23-11-06

- 자바스크립트 언어
- 배열과 반복문

<문제>

1.

```
//1 부터 100 까지의 정수중 3 의 배수의 합을 구하라!
let sum=0;

for (i=1;i<=100;i++){
    if(i%3==0) sum+=i; //sum=sum+i 로 써도 된다.

} console.log(sum);
```

결과 값 : 1683

2.

a= 1,2,3,4,5 for(a=1;a<=5;a++)
b= 1,2,3 for(b=1;b<4;b++)

a * b 의 모든 값을 구하라

```
for(let a=1;a<=5;a++){
    for (let b=1;b<=3;b++){
        c = a*b;
        console.log(c);
    }
}
```

```
PS C:\work\javascript\11-06> node exm05.js
1
2
3
2
4
6
3
6
9
4
8
12
5
10
15
```

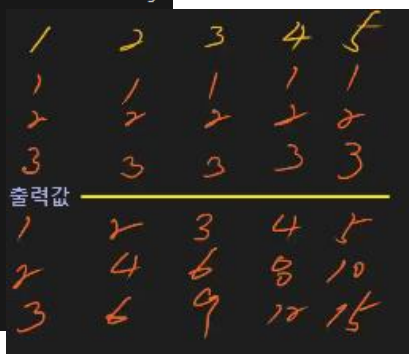
결과값 :

정답

```
for(let a=1;a<=5;a++){

    for (let b=1;b<=3;b++){
        console.log("a*b=", +a*b);
    }
}
```

```
PS C:\work\javascript\11-06> node exm05.js
a*b= 1
a*b= 2
a*b= 3
a*b= 2
a*b= 4
a*b= 6
a*b= 3
a*b= 6
a*b= 9
a*b= 4
a*b= 8
a*b= 12
a*b= 5
a*b= 10
a*b= 15
```



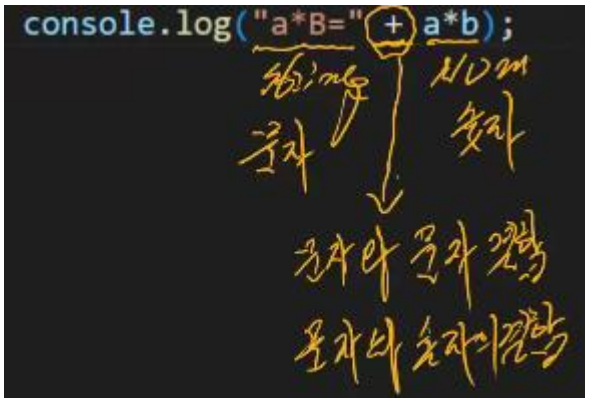
1,2,3,4,5 에서 1,2,3 반복해서 돌아간다.

구구단에 사용한다. / "a*b="이것은 string 문자이다. +a*b은 num 숫자.

반복문 사용하는 이유 : 반복문하지 않을 때 15번 입력해야 하는 것을 한번에 입력해 준다.

이것이 반복문 사용하는 이유

+의 의미 : 문자와 문자 결합, 문자와 숫자의 결합



3. 구구단 출력해보기(2 단~9 단까지)

```
4. //3 번. 2 단~9 단까지 구구단 출력해보기
5. for(let a=2;a<=9;a++){
6.     console.log("구구단 " +a ,"단");
7.     for (let b=1;b<=9;b++){
8.         console.log("a*b=", +a*b);
9.     }
```

```
a*b= 21
a*b= 24
a*b= 27
구구단 4 단
a*b= 4
a*b= 8
a*b= 12
a*b= 16
a*b= 20
a*b= 24
a*b= 28
a*b= 32
a*b= 36
구구단 5 단
a*b= 5
a*b= 10
a*b= 15
```

+ 추가로 이 값들 다 더한 값들 구하기

```
sum =0;
for(let a=2;a<=9;a++){
    console.log("구구단 " +a ,"단");
    for (let b=1;b<=9;b++){
        c=a*b;
        console.log(a,"*",b,"=", +c);
        sum +=c
    }
}
console.log("구구단 전체 합 = " +sum);
```

```
구구단 7 단
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
구구단 8 단
8 * 1 = 8
8 * 2 = 16
8 * 3 = 24
```

```
구구단 9 단
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
구구단 전체 합 = 1980
```

최종적인 합계는 1 번 for 문과 2 번 for 문 밖에 써야 최종 값이 나올 것.

