

브루트 포스 - 재귀

최백준 choi@startlink.io

1, 2, 3 더하기

<https://www.acmicpc.net/problem/9095>

2

$$N \leq 10$$

- 정수 n 을 1, 2, 3의 합으로 나타내는 방법의 수를 구하는 문제

- $n = 4$

- $1+1+1+1$

- $1+1+2$

- $1+2+1$

- $2+1+1$

- $2+2$

- $1+3$

- $3+1$

Handwritten diagram illustrating the problem: a sequence of circles representing the sum of 10 ones, with a bracket underneath labeled "10개" (10 times). To the right is an equals sign followed by a circle containing the number 10. Below this is another circle containing 3^{10} followed by an equals sign.

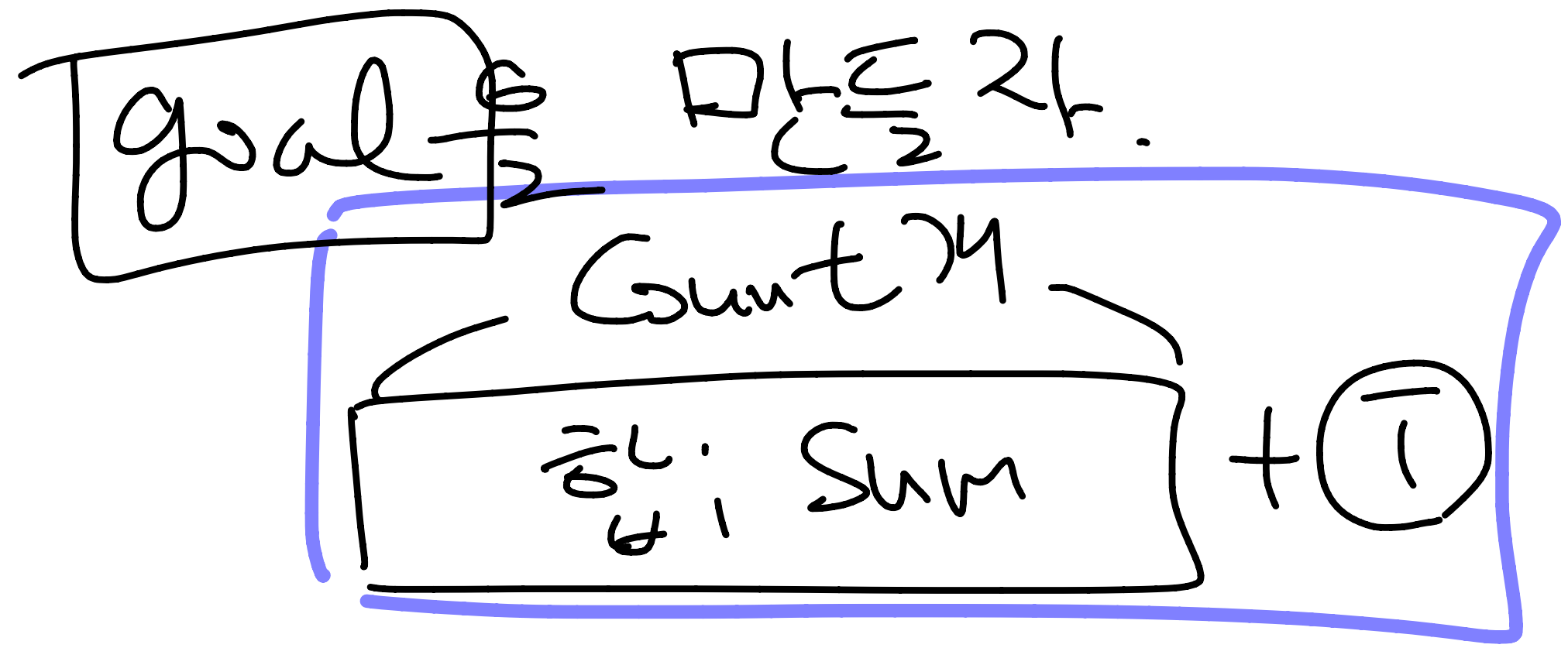
1, 2, 3 더하기

<https://www.acmicpc.net/problem/9095>

- $n \leq 10$ 이기 때문에
- 총 경우의 수는 3^n 이다.

