ZERONE [0&1]
C Language Basic

영과일 C언어 기초반 #2 - 반복문, 1차원 배열

0&1

컴퓨터학부 2022011812
송진우

1. Remind

```c
#include <stdio.h>

int main() {

    printf("Hello World!");

    return 0;
}
```

출력 (Output)

- printf("[쓰고싶은 말]");
  - 적는 말에 따라 출력되는 말이 달라짐
  - 명령어 뒤에 들어간 ;는 뺴먹지 않기!

```
int A;
int B = 10;
long long C = 21470000009;
char D = 'G';
```

변수 (Variables)

- 데이터를 담을 수 있는 이름이 있는 바구니, 상자
- 담을 수 있는 데이터의 종류를 다르게 할 수 있다.
- 변수를 선언하는 방법도 각양각색.

```
char character = 'A';

int integer = 0;

long long long_integer = 1234567891234;

float floating_point = 0.123;

double double_precision = 0.123456789;
```

자료형 (Variables type)

- char - 문자 하나를 담는 변수 자료형
- int - 정수 자료형
- long long - 더 큰 정수를 담을 수 있는 자료형
- float - 실수를 담을 수 있는 자료형
- double - 실수를 더 정밀하게 담을 수 있는 자료형

```c
int A, B;

scanf("%d %d", &A, &B);

printf("%d", A + B);

return 0;
```

변수 입출력 (Variables I/O)

- 변수를 우선 선언하고
- scanf 함수를 통해서 "변수를 어떻게 받아낼 지" 그리고 "어떤 변수를 저 형식으로 받아낼지" 를 알려주며 입력을 받게 명령합니다.
- printf 함수를 통해서 "변수를 어떻게 출력할지" 그리고 어떤 변수를 출력할지 적어줍니다.
- scanf 뒤에 있는 변수엔 꼭 '&' 를 붙여줍시다!

서식지정자 (format)

	char	int	long long	float	double
서식 지정자	%c	%d	%lld	%f	%lf

+	더하기
-	빼기
*	곱하기
/	나누기
%	나머지

산술 연산자

- 산술 연산자는 항이 두개입니다.
  - 예) A + B, A * B

A = 2	대입 연산자
A += 2	A에 A + 2 를 대입
A -= 2	A에 A - 2 를 대입
A *= 2	A에 A * 2를 대입
A /= 2	A 에 A / 2를 대입
A %= 2	A에 A % 2 를 대입

대입 연산자

- 숫자 자리에 변수가 들어갈 수 있습니다.

A == B	A와 B가 같다면 true 아니라면 false
A != B	A와 B가 다르다면 true 아니라면 false
A > B	A가 B보다 크다면 true 아니라면 false
A >= B	A가 B보다 크거나 같다면 true 아니라면 false
A < B	A가 B보다 작다면 true 아니라면 false
A <= B	A가 B보다 작거나 같다면 true 아니라면 false

관계 연산자

- A와 B의 관계에 따라 값이 달라집니다.

- 조건문 부분에 위 관계 연산자 등, true나 false를 넣습니다.
- 만약 true라면 실행문을 실행시킵니다.
  - 실행문은 한줄이 될 수도, 여러줄이 될 수도 있습니다.

NOT	![A]	[A]조건문이 true라면 false, false라면 true
AND	[A] && [B]	[A]와 [B] 모두 true라면 true, 아니라면 false
OR	[A] || [B]	[A]와 [B] 둘 중 하나라도 true라면 true, 아니라면 false

논리 연산자

- A와 B의 관계에 따라 값이 달라집니다.
- A와 B에는 a==b 와 같은 관계연산자, 혹은 true값 false 값이 들어갈 수 있습니다.

```
int A = 2, int B = 2, int C = 2;

if (A == B && B == C) {
    printf("모든 변수가 같은 값을 가지고 있습니다.");
}
```

- 위와 같이 사용합니다.