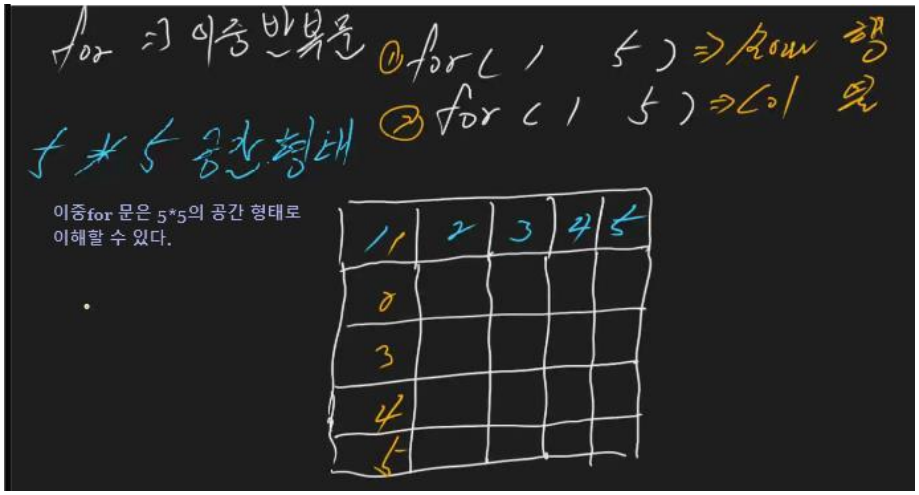
추가 문제는 1~5 의 숫자의 1,2,3 곱한 수의 합계 구하기

```
let sum=0;
 for(let a=1; a<=5;a++){
    for(let b=1;b<4;b++){
        console.log("a*B=" + a*b);
        sum = sum + a*b;
    }
}
console.log(sum);
```

1부터 5의 숫자의 1,2,3 곱한 수의 합계 구하기

```
PS C:\work\javascript> node exm04.js
a*B=1
a*B=2
a*B=3
a*B=2
a*B=4
a*B=6
a*B=3
a*B=6
a*B=9
a*B=4
a*B=8
a*B=12
a*B=5
a*B=10
a*B=15
90
```

배열 (Array) : 반복문하고 세트로 오는 아이, 저장공간을 담기 위해 만들어진 것.
배열과 반복문은 따로 가는 것이 아니라 하나의 덩어리이다.



선 -> 면 -> 공간(3 차원)

배열은 [ ] 로 표현한다.

```
let a= [1,2,3,4];
console.log(a);
```

// a 는 1,2,3,4, 값을 가지고 있는 배열이다.

결과값 :
```
PS C:\work\javascript\11-06> node exm05.js
[ 1, 2, 3, 4 ]
```

1,2,3,4 자료값을 가지고 있는 배열이다.

```
let a= [1,2,3,4]; //자료값의 위치가 각각 다르다
for (let i=0; i<a.length; i++){ //a.length 는 a 의 총길이(자료의 개수)
    console.log(a[i]);
}
```

결과값 :
```
PS C:\work\javascript\11-06> node exm05.js
1
2
3
4
```

a.length 인 총 길이, 자료의 개수만큼 a 의 값을 출력한 것, address 의 길이 값.
a[0] =1 로, 0 의 자리에 있는 값이 1 이다.

```
let a= [one,two,three,four];
for (let i=0; i<a.length; i++){
    // console.log(a.length);
    console.log(a[i]);
}
```

값 출력이 안된다 이유는 문자열 이기 때문에.