1, 2, 3 더하기

<https://www.acmicpc.net/problem/9095>



- go(count, sum, goal)
- 숫자 count개로 합 sum을 만드는 경우의 수

Count → Count + 1
Sum → Sum + 1

① 불가능한 경우
Sum > goal.

② 정답을 찾는 경우
Sum == goal

③ 재귀 호출
① go(count + 1, Sum + 1, goal)

1, 2, 3 더하기

5

<https://www.acmicpc.net/problem/9095>

- `go(count, sum, goal)`
- 숫자 count개로 합 sum을 만드는 경우의 수
- 불가능한 경우
 - `sum > goal`
- 정답을 찾은 경우
 - `sum == goal`

1, 2, 3 더하기

<https://www.acmicpc.net/problem/9095>

- `go(count, sum, goal)`
- 숫자 count개로 합 sum을 만드는 경우의 수
- 다음 경우
 - 1을 사용하는 경우
 - `go(count+1, sum+1, goal)`
 - 2를 사용하는 경우
 - `go(count+1, sum+2, goal)`
 - 3을 사용하는 경우
 - `go(count+1, sum+3, goal)`

1, 2, 3 더하기

<https://www.acmicpc.net/problem/9095>

```
int go(int count, int sum, int goal) {
    if (sum > goal) return 0;
    if (sum == goal) return 1;
    int now = 0;
    for (int i=1; i<=3; i++) {
        now += go(count+1, sum+i, goal);
    }
    return now;
}
```

불가능한 경우 :

재귀 호출을 계속해서
정답을 찾을 수
없는 경우

또는

문자열 조합을 취하는 경우

1, 2, 3 더하기

<https://www.acmicpc.net/problem/9095>

- 다 만들고 나서 보니 count는 별로 의미가 없다.

1, 2, 3 더하기

<https://www.acmicpc.net/problem/9095>

```
int go(int sum, int goal) {  
    if (sum > goal) return 0;  
    if (sum == goal) return 1;  
    int now = 0;  
    for (int i=1; i<=3; i++) {  
        now += go(sum+i, goal);  
    }  
    return now;  
}
```

1, 2, 3 더하기

<https://www.acmicpc.net/problem/9095>

- 소스: <http://codeplus.codes/0dfb2b005a6249c297748a3def93e4d2>

암호 만들기

<https://www.acmicpc.net/problem/1759>

- 암호는 서로 다른 L 개의 알파벳 소문자들로 구성되며 최소 한 개의 모음과 최소 두 개의 자음으로 구성되어 있다
- 암호를 이루는 알파벳이 암호에서 증가하는 순서로 배열되었어야 한다
- 암호로 사용할 수 있는 문자의 종류는 C 가지
- 가능성 있는 암호를 모두 구하는 문제

$$3 \leq L \leq C \leq 15$$

암호 만들기

<https://www.acmicpc.net/problem/1759>

• $L = 4, C = 6$

• 사용 가능한 알파벳: `a t c i s w`

• 가능한 암호

• acis

• acit

• aciw

• acst

• acsw

• actw

• aist

알파벳 정렬

C가 2

a	c	i	s	t	w
o	o	o			o
x	x	x			x



이 4

• aisw

• aitw

• astw

• cist

• cisw

• citw

• istw

암호 만들기

<https://www.acmicpc.net/problem/1759>

13

• `go(n, alpha, password, i)`

- `n`: 만들어야 하는 암호의 길이
- `alpha`: 사용할 수 있는 알파벳 네벌
- `password`: 현재까지 만든 암호
- `i`: 사용할지 말지 결정해야 하는 알파벳의 인덱스

① 정답을 찾는 경우

password의 길이 == n

② 불가능한 경우

$i \geq \text{alpha의 크기}$

③ 다음 경우

$\text{password} + \text{alpha}[i], i+1$

1

i 번째를 사용: $\text{go}(n, \text{alpha}, \text{password} + \text{alpha}[i], i+1)$

2

i 번째를 사용하지 않음: $\text{go}(n, \text{alpha}, \text{password}, i+1)$

암호 만들기

<https://www.acmicpc.net/problem/1759>

- `go(n, alpha, password, i)`
 - `n`: 만들어야 하는 암호의 길이
 - `alpha`: 사용할 수 있는 알파벳
 - `password`: 현재까지 만든 암호
 - `i`: 사용할지 말지 결정해야 하는 알파벳의 인덱스
- 정답을 찾은 경우 (문제의 조건에 맞는지 확인 과정은 여기서 필요함)
 - `n == password.length()`
- 불가능한 경우
 - `i >= alpha.size()`