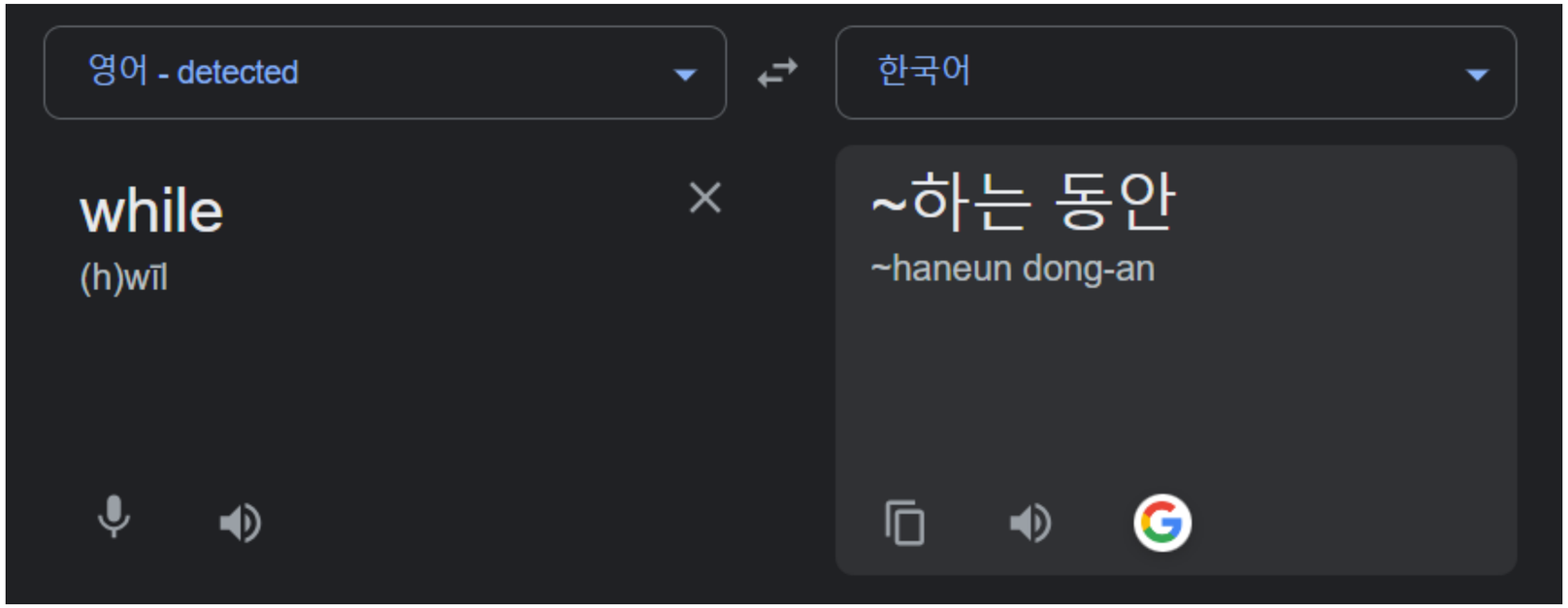
```c
int grade = 72;

if (grade >= 90) {
    printf("A");
}
else if (grade >= 80) {
    printf("B");
}
else if (grade >= 70) {
    printf("C");
}
else if (grade >= 60) {
    printf("D");
}
else {
    printf("F");
}
printf("학점 입니다.");
```

IF-ELSE

- if-else 문에서의 조건은 다음과 같습니다.
  - if문은 반드시 한개 존재해야 합니다.
  - else-if문은 몇개든 상관 없습니다.
  - else문은 없거나 한개만 존재해야 합니다.
- 각각의 조건문은, 본인의 조건문 위의 조건문이 false여서 내려왔을 때 조건을 체크합니다.
- if문에 포함되지 않은 코드는 조건문이 없으므로 그냥 실행됩니다.

- 옆과 같이 사용합니다.

2. While 반복문



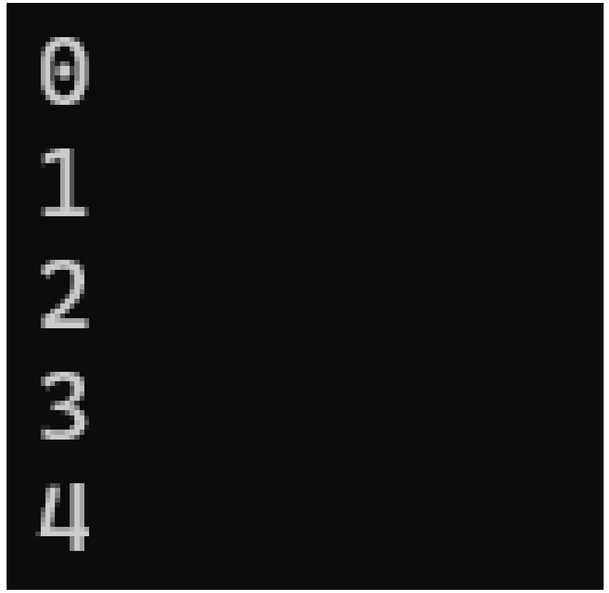
반복 (Repeat)