아래와 같이 문자열로 바꾸면 출력이 된다. ↓

```
let a= ["one","two","three","four"];
for (let i=0; i<a.length; i++){
    console.log(a.length +"/" +i);
    console.log(a[i]);
}
```

결과값 :

```
PS C:\work\javascript\11-06> node exm05.js
4/0
one
4/1
two
4/2
three
4/3
four
```

4/0
one
4/1
two
4/2
three
4/3
four

i=0 으로 하는 이유는 배열을 셀 때 0 부터 세기때문이다. i=1 로 바꾸면 two 인 두번째부터 시작함

JSON 데이터 : 각각의 데이터를 "큰 따옴표로 묶어 주는 것. 수치자료는 "큰 따옴표 사용안했다.

최종적으로 데이터 주고받을 형태

"one","two",3,"four"

Typeof : 자료형태 무엇인지 알려주는 명령어

```
let a= ["one","two",3,"four"];
for (let i=0; i<a.length; i++){
    console.log(a.length +"/" +i);
    console.log(a[i]);
    console.log(typeof a[i]);
    console.log(typeof a);
}
```

```
PS C:\work\javascript\11-06> node exm05.js
4/0
one
string
object
4/1
two
string
object
4/2
3
number
object
4/3
four
string
object
```

Array = 객체(object)이다.

결합체이다.

[ ]의 주소는 0 부터 시작 address

[0,1,2,3,] => 이것의 개수는 이다.

Let a=[3]

console.log(a.length); // 결과값 : 1

a[5] = 456; //a[5]라고 하는 것은 a 배열의 다섯번째 값으로 주소를 지정해준 것이다.

console.log(a.length); // 결과값 : 6

```
let a=[3];
console.log(a.length);
a[5] = 456;
console.log(a.length);
console.log(a);
```

지정하지 않은 값은 Null 값으로 비어있는 값이 된다.

[3,null,null,null,null,456]

```
PS C:\work\javascript\11-06> node exm05.js
1
6
[ 3, <4 empty items>, 456 ]
```

```
for(let i=0;i<a.length; i++){
    console.log("a[%d] : %s",i,a[i]);
}
```

```
a[0] : 3
a[1] : undefined
a[2] : undefined
a[3] : undefined
a[4] : undefined
a[5] : 456
```

구문이다.

%d 는 수치의 값, %s 는 문자열 값 출력

```
let a=[3];
console.log(a.length);

a[5]=456;
console.log(a.length);

for( i=0; i<a.length; i++){
    console.log("a[%d] : %s", i, a[i]);}

a[3]=12;
console.log(a.length);

for( i=0; i<a.length; i++){
    console.log("a[%d] : %s", i, a[i]); }
```

a=[3] 은 하나의 값을 배열로 가지고 있는 것.
console.log(a.length)는 길이를 말하는 것으로 1이 나옴

a 배열이 어떻게 되는지는 모르지만
a[5] 주소에 456을 집어넣겠다고 하는 것
console.log(a.length)는 6이나온다 0~5번째값 총 개수는6개.

=> 구문이다.
자료값을 출력해라 i의 값으로
대응값이 %d와 %s인 구문값을 넣어줘서 표현한다.
%d : i의 값을 넣음으로 길이를 말하고 있다.

a[3]에 12값을 넣어주면, 길이는 변하지 않는다
(전체 배열안에서 넣어주는 값이기 때문에)
a[3]의 값만 바뀐다.

결과값 머리로 먼저 시뮬레이션 해보기

결과값 :

```
let a=[3];
console.log(a.length);

a[5]=456;
console.log(a.length);

for( i=0; i<a.length; i++){
    console.log("a[%d] : %s", i, a[i]);}

a[3]=12;
console.log(a.length);

for( i=0; i<a.length; i++){
    console.log("a[%d] : %s", i, a[i]); }
```

```
PS C:\work\javascript> node exm05.js
1
6
a[0] : 3
a[1] : undefined
a[2] : undefined
a[3] : undefined
a[4] : undefined
a[5] : 456
6
a[0] : 3
a[1] : undefined
a[2] : undefined
a[3] : 12
a[4] : undefined
a[5] : 456
```

