

MATRIX CALCULATOR

# 행렬계산기

# 목차

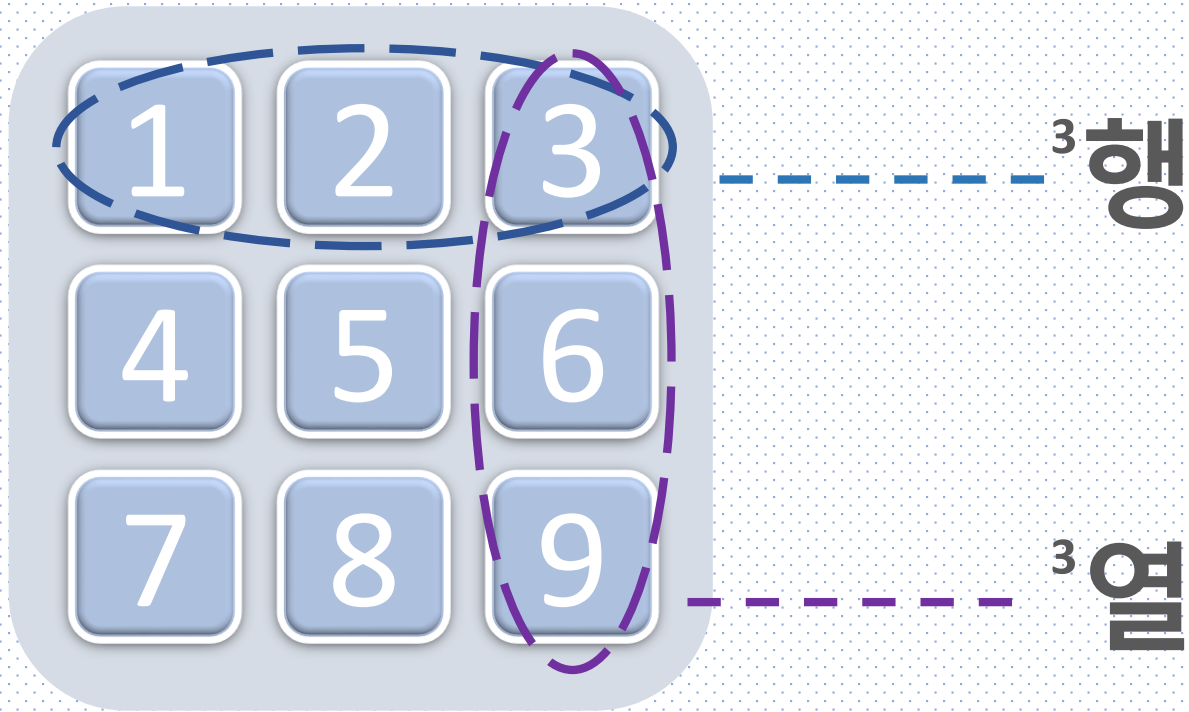
1. 행렬이란? p.3~7
2. 행렬계산기 소개 p.8~18
3. HTML / JAVASCRIPT p.19~25
4. 후기 p.26~29

## MATRIX CALCULATOR

# 1. 행렬이란?

p.3~7

# 1. 행렬이란?



# 1. 행렬이란?



행

3행 3열

열

# 1. 행렬이란?



3 X 3



3 X 3

행과 열이 똑같은 서로 다른 행렬은 더하기 빼기 가능

# 1. 행렬이란?

1	2	3	10
4	5	6	11
7	8	9	12

4 X 3

3	2	1
6	4	9
8	7	5

3 X 3

첫번째 행렬의 열과 두번째 행렬의 행이 같으면 곱셈 가능

MATRIX CALCULATOR

## 2. 행렬계산기 소개

---

p.8~18



## 2. 행렬계산기 소개

[행렬계산기 바로가기](#)