- ~하는 동안 멈추지 않고 실행을 하는 반복문
  - 그렇다면 어떤 경우에 실행하는지
  - 어떤 것들을 실행하는지가 필요합니다.

CODE

```c
int i = 0;

while (i < 5) {
    printf("%d₩n", i);
    i += 1;
}
```

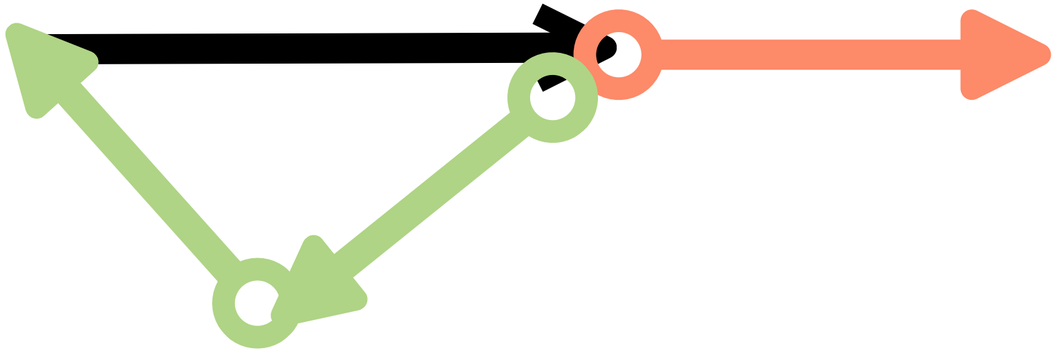
OUTPUT



- i라는 변수가 5보다 작을경우 (0,1,2,3,4)
  - printf로 i의 값을 출력하고
  - i의 값에다가 1을 더하는것을
- 반복하는 코드

- 따라서, 결과는 0, 1, 2, 3, 4로 i가 5가 된 순간부터 반복은 이루어지지 않습니다.

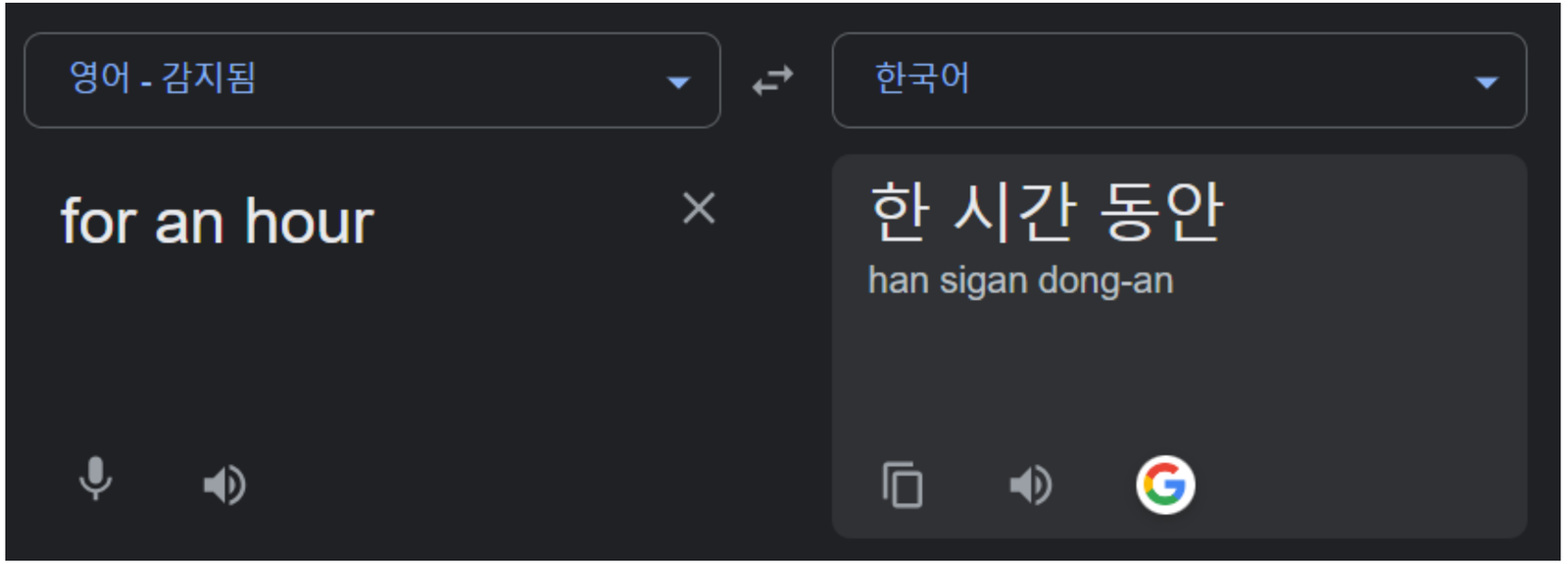
```
while (    조건문    ) {

    실행문

}
```