Undefined 는 자료를 저장할 공간도 없다는 것

Null 자료를 저장할 공간 있지만 값은 아무것도 없다는 것

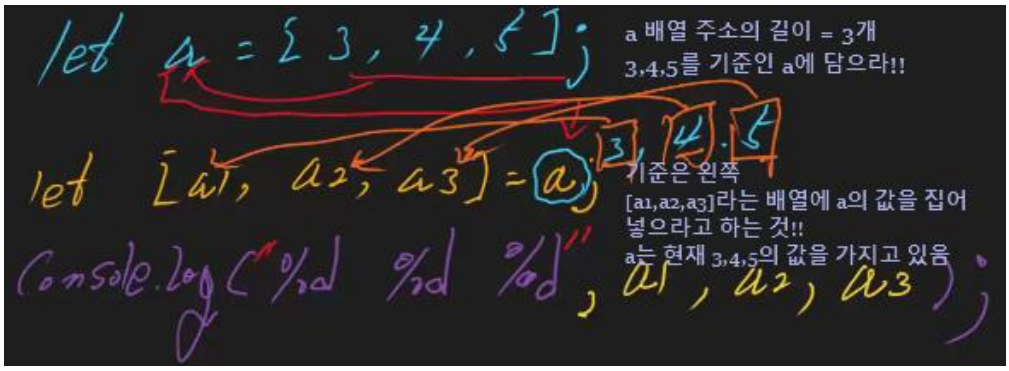
➔ Undefined 로 나오지 않게 하려면 어떻게 해야 할까?

- Let a=[3,” ”,” ,12.” ”,456] 이런식으로 주소의 값을 blank 든 이렇게 주어야 한다 “”로 준 것은 null 값으로 준 것이다.
- 현재 프로그램은 undefined 와 null 의 기준값을 제대로 주지 않았지만 다른 프로그램에서는 undefined 와 null 의 차이가 분명히 있다.

```
let a=[];
console.log(a.length);
```

결과값 : 0

a 는 빈 배열 만들어 놓은 것이다. Length 는 주소의 길이 값으로 0 이 나온다.



a=4;

b=a;

b=? // b 는 4 가 된다 (위의 원리가 이것과 같다)

```
let a=[3,4,5];
let [a1,a2,a3]=a;
console.log("%d %d %d",a1,a2,a3);
```

```
PS C:\work\javascript\11-06> node exm06.js
3 4 5
```

a 의 값을 a1,a2,a3 배열에 넣어서 출력해준 것.

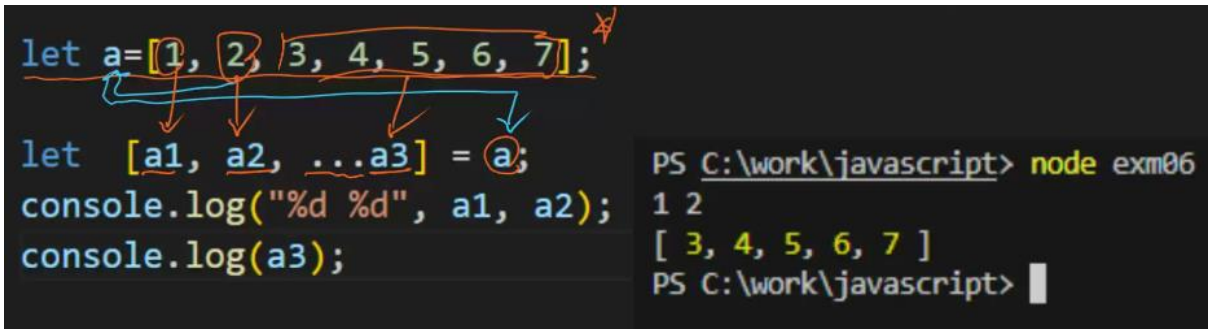
```
[b1,b2]=a;
console.log("%d %d",b1,b2);
```

결과값 : `3 4`

무조건 왼쪽부터 집어넣는다, 그래서 3,4 를 넣어주니 더 넣어줄 것이 없어서 결과는 3,4 가 나온다.

Array 객체(object)는 객체로 묶어주는 것이지만 분해도 가능하다. 위에는 분해하는 과정이고,
분해는 임의로 쪼개는 과정인 것이다.