### MATRIX CALCULATOR



#### 디자인의 계기

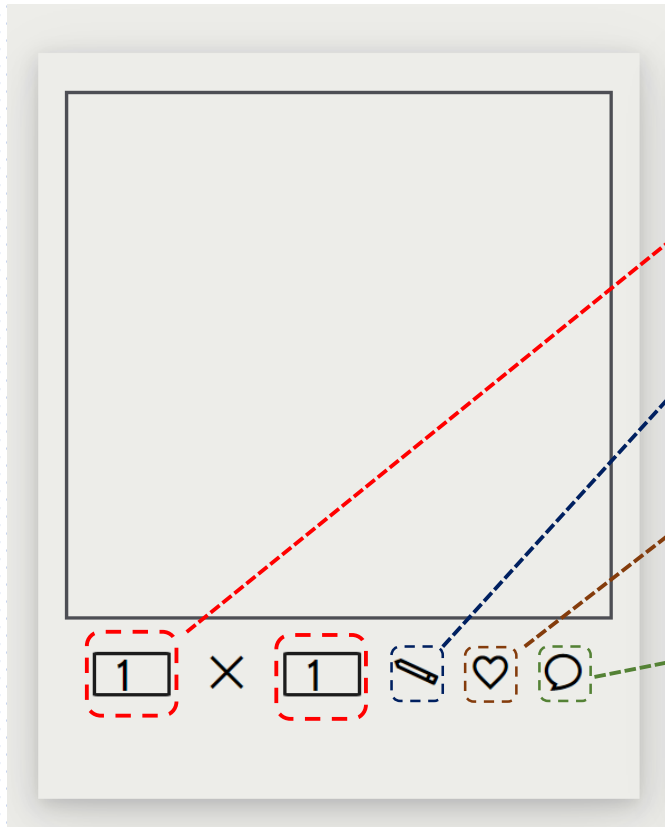
실생활에서 자주 사용하는 앱(인스타그램)을 모티브로 하여 친숙하면서도 세련되어 보이는 디자인을 하였습니다.

#### 인터페이스 디자인

인스타그램은 (좋아요 ) , (댓글창 ) 왼쪽에 있지만, 행렬계산기 특성상 먼저 숫자를 입력 받고 버튼을 클릭 하는 게 사용자의 편의성에 더 적합하다고 느껴 오른쪽에 디자인 하였습니다.

## 2. 행렬계산기 소개

[행렬계산기 바로가기](#)



1. 행렬 입력 칸 1~9 까지 입력 가능

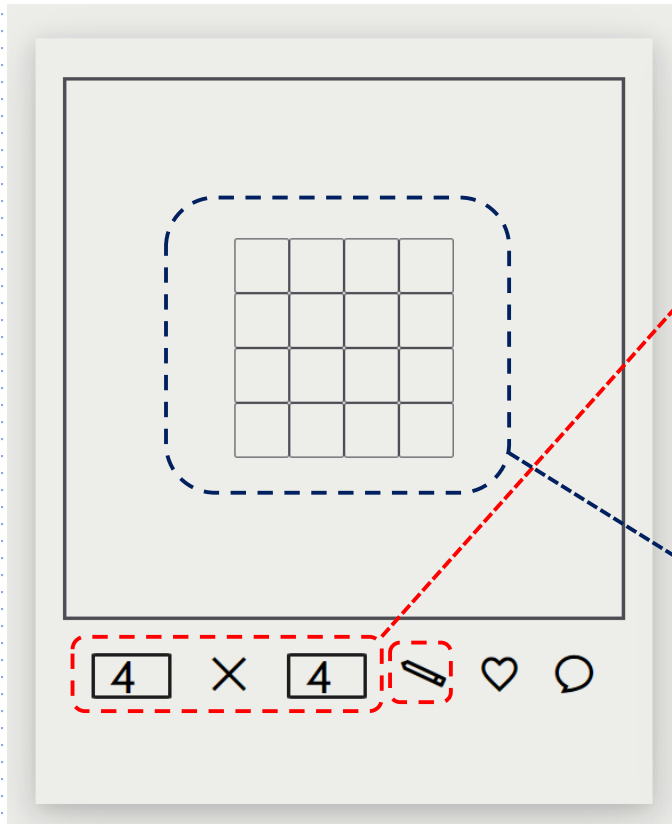
2. 행렬 만들기 버튼

3. 랜덤 값 버튼 -50 에서 50 사이의  
랜덤 숫자 입력

4. 초기화 버튼 행렬 안의 숫자가  
0 으로 초기화

## 2.행렬계산기 소개

[행렬계산기 바로가기](#)