암호 만들기

<https://www.acmicpc.net/problem/1759>

- 다음 경우
 - i번째 알파벳을 사용하는 경우
 - `go(n, alpha, password+alpha[i], i+1)`
 - i번째 알파벳을 사용하지 않는 경우
 - `go(n, alpha, password, i+1)`

암호 만들기

<https://www.acmicpc.net/problem/1759>

16



```
void go(int n, vector<char> &alpha, string password, int i) {
```

```
    if (password.length() == n) {  
        if (check(password)) {  
            cout << password << '\n';  
        }  
        return;  
    }
```

```
    if (i >= alpha.size()) return;  
    go(n, alpha, password+alpha[i], i+1);  
    go(n, alpha, password, i+1);
```

```
}
```

$$2^{n=C} = 2^C$$

암호 만들기

<https://www.acmicpc.net/problem/1759>

```
bool check(string &password) {  
    int ja = 0;  
    int mo = 0;  
    for (char x : password) {  
        if (x == 'a' || x == 'e' || x == 'i' || x == 'o' || x ==  
'u') {  
            mo += 1;  
        } else {  
            ja += 1;  
        }  
    }  
    return ja >= 2 && mo >= 1;  
}
```

암호 만들기

<https://www.acmicpc.net/problem/1759>

- 소스: <http://codeplus.codes/f38ae1748ce24e85816698676b4cc6cf>

퇴사

<https://www.acmicpc.net/problem/14501>

19

- N+1일이 되는 날 퇴사를 하려고 한다 ($1 \leq N \leq 15$)
- 남은 N일 동안 최대한 많은 상담을 하려고 한다
- 하루에 하나의 상담을 할 수 있고
- i일에 상담을 하면, T[i]일이 걸리고 P[i]원을 번다

N=5

	1	2	3	4	5	6
T	1	2	3	4	5	6
P	2	5	3	3	100	
	0	0	0	0	0	
	X	X	X	X	X	

2^N

2¹⁵ = 32768

퇴사

<https://www.acmicpc.net/problem/14501>

20

① 정답을 찾는 경우

day == n+1

• go(day, sum)

• day일이 되었다. day일에 있는 상담을 할지 말지 결정해야 한다.

• 지금까지 얻은 수익은 sum이다

② 불가능한 경우

day > n+1

③ 다음 경우 고려

1

상담을 한다

: go(day + T[day], sum + p[day])

2

상담을 하지 않는다

: go(day + 1, sum)

퇴사

<https://www.acmicpc.net/problem/14501>

$$\begin{array}{rcl} 1 \sim N \text{ 일} & N+1 \text{ 일} & \\ \hline 0 \sim (N-1) \text{ 일} & N \text{ 일} & \end{array}$$

- go(day, sum)
 - day일이 되었다. day일에 있는 상담을 할지 말지 결정해야 한다.
 - 지금까지 얻은 수익은 sum이다
- 정답을 찾은 경우
 - $\text{day} = n$
- 불가능한 경우
 - $\text{day} > n$
- 다음 경우
 - 상담을 한다: go(day+t[day], sum+p[day])
 - 상담을 하지 않는다: go(day+1, sum)

퇴사

<https://www.acmicpc.net/problem/14501>

- 소스: <http://codeplus.codes/fdc0c3b04589497995d3035b94b764c2>

백트래킹

백트래킹

Backtracking

- 재귀 함수를 이용해 브루트 포스를 하다 보면, 더 이상 함수 호출이 의미 없는 경우가 있다.
- 이 때, 이런 경우를 제외하고 브루트 포스를 진행하면 백트래킹이라고 한다.