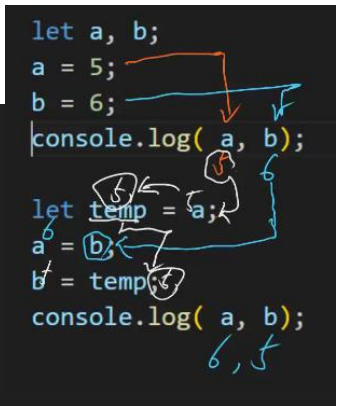
배열의 임의 값 끄집어 낼 때 그 값 중에 특정한 값만 출력하고 싶을 때 배열은 그것이 가능하다.



a3 앞에 ...(점 3 개)는 나머지 주소를 모두 a3 에 넣어준 것. 점 3 개가 없으면 3 만 넣어줬을 것.
a3 는 나머지 값들을 배열로 받았고, 그 값들을 출력했다.
'구조분해 할당'이라고 부른다

<구조분해 문제>

1. a, b 의 값은?




```
let a,b;
a=5;
b=6;
console.log(a,b);

let temp = a;
a=b;
b=temp;
console.log(a,b);
```

a 에 5 가 대입이 되고 temp 에 a 가 넣어지니까 b 의 값은 temp 의 값인 5 가 들어간다.
a 에 b 의 값이 들어가는데, b 는 6 이므로 a 는 6 이 들어간다

2.

```
let a, b;
[a, b] = [5, 6];
console.log(a,b);

[a, b] = [b, a];
console.log(a,b);
```

a 와 b 에 5,6 의 값이 각각 입력된다.

두번째에서는 [a,b]의 값이 [b=6,a=5]가 대입되면서 a=6, b=5 가 된다.

3.

```
let = a=[];
a[0] = 3;
a[1] = 4;
console.log(a);

let temp =a[0]
a[0] = a[1];
a[1] = temp;
console.log(a);
```

4.

```
let i =3, j = 4;
let a = [5, 6, 7];

let a1 = [i, j, a];
console.log(a1.length);
console.log(a1);

let a2 = [i, j, ...a];
console.log(a2.length);
console.log(a2);
```

```
PS C:\work\javascript\11-06> node exm06.js
3
[ 3, 4, [ 5, 6, 7 ] ]
5
[ 3, 4, 5, 6, 7 ]
```

a1 에 a 는 배열로 나와야 한다.

...a 나머지 배열값을 채우라는 것

a1 의 길이는 배열을 한 덩어리라고 보면 3 이 나온다.

a2 의 길이는 배열의 값을 같이 나열했으니까 5 로 배열의 값이 늘어났다.

```
let a=[0, 1, 2, 3];
a.splice(1,0,"a");
console.log(a);
console.log(a.toString());
```

```
PS C:\work\javascript> node exm07
[ 0, 'a', 1, 2, 3 ]
0,a,1,2,3
```

toString : 문자열로 다 바꿔주는 것

Splice(1_1 번주소의 값, 0_값을 0 개 지워라 즉, 안지운다, "a"_a 를 그 자리에 삽입한다)

결과 값에서는 [0,'a',1,2,3]으로 삭제 된 것이 없이 1 번 주소값에 'a'가 삽입이 되었다.

```
a.splice(1,1,"a");
```

결과값 : `[ 0, 'a', 2, 3 ]`

a.splice(1,1,"a");는 1 번 주소의 값을 지우고, a 의 값을 삭제하라는 것.

```
Array.splice(start: number, deleteCount?: number |
undefined): T[]
```

Splice(시작값, 지울지 안지울지 결정하는 값, 삽입값)

slice : 반복

slice(시작값,종료값_위치의 끝나는 지점) // 0 부터 시작할 땐 몇 개, 1 부터시작할 땐 자료위치값

b.slice(0,1) -> 0 부터 시작 , 1 개 출력 => [0] 출력

b.slice(0,2) -> 0 부터 시작 , 2 개 출력 => [0,1] 출력

b.slice(1,2) -> 1 부터 시작 , 2 번째자리까지 출력(b=[1,2,3,4];) => [1] 출력

b.slice(0,1) -> 0 부터 시작 , 3 번째자리까지 출력 => [1,2] 출력

Let b =a.slice(0);

a 를 b 에 복제하라.