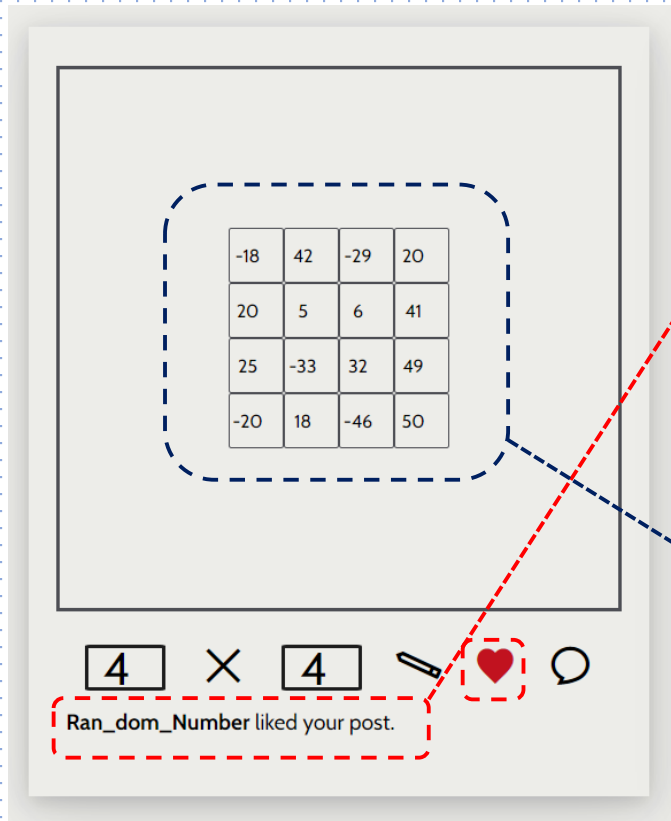


만들고 싶은 행과 열을  
입력 한 후, 연필버튼을 누르면  
행렬이 만들어 집니다.

행렬안에 넣고 싶은 숫자를  
입력할 수 있습니다.  
범위는 -99 부터 99 까지 입니다.

## 2.행렬계산기 소개

[행렬계산기 바로가기](#)

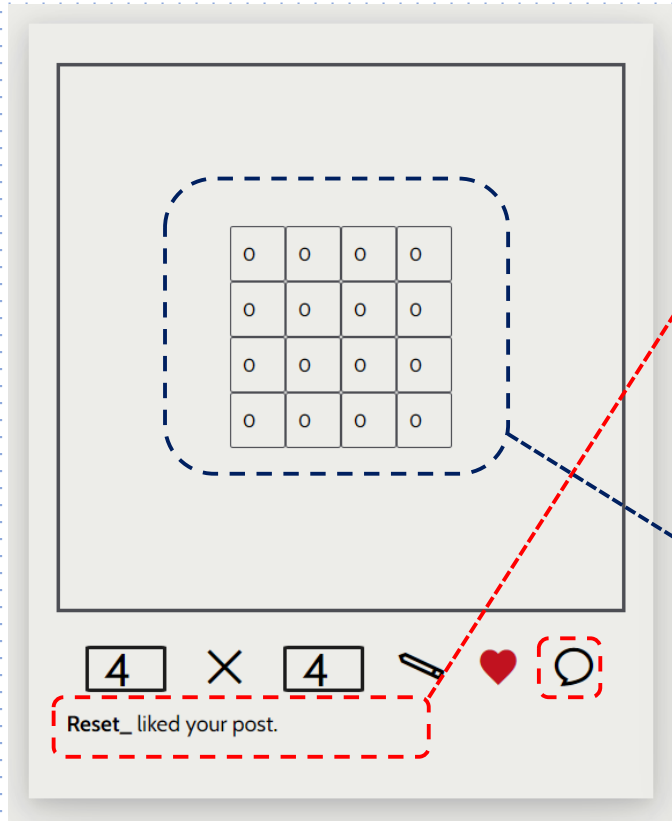



좋아요♡버튼을 누르면  
랜덤버튼 이라는 것을  
간접적으로 보여줌으로써 사용자의  
혼란을 막았습니다.

-50부터 50까지의  
랜덤 숫자를 입력합니다

## 2.행렬계산기 소개

[행렬계산기 바로가기](#)

