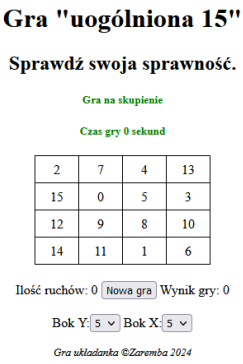
Lab 5
Game „Generalized 15”

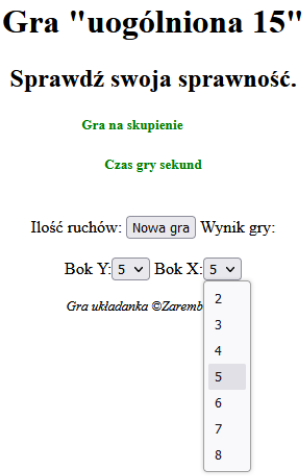
General description of the game.

1) The application allows you to play "generalized 15".
   a) The number "0" - means an empty field. The adjacent square with the number is moved to the square with zero.
   b) New game: number of moves = 0, game result = 0, random arrangement of numbers on the board.
   c) If the numbers are arranged "correctly" (or "almost correctly"):
      i) the screen should display information about the victory,
      ii) they should play a fanfare.



Step 1

We create an HTML file – the game interface.
Verbal descriptions, two selects with options set in the JS script.
All our further work is to expand the JS script.



html

```
<!DOCTYPE html>
<html lang=„en”>
<head>
   <meta charset=„UTF-8”>
```

```
    <link rel=„stylesheet” href=„.\css\style.css”>
    <script src=„.\js\skrypty.js”></script>
    <title>Gra „moja-15” PZ</title>
</head>
<body onload=„initGameXY()”>
    <header>
        <h1>Gra „uogólniona 15”</h1>
        <h2>Sprawdź swoja sprawność.</h2>
        <h5>Gra na skupienie krok 1</h5>
        <h5>Czas gry <span id=„time”></span> sekund</h5>
    </header>
    <table id=„plansza”></table><br>
    <label>Ilość ruchów: <span id=„current”></span></label>
    <button onclick=„startGameXY()”>Nowa gra</button>
    <label>Wynik gry: <span id=„wynik”></span></label><span id=„bingo”></span></label>
    <br><br>
    <label for=„bokY”>Bok Y:</label><select id=„bokY”></select>
    <label for=„bokX”>Bok X:</label><select id=„bokX”></select>
    <footer>
        <br>Gra układanka &#169;Zaremba 2024
    </footer>
</body>
</html>
```

Znaki specjalne: Copyright (znak praw autorskich):

- numer (decimal): ©
- numer (hex): ©
- encja: ©

CSS

```
body{
        text-align: center;
}
footer{
        font-style: italic;
        font-size: small;
}
table, td, th{
    border-collapse: collapse;
    margin-left: auto;
    margin-right: auto;
}
td{
    border: 1px solid;
    width: 50px;
    height: 30px;
}
```

JS

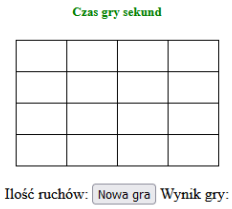
```
let bok0, bok1;     //ograniczenia dla selecta

function initGameXY(){bok0=2; bok1=8; zrobSelect();}
```

```
function startGameXY(){}
function zrobSelect(){
   let selectX = document.getElementById("bokX");
   let selectY = …;
   for (let i=bok0; i<=bok1; i++){
      let optionX = document.createElement('option');
      optionX.innerHTML = optionX.value = i;
      selectX.appendChild(optionX);
      …
   }
   let selected = Math.floor((bok1-bok0)/2);
   selectX.getElementsByTagName('option')[selected].selected = 'selected';
   selectY.get …;
}
```

Step 2

We add a table in which the game will be played.