-> 인덱스 0 번부터 종료점을 지정하지 않았으므로 전체에 해당

```
let a=[];
for(let i=0;i<10;i++){
    a.push(i);
}

console.log(a);
console.log(a.toString());
```

```
PS C:\work\javascript\11-06> node exm07.js
[
  0, 1, 2, 3, 4,
  5, 6, 7, 8, 9
]
0,1,2,3,4,5,6,7,8,9
```

a 배열에 아무 값 넣어주지 않았다.

0 에서 9 까지 배열로 출력됨 => 값은 10 개 나옴

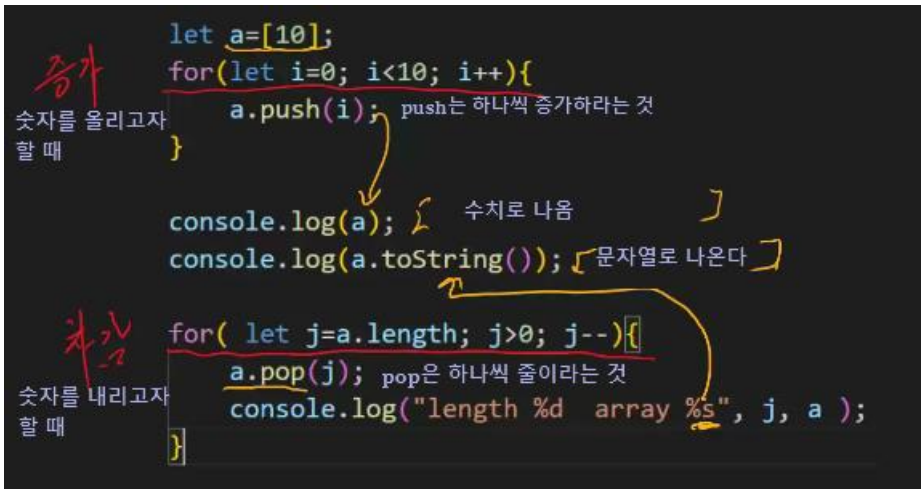
push : 빈 배열에 항목을 추가하는 아이 (증가값)

```
let a=[10];
for(let i=0;i<10;i++){
    a.push(i);
}

console.log(a);
console.log(a.toString());
```

```
PS C:\work\javascript\11-06> node exm07.js
[
  10, 0, 1, 2, 3,
   4, 5, 6, 7, 8,
   9
]
```

a 배열에 10 을 넣어줌 =>10 을 넣고 이후 0,1,2,3,4,5,6,7,8,9 를 넣어줌 => 값은 11 개가 나옴



```
let a=[1,2,3];
let b=[4,5,6];
let c=a+b; //문자열이 됨
let d=a.concat(b); // 배열과 배열을 하나의 배열로 만들어주는 것
                   //원래 있던 배열은 건드리지 않는다
console.log(a);
console.log(b);
console.log(c);
console.log(d);
```

```
PS C:\work\javascript\11-06> node exm07.js
[ 1, 2, 3 ]
[ 4, 5, 6 ]
1,2,34,5,6
[ 1, 2, 3, 4, 5, 6 ]
```

```
let a=[31,11,25,7,1,2,3,13,9];
a.sort();//정렬할거야
console.log(a);
```

```
PS C:\work\javascript\11-06> node exm07.js
[
  1, 11, 13, 2, 25,
  3, 31,  7, 9
]
```

a의 배열을 정렬하여 출력한다는 것.

배열한 값을 보면 숫자의 값 1,2,3,7,9 ~로 정렬되지 않았다.

-> 이 값들은 수치처리가 아닌 문자로 데이터 처리한 것.

-> 기본적으로 string 문자로 데이터 처리한 것

-> 문자로는 2보다 11이 먼저이다/'ㄴ'사이에 'ㄱㄱ'이 있으면 'ㄱㄱ'이 먼저인 것과 같은 것

순서	순서
1	001
11	003
111	011
3	031
31	044
44	066
451	111
66	451
779	779

두개의 차이는 자리수가 다르다
왼쪽도 오른쪽도 3자리 숫자이지만
정렬해주는 것과 안해주는 건 다르다.