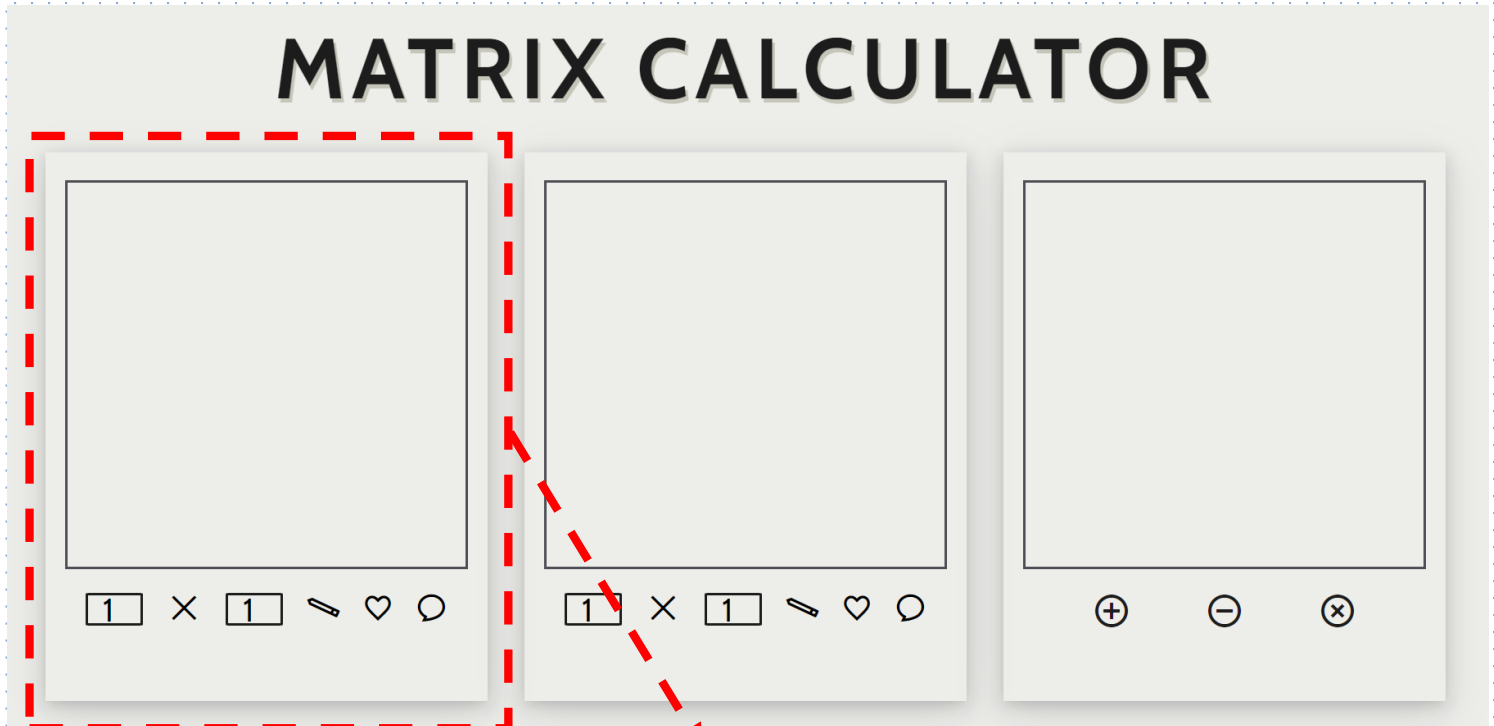


댓글칭  버튼을 누르면  
리셋 버튼 이라는 것을  
간접적으로 보여줌으로써 사용자의  
혼란을 막았습니다.

입력값이 0 으로 리셋됩니다.

## 2. 행렬계산기 소개

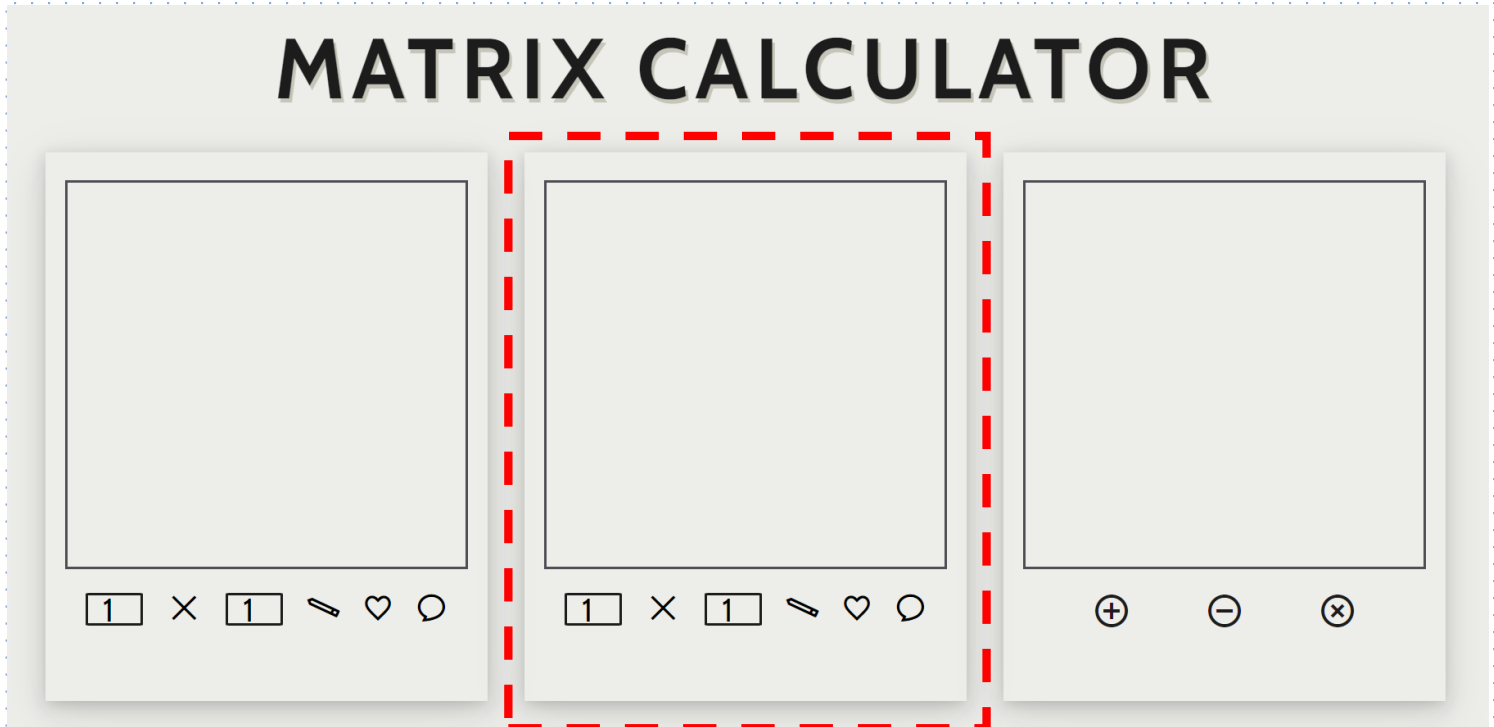
[행렬계산기 바로가기](#)