JS – we add code

```
…
let sizeX, sizeY;              // board size

function initGameXY(){
   bok0=2; bok1=8;
   zrobSelect();
   sizeX=4, sizeY=4;          // it's temporary,
   initPlanszaXY();           // to make something happen, then we'll move
}
…
function initPlanszaXY(){
   let plansza = document.getElementById("…");
   for (let i=0; i<sizeY; i++){
      let row = plansza.insertRow(i);
      for (let j=0; j<sizeX; j++){
         let cell = row.insertCell(j);
      }
   }
}
```

Step 3

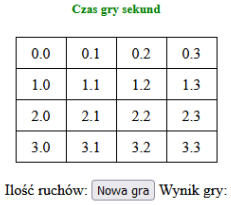
At the beginning of the script, declare two arrays of variables:

```
let liczbyWzorzec=[];  // a pattern showing what the arranged board looks like
let liczby=[];         // numbers after permutation using the Fisher-Yeats algorithm, printed on the
                          board
```

Visible changes – in the <table> table:

- we assign **id** to cells

- we write **id** in cells (only temporarily, we will change it soon)




```
function initPlanszaXY(){
…
   for (let i=0; i<sizeY; i++){
      let row = plansza.insertRow(i);
      for (let j=0; j<sizeX; j++){
         let cell = row.insertCell(j);
         cell.id = i+'.'+j;
         cell.innerHTML = cell.id
      }
   }
}
```

Changes (temporarily) not visible.
We prepare two arrays to store numbers and initialize them:

- array-pattern how numbers are to be arranged
- array with permutation of numbers from the pattern to arrange

```
function inicjujLiczbyXY(){
   for (let i=0; i<sizeX*sizeY; i++){
      liczbyWzorzec[i] = 1+(liczby[i]=i);
   }
}
```

We do the permutation using the **Fisher–Yeats** algorithm:

```
let c;
for (let j, i=sizeX*sizeY-1; i>0; i--){
   j = Math.floor(Math.random()*(i+1));
   c = liczby[j];
   liczby[j] = liczby[i];
   liczby[i] = c;
}
```

Step 4

We modify the initGame() function by transferring the creation of the game board to startGame() according to the choices bokY, bokX:

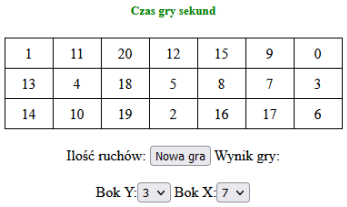
```
function initGameXY(){
   bok0=2; bok1=8;
   zrobSelect();
   startGameXY();
}
```

- We read the choices bokX, bokY
- We remove the old board
- We initialize new numbers to display, permute
- We create a new board

```
function startGameXY(){
   sizeX=document.getElementById("bokX").value;
   sizeY=document.getElementById("bokY").value;
   let plansza = document.getElementById("plansza");
   if (plansza.children.length > 0) {
      plansza.removeChild(plansza.lastChild);
   }
   inicjujLiczbyXY();
   FisherYeatsXY();
   initPlanszaXY();
}
```

In the initPlansza () function, instead of id we enter the numbers after permutation:

```
function initPlanszaXY(){
   …
         cell.innerHTML = liczby[i*sizeX+j];
         // cell.innerHTML = cell.id
      }
   }
}
```



Step 5

We add an event handler – clicking on a cell in the table.

Clicking moves the “table – number”, if it can be moved, simultaneously, if necessary, modifying the game result.

We extend the function

```
function initPlanszaXY(){
      …
   cell.innerHTML = liczby[i*sizeX+j];
   cell.onclick = function(){
      if (cell.innerHTML>0){
         if ((j>0)&&(document.getElementById(i+'.'+(j-1)).innerHTML==0)){
            wynik -= (cell.innerHTML==(i*sizeX+j+1));
            wynik += (cell.innerHTML==(i*sizeX+j));
            document.getElementById(i+'.'+(j-1)).innerHTML = cell.innerHTML;
            cell.innerHTML = 0;
         } else if ((i>0)&&(document.getElementById((i-1)+'.'+j).innerHTML==0)){
               …
         } else if((j<sizeX-1)&&document.getElementById(i+'.'+(j+1)).innerHTML==0) {
               …
         } else if ((i<sizeY-1)&&(document.getElementById((i+1)+'.'+j).innerHTML==0)){
               …
         }
      }
   }
}
```

```
    }
}
```

Step 6

We count the number of correctly set tiles.
We declare two additional variables:

```
let currentMove;      // next move number - number of moves made
let wynik;            // current number of correctly set plates
```

We add two functions:

- Counting "random" correct settings after permutation:

```
function wynikGryPoPermutacjiXY(){
  for (let i=0; i<sizeX*sizeY; i++){
    wynik += (liczby[i]==liczbyWzorzec[i]);
  }
}
```