- 조건문이 true라면
  - 실행문에 적힌 명령들을 실행합니다

- 이후 다시 처음으로 돌아와, 조건문을 확인하고 true라면 이 과정을 반복합니다.

2. For 반복문



반복 (Repeat)

- ~하는 동안 멈추지 않고 실행을 하는 반복문
  - 그렇다면 어떤 경우에 실행하는지
  - 어떤 것들을 실행하는지가 필요합니다.

CODE

```c
int i;
for (i = 0; i < 10; i++) {
    printf("%d₩n", i);
}
```

- i=0을 실행시키기
- i < 10인지 체크하고
- 실행문 부분에 적힌 명령을 실행
- i++을 실행

- i가 0으로 설정되고
  - i가 10보다 작다면
    - printf()함수로 i의 값을 출력하고
    - i에다 1을 추가함
  - i가 10을 초과하는 순간 for문 종료.

OUTPUT

```
0
1
2
3
4
5
6
7
8
9
```

- 따라서, i가 10인 결과는 나오지 않습니다.

```
int i;

for (초기식; 조건식; 증감식) {

  실행문


}
```

- 초기식을 실행한 뒤, 조건식이 true라면 실행문에 적힌 명령들이 작동되고, 증감식을 실행하는것으로 1회 반복됩니다.
- 해당 반복은 조건식이 false가 될 때 까지 반복됩니다.

```
int i;
for (i = 0; i < 10; i++) {
    printf("%d₩n", i);
}
```

- 위 코드와 아래 코드는 같은 동작과 같은 결과를 가집니다.
- 위 코드는 for문으로, 아래 코드는 while문으로 작성되었습니다.

```
int i = 0;
while (i < 10) {
    printf("%d₩n", i);
    i++;
}
```

```c
for (i = 0; i < 10; i += 2) {
    printf("%d₩n", i);
}

for (i = 1; i < 250; i *= 2) {
    printf("%d₩n", i);
}
```

- 옆과 같이, 증감식에 i++ 이 아닌 다른 식을 넣을 수 있습니다.
- 증감의 폭을 다르게 할 수 있습니다.

- 옆과 같이, 초기식, 조건식, 증감식 모두 필수가 아닌 필요에 따라 적을 수 있습니다.
- 모두 생략한다면, 조금 있다 배울, break를 활용하여 반복문을 종료해 주어야 합니다.
    - 그렇지 않다면 영원히 끝나지 않는 프로그램이 만들어집니다.

```c
for (;;) {
    if (i == 5) break;
    printf("%d₩n", i);
    i++;
}
for (;i!=5;) {
    printf("%d₩n", i);
    i++;
}
```

```c
for (i = 0; i < 5; i++) {
    if (i == 2) break;
    printf("%d₩n", i);
}
```

```
0
1
```

- break는 말 그대로, 반복문을 종료하라는 명령
- 옆 코드에서는 i가 2가 될 때 반복문을 강제로 종료
  - 따라서, printf도 실행되지 않고 프로그램이 종료됨.

```c
for (i = 0; i < 5; i++) {
    if (i == 2) continue;
    printf("%d₩n", i);
}
```

```
0
1
3
4
```

- continue는 말 그대로, 계속해서 반복문을 실행시키라는 명령
  - 단, continue 명령을 받은 순간 진행되던 실행문은 skip!
  - 즉, i가 2가 된 순간 printf명령은 무시하고 다시 증감식을 받아 i는 3이 되고, 3이 출력된 것.

