piece in Exercise 3.

Exercise 4 requires to compose the functionalities and enable us to play the game.

While you can implement in your own way, you can refer to the next slide.

Exercise 4: Completion of implementation of the game (example)

For example, you can implement the game by displaying the current state of the board when the player places a piece by specifying its coordinate.

```
X:1
Y:2
Placed the piece at (1,2)!
□□□□□□
□□□□□□
□□□□□□
□●□□□□
●○●○●○●
○●○●○●○
```

Grading Policies

- You must submit your report by 17th, January, 2023.
- Your report must include source code, execution results of each exercise and the followings;
 - Originality and ingenuity of the algorithms you implemented if any
 - Comparison of the algorithms if you implemented several ones
- Your overall grade will be decided based on ease of understanding of your report as well as the report content.

Supplements

Introduction of online compilers

Online compilers enable you to compile and execute programs of various languages such as C and Java. They will help you if you have difficulties in setting the programming environments

paiza.io: <https://paiza.io/en>

The screenshot shows the paiza.io website interface. At the top is a blue navigation bar with the paiza.io logo, links for 'New code', 'Recent codes', and 'WebDev', a language dropdown set to 'English', and 'Sign Up' and 'Sign In' buttons. Below the navigation bar, the main content area has a light green background with a faint image of a tree-lined path. On the left, the text 'Just write and run code online !' is displayed above a question 'Which language will you use ?'. Below this is a grid of 24 green buttons, each containing a language name: Bash, C, C#, C++, Clojure, Cobol, CoffeeScript, D, Elixir, Erlang, F#, Go, Haskell, Java, JavaScript, Kotlin, MySQL, Objective-C, Perl, PHP, Python2, Python3, R, Ruby, Rust, Scala, Scheme, Swift, and VB. On the right, there is a large monitor displaying a code editor with Ruby code. The code includes comments and a function to parse a JSON response from an external API. Below the code editor is a video player showing a Vimeo video titled 'Any idea Learn new language external API batch jobs wrapping web sites'. The video player has a play button and a progress bar. To the right of the monitor is a blue cartoon bird logo.

Beta

paiza.io New code Recent codes WebDev English Sign Up Sign In ?

Just write and run code online !

Which language will you use ?

Bash C C# C++ Clojure Cobol
CoffeeScript D Elixir Erlang F# Go
Haskell Java JavaScript Kotlin MySQL
Objective-C Perl PHP Python2 Python3 R
Ruby Rust Scala Scheme Swift VB

Any idea
Learn new language
external API
batch jobs
wrapping web sites

Run (Ctrl-Enter)

01:29

vimeo

Introduction of online compilers

If you choose a language you want to use in the red rectangle, an editor appears and you can start coding.

paiza.io: <https://paiza.io/en>

The screenshot displays the paiza.io website. At the top is a blue navigation bar with the paiza.io logo, links for 'New code', 'Recent codes', and 'WebDev', a language dropdown set to 'English', and 'Sign Up' and 'Sign In' buttons. Below the navigation bar, the main content area features the text 'Just write and run code online !' and a prompt 'Which language will you use ?'. A grid of 28 green buttons representing various programming languages is shown, with a red rectangle highlighting the first five rows. To the right, a computer monitor displays the paiza.io code editor with Ruby code and a video player showing a tutorial. A blue bird logo is positioned in the bottom right corner.

Beta paiza.io New code Recent codes WebDev English Sign Up Sign In ?

Just write and run code online !

Which language will you use ?

Language selection grid (highlighted in red):

- Bash, C, C#, C++, Clojure, Cobol
- CoffeeScript, D, Elixir, Erlang, F#, Go
- Haskell, Java, JavaScript, Kotlin, MySQL
- Objective-C, Perl, PHP, Python2, Python3, R
- Ruby, Rust, Scala, Scheme, Swift, VB

Code editor (Ruby):

```
1 # http your code ?
2 require 'net/http'
3 require 'uri'
4 require 'pp'
5 require 'json'
6
7 url = URI.parse(URI.escape("https://app.paiza.co.jp/services/api/v1/submit/#{@id}/28148222?applicationId=104447789"))
8 res = Net::HTTP.start(uri.host, uri.port, :use_ssl => true){|https|
9   http.get(uri.path + "?url=#{url.query}")
10 }
11 obj = JSON.parse(res.body)
12 puts obj
13 puts "table border=1"
14 obj['items'].each{|item|
15   item = item['item']
16   if ! item['mediaImageUrl']
17     
```