첫번째 행렬 만드는 곳

## 2. 행렬계산기 소개

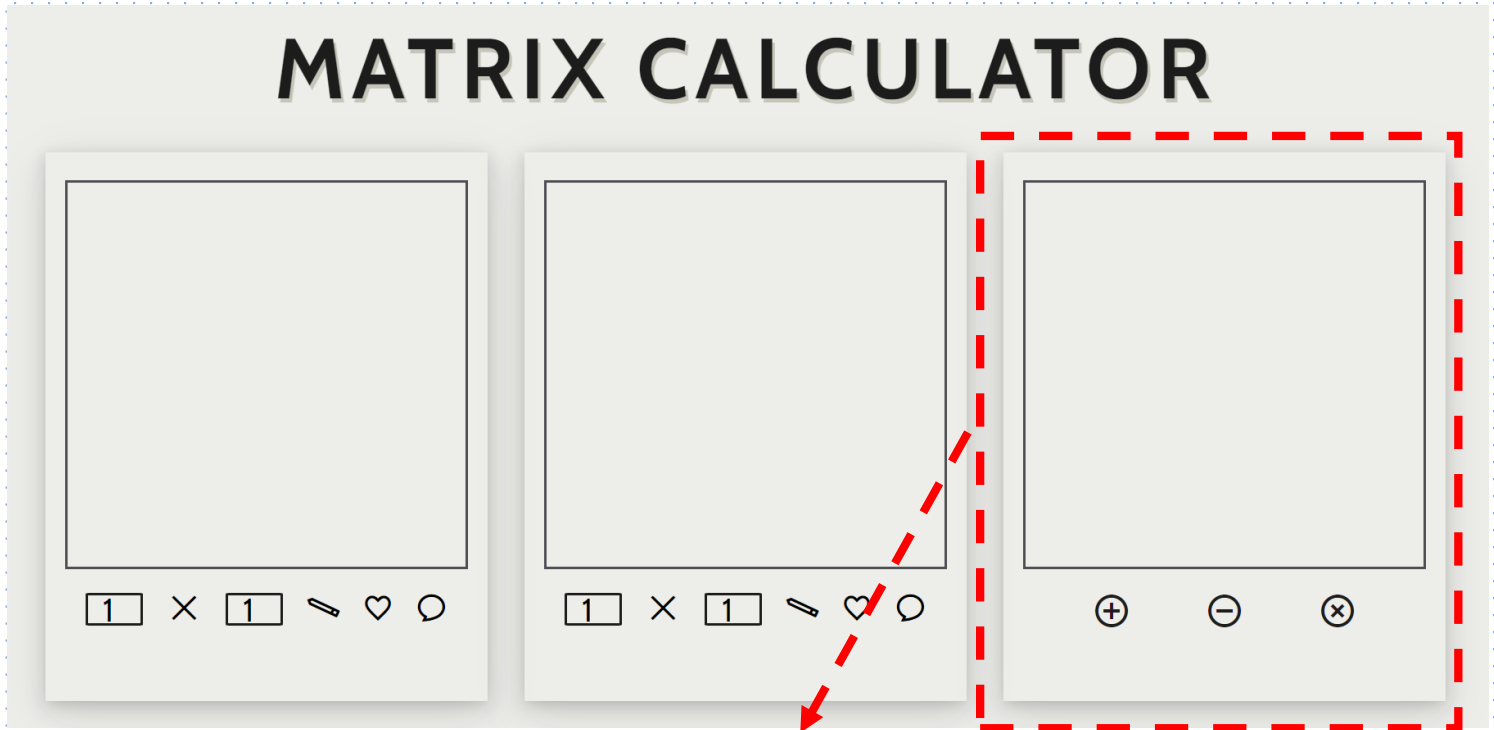
[행렬계산기 바로가기](#)



두번째 행렬 만드는 곳

## 2. 행렬계산기 소개

[행렬계산기 바로가기](#)



결과 나오는 곳



## 2. 행렬계산기 소개

[행렬계산기 바로가기](#)

17	-37	19	-38	-44
-12	32	27	-16	-41
8	-49	24	-22	11
-29	-49	47	10	-14
19	48	22	-29	41

5

×

5



Ran\_dom\_Number liked your post.