- Printing information about the current game result and the next move number:

```
function writeInfoXY(){
  document.getElementById("wynik").innerHTML = wynik;
  document.getElementById("current").innerHTML = currentMove;
}
```

We call these functions

- When the game starts:

```
function startGameXY(){
  …
  initPlanszaXY();
  wynikGryPoPermutacjiXY();
  writeInfoXY()
}
```

We add this **writeInfoXY** function and call it to handle clicking on the "tile".
After clicking, we print the new, current game result - the number of clicks and the number of well-placed tiles.

```
function initPlanszaXY(){
    …
  cell.innerHTML = liczby[i*sizeX+j];
  cell.onclick = function(){
    if (cell.innerHTML>0){
      if ((j>0)&&(document.getElementById(i+'.'+(j-1)).innerHTML==0)){
        wynik -= (cell.innerHTML==(i*sizeX+j+1));
        wynik += (cell.innerHTML==(i*sizeX+j));
        document.getElementById(i+'.'+(j-1)).innerHTML = cell.innerHTML;
        cell.innerHTML = 0;
        currentMove++;
        writeInfoXY();
      } else if ((i>0)&&(document.getElementById((i-1)+'.'+j).innerHTML==0)){
          …
```

```
                } else if((j<sizeX-1)&&document.getElementById(i+'.'+(j+1)).innerHTML==0) {
                    …
                } else if ((i<sizeY-1)&&(document.getElementById((i+1)+'.'+j).innerHTML==0)){
                    …
                }
            }
        }
    }
}
```

Step 7

We add information about winning:

- **Bingo!**
- bingo!

We know you can't always win!

```
function writeInfoXY(){
…
   document.getElementById("current").innerHTML = currentMove;
   if (wynik == sizeX*sizeY-1){
      document.getElementById("..").innerHTML = " Bingo!";
   } else if ((wynik == sizeX*sizeY-3)&&rewersLast2XY()){
      document.getElementById("..").innerHTML = " bingo!";
   }
}

function rewersLast2XY(){
   let rewers = 1;
   if (sizeX<3){ rewers=0; return;}                    // tu musimy zmienić kod!!!
   rewers &= document.getElementById((sizeY-1)+'.'+(sizeX-3)).innerHTML == (sizeX*sizeY-1);
   rewers &= document.getElementById((sizeY-1)+'.'+(sizeX-2)).innerHTML == (sizeX*sizeY-2);
   return rewers;
}
```

Step 8

We add information about the game time.

We add a declaration of four variables:

```
let dateStart, dateEnd;     // new Date();
let timeStart, timeEnd;     // seconds
```

At the start of the game we "start the clock":

```
function startGameXY(){
…
   wynikGryPoPermutacjiXY();
   dateStart = new Date();
   timeStart = Math.round(dateStart.getTime()/1000);
   writeInfoXY()
}
```

We add time information to the printed information:

```
function writeInfoXY(){
```

```
…
   document.getElementById("bingo").innerHTML = "";
   dateEnd = new Date();
   timeEnd = Math.round(dateEnd.getTime()/1000);
   document.getElementById("time").innerHTML = timeEnd - timeStart;
   if (wynik == sizeX*sizeY-1){
      …
   }
}
```

Step 9

We add victory fanfares (sound).

In **HTML** element<audio> and two buttons

```
…
   <label for="bokX">Bok X:</label><select id="bokX"></select>
   <br><audio controls id="music" hidden>
      <source id="myMusic" src="./Sound/beetSym5.mp3" type="audio/mpeg">
   </audio>
   <br><button onclick="startPlay()" id="btnStart" hidden>Start sound</button>
         <button onclick="stopPlay()" id="btnStop" hidden>Stop sound</button>
  <footer?
…
```

In the JS script

Sound support – three functions:

```
function startPlay(){
   mySound = document.getElementById('music');
   mySound.play();
}
function stopPlay(){
   mySound = document.getElementById('music');
   mySound.pause();
}
function playSound(p){
   if (p){
      document.getElementById("btnStart").hidden = false;
      document.getElementById("btnStop").hidden = false;
      startPlay();
   }else{
      document.getElementById("btnStart").hidden = true;
      document.getElementById("btnStop").hidden = true;
      stopPlay();
   }
}
```

We still need to add a call to the **playSound** function in the **writeInfo** function to turn the fanfares on and off at the right time.

And that's it.