3. 문제 풀이

- **2739: 구구단**
- **2438: 별 찍기 - 1**

	char	int	long long	float	double
서식 지정자	%c	%d	%lld	%f	%lf

NOT	![A]	[A]조건문이 true라면 false, false라면 true
AND	[A] && [B]	[A]와 [B] 모두 true라면 true, 아니라면 false
OR	[A] || [B]	[A]와 [B] 둘 중 하나라도 true라면 true, 아니라면 false

+	더하기	/	나누기	scanf()	& 붙이기
-	빼기	%	나머지	printf()	& 안붙이기
*	곱하기	A != B	다르다	\n	줄바꿈

```c
int main() {

    int N;

    scanf("%d", &N);

    int i;
    for (i = 1; i < 10; i++) {
        printf("%d * %d = %d₩n", N, i, N * i);
    }

    return 0;
}
```

- 사용자로부터 N을 입력받습니다.

- i를 1부터 9까지 천천히 하나씩 증가시키며
  - N * i = N*i 를 출력시킵니다.
  - 따라서 "%d * %d = %d" 로 하고
    - N, i, N * i를 차례로 넣게 합니다.

```
int i, j;
for (i = 1; i <= N; i++) {
    for (int j = 0; j < i; j++) {
        printf("*");
    }
    printf("₩n");
}
```

- for문은 중첩이 가능합니다.
  - 가장 바깥쪽 for문의 변수인 i를 가지고 안쪽 for문이 j라는 변수를 증감시키면서 반복을 진행합니다.
  - 이는 당연히 3중 4중 반복도 가능하며, 반복되면 반복될수록 실행 횟수는 기하 급수적으로 증가합니다.

2442: 별찍기 - 5

- 조금 더 생각을 해보아야 할 문제

수업 진행 방식 변경

- 조금씩 C언어에 적응해 가시고 있고 계신 것으로 믿고, 수업의 진행 방향을 바꿔보려 합니다.
  - 기존: 사용법을 학습 후, 문제를 보고 풀이
  - 변경: 문제를 보고 풀어본 후, 새로운 지식을 익히고 다시 풀이

- 기본적인 문법은 다 학습 하셨기에 충분히 시간을 들여 풀어보시고 해설을 들어주셨으면 좋겠습니다.

IDE 설정

- MAC
- https://nadocoding.tistory.com/95
- WINDOW
- https://akdl911215.tistory.com/367

4. 배열 [ARRAY]

개수 세기 성공

브론즈 V

- **10807: 개수세기**

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
1 초	256 MB	122584	75379	64222	62.484%

문제

총 N개의 정수가 주어졌을 때, 정수 v가 몇 개인지 구하는 프로그램을 작성하시오.

입력

첫째 줄에 정수의 개수 N(1 ≤ N ≤ 100)이 주어진다. 둘째 줄에는 정수가 공백으로 구분되어져있다. 셋째 줄에는 찾으려고 하는 정수 v가 주어진다. 입력으로 주어지는 정수와 v는 -100보다 크거나 같으며, 100보다 작거나 같다.

출력

첫째 줄에 입력으로 주어진 N개의 정수 중에 v가 몇 개인지 출력한다.

문제에 차근차근 접근

- 우선, N개의 정수를 입력 받아야 무언가 진행될테니, 우선 N개의 정수를 입력받아봅시다.

```c
int N;
scanf("%d", &N);
```

```c
int i;
for (i = 0; i < N; i++) {
    scanf("%d", &...?);
}
```

- 다음으로, N개 만큼의 수를 입력받아야 하니까, 반복문으로 입력을 받아보려고 시도해봅시다.
  - N번 반복문을 돌리는것은 할 수 있는데, 어디에다가 받은 수를 저장해야하지...?