34	28	-7	-38	33	1	-23	-20	-39
-23	-32	20	-1	-37	-10	-19	-43	19
39	-27	-6	46	38	-23	-35	24	-41
33	-31	49	7	-31	-46	-39	47	-39
31	-38	-10	-47	9	-43	-2	8	-48

5

×

9



Ran\_dom\_Number liked your post.



Row and column are different...



PLUS\_ liked your post.

첫번째 행렬의 열과 두번째 행렬의 열이 다릅니다!

더하기 빼기 불가능

## 2.행렬계산기 소개

[행렬계산기 바로가기](#)

17	-37	19	-38	-44
-12	32	27	-16	-41
8	-49	24	-22	11
-29	-49	47	10	-14
19	48	22	-29	41

5

×

5

Ran\_dom\_Number liked your post.

34	28	-7	-38	33	1	-23	-20	-39
-23	-32	20	-1	-37	-10	-19	-43	19
39	-27	-6	46	38	-23	-35	24	-41
33	-31	49	7	-31	-46	-39	47	-39
31	-38	-10	-47	9	-43	-2	8	-48

5

×

9

Ran\_dom\_Number liked your post.

-448	3,997	-2,395	2,067	3,434	3,590	1,217	-431	1,449
-1,890	-35	188	3,481	-427	1,546	-571	-1,568	2,561
1,950	1,408	-2,368	178	3,770	485	743	1,577	-1,897
1,870	-291	-429	4,041	2,206	-478	-409	4,173	-1,445
714	-2,257	-1,136	-1,888	955	-1,396	-1,070	-2,951	-1,568

+

-

×

TIMES\_ liked your post.

첫번째 행렬의 열과 두번째 행렬의 행이 같습니다!

곱하기 가능!

MATRIX CALCULATOR

## 3. HTML / JAVASCRIPT

p.19~25

# 3. HTML / JAVASCRIPT

## 행렬계산기 바로가기

```
<article>
  <input type="number" id="row1" class="row1 getValue" value="1" max="10" min="1" pattern="[0-9]"
    maxlength="2" autofocus oninput='matrix.handleOnInput(this, 1)'/>
  <i class="xi-close"></i>
  <input type="number" id="column1" class="column1 getValue" value="1" max="10" min="1" pattern="[0-9]"
    maxlength="2" oninput='matrix.handleOnInput(this, 1)'/>
  <i class="xi-pen-o" id="makeMatrixBtn1"></i>
  <i class="xi-heart-o" id="xi-heart-o1"></i>
  <i class="xi-heart" id="xi-heart1"></i>
  <i class="xi-speech-o" id="xi-speech-o1"></i>
```

편의성을 위해  
포커싱을  
맞추었습니다.

1~9까지의 숫자를 입력  
받기 위해 함수를  
실행하였습니다.

정규식을 사용하여  
숫자만 입력  
받았습니다.

# 3. HTML / JAVASCRIPT

행렬계산기 바로가기

```
},  
handleOnInput: function (el, maxlength) {  
    if (el.value === "0") {  
        el.value = "";  
    }  
    if (el.value[0] === "-") {  
        el.value = el.value.substr(0, maxlength + 1);  
    }  
    else if (el.value.length > maxlength) {  
        el.value = el.value.substr(0, maxlength);  
    }  
},  
return true;  
});
```

행렬 안 값에 음수를  
입력하면 3자리 까지  
받을 수 있도록 하였습니다.

행렬 입력에는 1자리를,  
행렬 안의 입력에는 2자리를  
받을 수 있게 하였습니다.

# 3. HTML / JAVASCRIPT

## 행렬계산기 바로가기

```
matrixArea1:
function () {
  matrix.makeMatrixBtn1.addEventListener(
    'click', function () {
      matrix.inputValue();
      document.querySelector('#real1').innerHTML = "";
      for (let i = 0; i < matrix.row1; i++) {
        for (let j = 0; j < matrix.column1; j++) {
          document.querySelector('#real1').innerHTML += <input type="number" max="99" pattern="[0-9]" min="-99" maxlength="2" oninput='matrix.handleOnInput(this, 2)' id="m1${i}${j}">;
          document.getElementById('real1').style.width = (48 * matrix.column1) + 'px';
        }
      }
    }
  )
}
```

행과 열을 입력 값에 맞게  
설정하기 위해서  
넓이 값을 지정하였습니다.

행렬을 만듦과 동시에 아이디를  
만들어 값을 가져 올 수 있도록  
하였습니다.