스타트와 링크

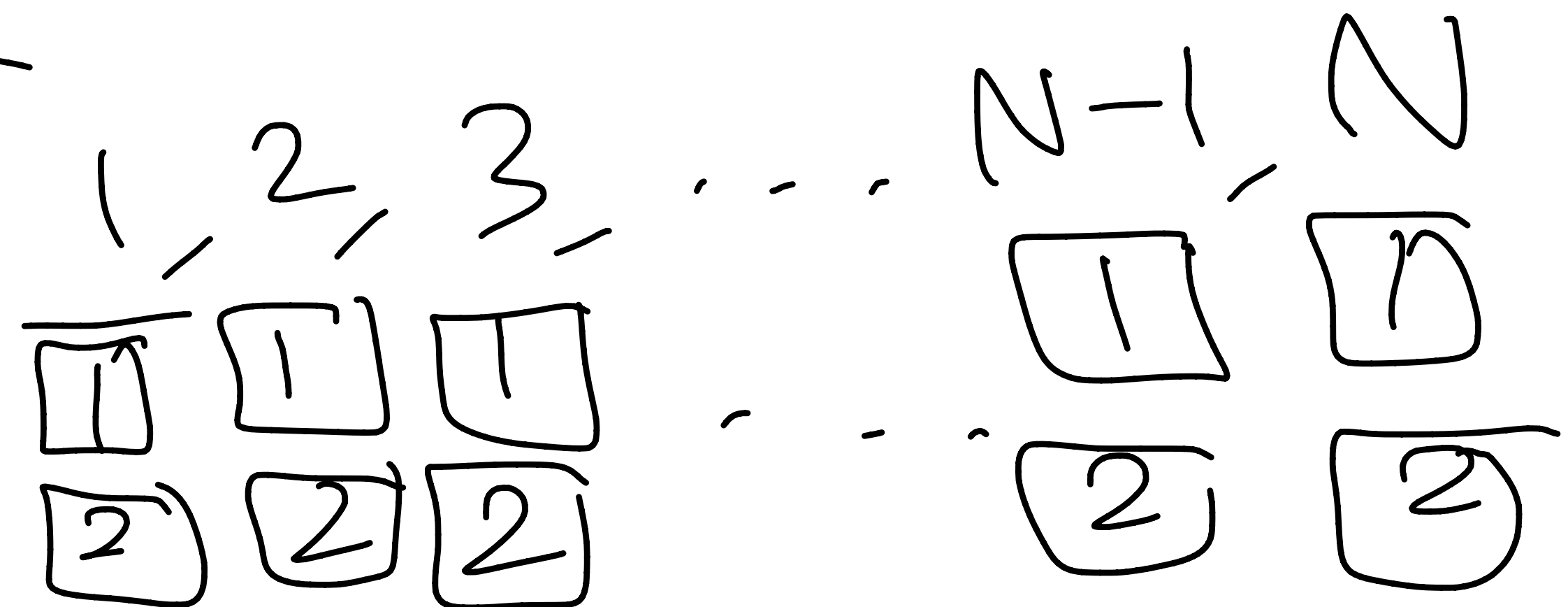
<https://www.acmicpc.net/problem/14889>

능력치 : 2^N

25

- N명을 $N/2$ 명씩 두 팀으로 나누려고 한다. ($4 \leq N \leq 20$, N은 짝수)
- 두 팀의 능력치를 구한 다음, 차이의 최소값을 구하는 문제
- $S[i][j]$ = i번 사람과 j번 사람이 같은 팀에 속했을 때, 팀에 더해지는 능력치
- 팀의 능력치: 팀에 속한 모든 쌍의 $S[i][j]$ 의 합