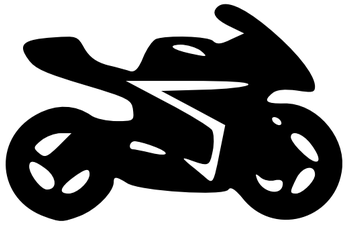
입력으로 주어지는 정수와 v는 **-100**보다 크거나 같으며, **100**보다 작거나 같다.

  - count_-100 부터 count_100까지 201개의 변수를 선언해서 count를 올릴까요...?

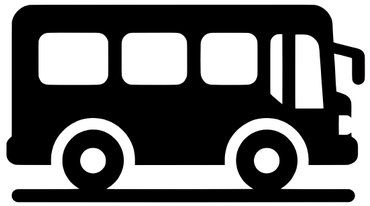
- 아직 v를 입력조차 받지 못했는데, 변수 저장부터 막힌 상황에 해당 상황을 해결할만한 해결책이 있을까요?
  - 뭔가 순서대로 저장되는 변수같은건 없을까?
  - 뭔가 변수를 한꺼번에 여러개 선언할 순 없을까?
  - 아니면 뭔가 다른 방법이 있을까?

배열 [Array]

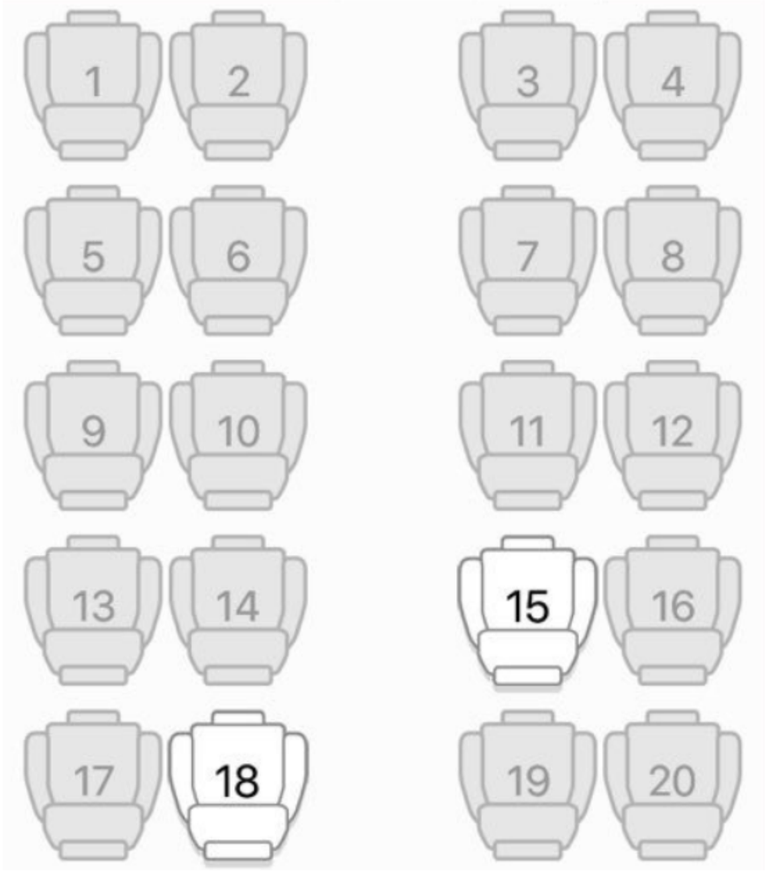
- 기존에 알고있던 변수가 1인용 오토바이라고 생각해봅시다.
- 각 오토바이에 탈 수 있는 사람의 종류는 정해져 있습니다. (int, char, long long ...)
- 배열은, 이 오토바이가 그저 "버스" 로 바뀐것이라고 생각하시면 됩니다.



```
int N = 10;
```

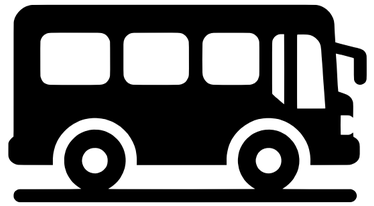


```
int arr[1001];
```



배열 [Array]