Video player: Any idea Learn new language external API batch jobs wrapping web sites 01:29 vimeo

Blue bird logo

Introduction of online compilers

If you choose a language you want to use in the red rectangle and click the Start button, an editor appears and you can start coding.

paiza.io: <https://paiza.io/en>

Beta @paiza.io Online C compiler New code Recent code WebDev English Sign Up Sign In

Online C compiler

Online C compiler is online editor and compiler.
C, C++, Java, Ruby, Python, PHP, Perl,... More than 20 languages are supported.
You can use for learn programming, scrape web sites, write batch, etc...

Start Online C compiler (Free)

Or, choose [other languages...](#)

Any idea
Learn new language
external API
batch jobs
scraping web sites

```
1 # Write your code here
2 require 'net/http'
3 require 'uri'
4 require 'pp'
5 require 'json'
6
7 url = URI.parse(URI.escape("https://api.paiza.io/services/api/v1/bot/tasks/28148222/applicationId=104447789"))
8 res = Net::HTTP.start(uri.host, uri.port, use_ssl: true){|http|
9   http.get(uri.path + "?p=" + url.query)}
10
11 obj = JSON.parse(res.body)
12
13 puts "stable border=1"
14 obj['item'].each{|item|
15   item = item['item']
16   if ! item['mediaImageId']
17
18   }
19 }
```

Run (Ctrl-Enter)

01:29

vimeo

Introduction of online compilers

You type the code in the black part of the editor. If you click the Run button, the code is compiled and executed.

The screenshot shows the Paiza.io Online C compiler interface. At the top, there is a blue header bar with the Paiza.io logo, navigation links for 'New code', 'Recent code', and 'WebDev', a language selector set to 'English', and buttons for 'Sign Up' and 'Sign In'. Below the header, a green tab labeled 'C' is active, and a text input field prompts the user to 'Enter a title here'. The main editor area is dark-themed and contains a C program template with line numbers 1 through 6. The code is as follows:

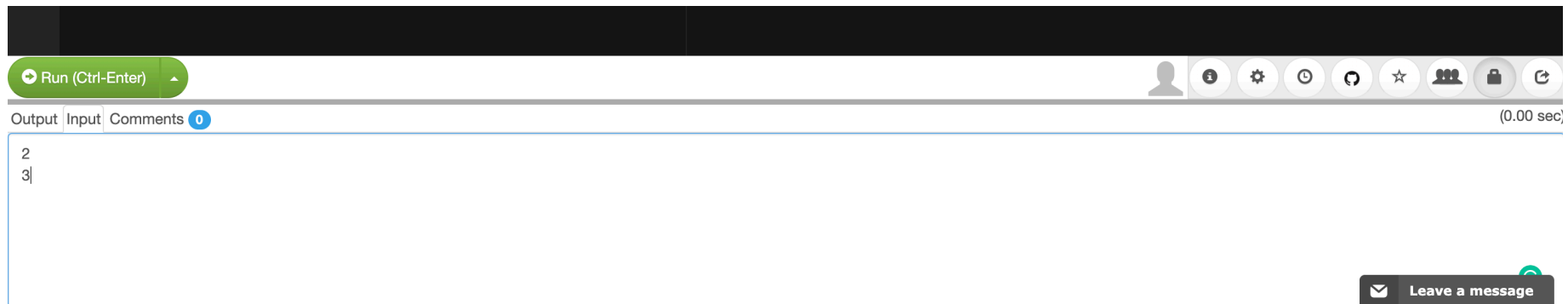
```
1 #include <stdio.h>
2 int main(void){
3     // Your code here!
4
5 }
6
```

At the bottom of the interface, there is a green 'Run (Ctrl-Enter)' button and a row of utility icons including help, settings, a clock, a refresh button, a star for bookmarks, a user profile icon, a lock icon, and a share icon.

Introduction of online compilers

When you use the standard input needed in Exercises 3, 4, etc., you need to input the data in the "input" tab at the bottom of the editor.

If it is inconvenient in playing the game, it will be better to prepare for programming language environments in your PC and use them.



Introduction of Connect4

For those who feel difficult in understanding Connect4, I introduce the following games you can play in practice.

- [4 in a Line](#)
- [Nintendo Clubhouse Games: 51 Worldwide Classics \(Four-in-a-row\)](#) (Nintendo Switch)