# 3. HTML / JAVASCRIPT

[행렬계산기 바로가기](#)

```
matrixArray1:
function () {
    matrix.tempA = [];
    let tempM1 = [];
    for (let i = 0; i < matrix.row1; i++) {
        for (let j = 0; j < matrix.column1; j++) {
            let m1 = document.querySelector(`#m1${i}${j}`).value;
            tempM1[j] = m1;
        }
        matrix.tempA[i] = tempM1;
        tempM1 = [];
    } console.log(matrix.tempA);
}
```

행렬 안에 있는 입력 값을  
아이디를 통해 가져와 빈 배열에  
넣어주었습니다.

# 3. HTML / JAVASCRIPT

## 행렬계산기 바로가기

```
random1: function () {  
  this.randomBtn1.addEventListener(  
    'click',  
    function () {  
      document.querySelector('#xi-heart-o1').style.display = "none";  
      document.querySelector('#xi-heart1').style.display = "inline-block";  
      document.querySelector('#instagramBox1').style.display = "block";  
      document.querySelector('#instagramBox6').style.display = "none";  
      for (let i = 0; i < matrix.row1; i++) {  
        for (let j = 0; j < matrix.column1; j++) {  
          document.querySelector(`#m1${i}${j}`).value = (Math.floor(Math.random() * 101) - 50);  
        }  
      }  
      matrix.matrixArray1();  
    }  
  }  
}
```

랜덤 값으로 인해  
달라진 행렬의 값을  
배열 안에 다시 넣어주는  
함수를 실행하였습니다.

랜덤 값을 - 50 ~ 50 으로  
지정해주었습니다.



# 3. HTML / JAVASCRIPT

## 행렬계산기 바로가기

```
if (matrix.row1 === matrix.row2 && matrix.column1 === matrix.column2) {  
    SUM();  
} if (matrix.row1 !== matrix.row2 || matrix.column1 !== matrix.column2) {  
    document.querySelector('#real3').innerHTML = `Row and column are different ...`;  
}
```

첫번째 행렬의 행과  
두번째 행렬의 행이 같고,  
첫번째 행렬의 열과  
두번째 행렬의 열이 같을 때  
더하기 함수가 실행되도록  
하였습니다.

만약 다르다면 함수가  
실행되는 것이 아닌  
오류 창이  
나오게 하였습니다.