- 배열은, 다음과 같은 성질을 가집니다.
  - 배열은 한줄로 나열되어있는 데이터들이라고 생각해도 됩니다.
  - 나열되어있는 데이터들은 순서가 지정되어있고, 각 순서에 저장되어있는 데이터를 '원소' 라고 합니다.
  - 각 원소는 '0번' 부터 'N-1번' 까지 번호가 있고, 번호로 각 원소에 접근할 수 있습니다.



```
int arr[1001];
```

1	2	4	8	16	32

```
arr[2]
```

```
자료형 변수이름[배열크기] = {배열 원소 목록}
```

배열 선언, 정의 (declaration / definition)

```
변수이름[번호]
```

원소 접근 (Access)

자, 그럼 배열을 통해 해결해봅시다.

- N개의 수를 담아낼 수 있는 배열을 만들고, 각 원소에 각 수를 저장해봅시다.
- N의 범위가 1부터 100까지 이므로, 배열의 크기를 101로 해두면 충분할 것입니다.

```
int i;
int arr[101];
for (i = 0; i < N; i++) {
    scanf("%d", &arr[i]);
}
```

```
int v;
scanf("%d", &v);
```

- 다음으로, v를 사용자로부터 입력 받아서, 저장받은 배열에서 v를 찾을 준비를 해봅시다.

```
int count = 0;
for (i = 0; i < N; i++) {
    if (arr[i] == v) count++;
}
```

- 다음으로, count라는 변수를 준비해주고, i를 0부터 N-1까지 움직여 가면서...
  - arr의 i번째 원소에 접근해서, v와 같다면 count를 하나씩 올려줍니다
  - 그말인 즉, 버스 좌석 하나하나 확인하며 성이 '김'인 사람을 찾으면 count를 하나 올려서, 버스에 탄 '김'인 사람의 수를 찾는것이라고도 볼 수 있습니다.

```
printf("%d", count);
```

- 그리고, count를 출력하면 arr안의 v의 개수를 찾은것을 출력할 수 있게 됩니다.

5. 문제 풀이

- 10871: X보다 작은 수
- 3052: 나머지

	char	int	long long	float	double
서식 지정자	%c	%d	%lld	%f	%lf

NOT	![A]	[A]조건문이 true라면 false, false라면 true
AND	[A] && [B]	[A]와 [B] 모두 true라면 true, 아니라면 false
OR	[A] || [B]	[A]와 [B] 둘 중 하나라도 true라면 true, 아니라면 false

+	더하기	/	나누기	scanf()	& 붙이기
-	빼기	%	나머지	printf()	& 안붙이기
*	곱하기	A != B	다르다	\n	줄바꿈

```c
#include <stdio.h>

int A[10001];

int main() {

    int N, X;

    scanf("%d %d", &N, &X);

    int i;
    for (i = 0; i < N; i++) {
        scanf("%d", &A[i]);
    }

    for (i = 0; i < N; i++) {
        if(A[i] < X) printf("%d ", A[i]);
    }

    return 0;
}
```

- 사용자로부터 우선 N과 X를 받아와서
- N의 크기는 최대 10000이므로, A 배열의 크기의 크기도 10001으로 설정해둡니다.
  - A[N+1] 과 같이 변수가 들어간 선언은 허용하지 않는 컴파일러도 있습니다.
- 다음으로, N번 루프를 돌며, A에 값들을 저장합니다.
- 그런 뒤, X와 A의 i번째 원소를 비교하면서 i를 N-1까지 이동시키며 반복하여 조건에 따라 A의 i번째 원소를 출력해줍니다.

```c
int arr[10001];

int main() {

    int A, i;

    for (i = 0; i < 10; i++) {
        scanf("%d", &A);
        arr[A % 42]++;
    }

    int count = 0;
    for (i = 0; i < 42; i++) {
        if (arr[i] != 0) count++;
    }

    printf("%d", count);

    return 0;
}
```

- A를 받아서 0부터 41번째까지의 arr 배열에 각 번호수 나머지만큼 1을 더합니다.
- 각 원소에서...
  - 0은 해당하는 나머지가 나오지 않았다는 것. 즉, 차운트를 안늘립니다.
  - 1이상이면 카운트를 하나만 늘립니다. 2든 3이든 다른걸 체크해야 하므로 하나만 늘립니다.
- count를 출력하면 나머지값을 출력합니다.