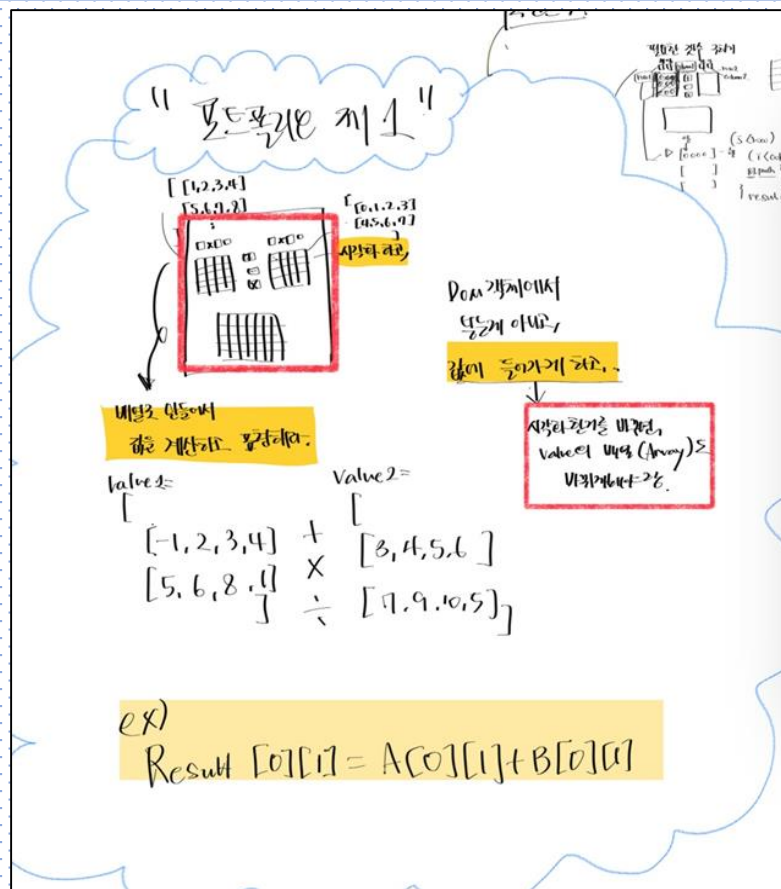
## MATRIX CALCULATOR

# 4. 후기

---

p.26~29

## 4. 후기 (마음가짐)

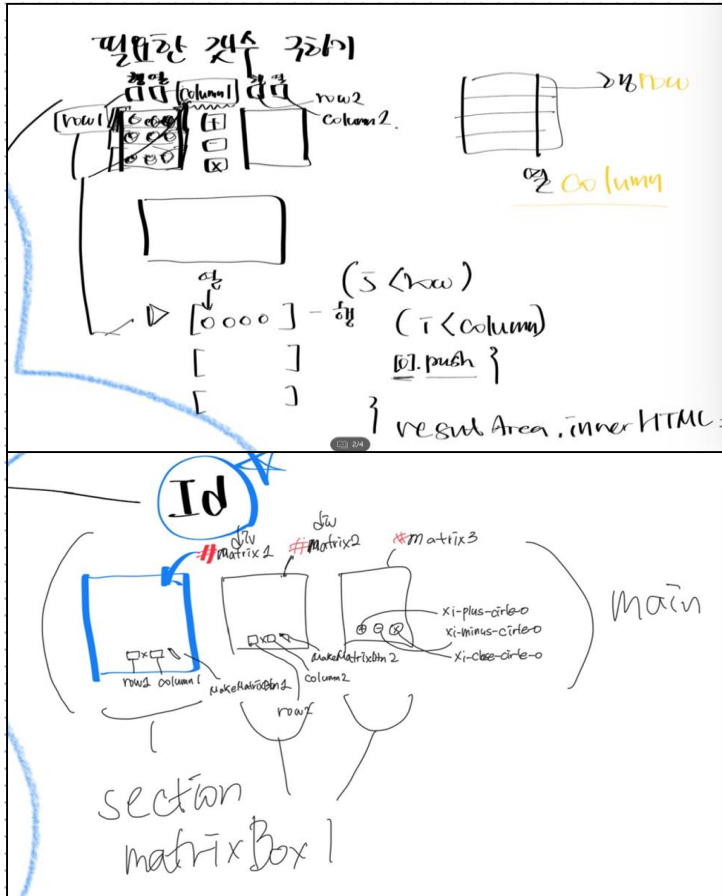


처음 행렬계산기를 만들 때,  
값을 배열 안에 넣고 다시 그 값을  
꺼내서 화면에 출력해야 한다는 것을  
이해하기 힘들었습니다.  
그림을 그리면서 이해하려고  
노력하였습니다.

그림을 그리면서 스스로에게  
설명하면서 제 자신을  
이해시키려고 노력했습니다.

결국 이해가 되었고 그 후에 바로 디자인 작업으로 들어갔습니다.

# 4. 후기 (디자인)



처음 디자인은  
양쪽에 행렬이 있고  
아래에 계산 결과가 나오게 하려고  
디자인 하였는데 그렇게 하면  
디자인이 깔끔하지 않을 것 같았습니다.

똑같은 크기의 상자 3개를 나란히  
놓는게 깔끔할 것 같아서 구상을  
시작했습니다.

인스타그램이라는 컨셉을 잡기까지  
고민을 많이 했지만 친숙하면서  
세련된 디자인을 표현하면서 버튼 표현도  
가능하기에 선정하게 되었습니다.

# 4. 후기 (JAVASCRIPT)

다 (1) (2) (3) (4)

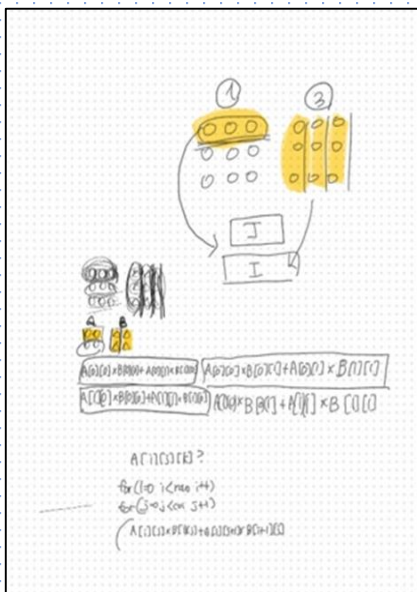
$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} e & f \\ g & h \end{pmatrix}$$

ae

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} e & f \\ g & h \end{pmatrix}$$

$$= \begin{pmatrix} a \times e + b \times g & a \times f + b \times h \\ c \times e + d \times g & c \times f + d \times h \end{pmatrix}$$

temp A [a b] temp B [e f]  
[c d] [g h]



행렬의 곱셈을 처리하는 것이  
너무 어려웠습니다.

For 문을 두 개 까지는  
머릿속에서 어떻게든 돌릴 수 있지만  
세 개 이상은 저의 뇌가 받아들이기  
힘들었습니다.

글씨로 계속 써보고  
스스로 규칙을 찾으려고 생각하면서  
고적이다보니, 깨닫는 순간이 왔습니다.

그 순간 제 자신이 대견하면서  
만드는게 쉽지는 않지만  
결국은 해낼 수 있을 것이라는  
희망을 얻었습니다.

MATRIX CALCULATOR

**감사합니다**