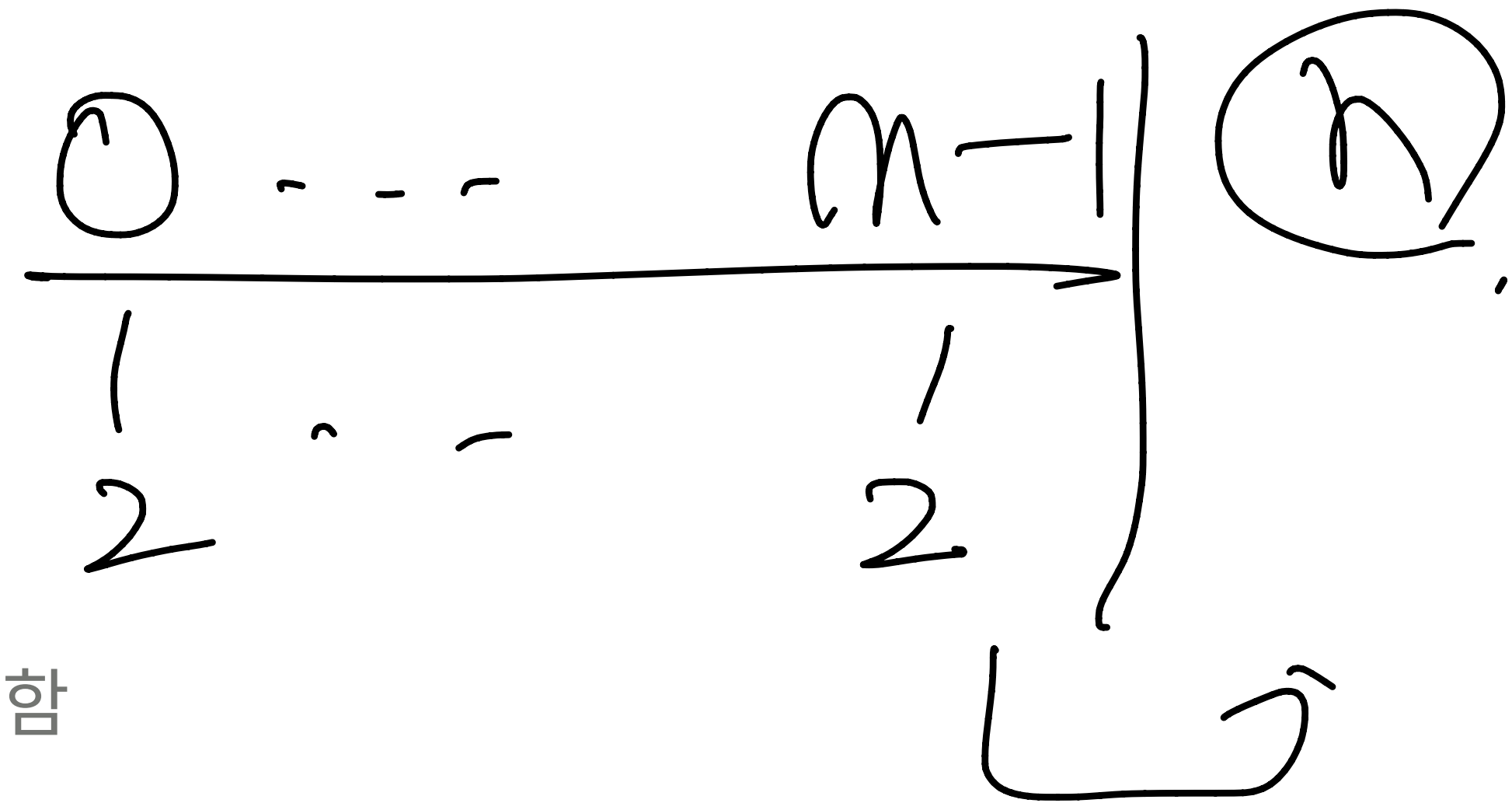
$$2^{20} \leq 1048576$$



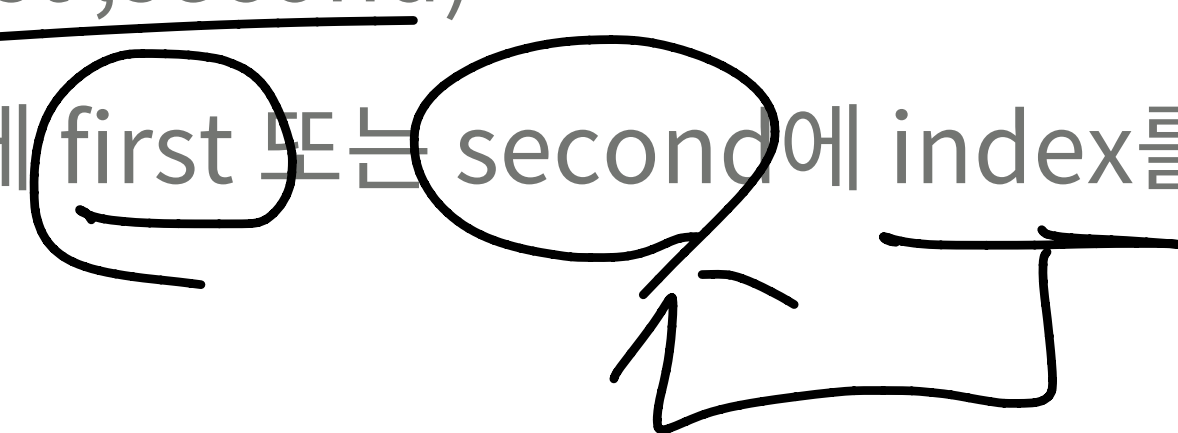
스타트와 링크

<https://www.acmicpc.net/problem/14889>

26



- go(index, first, second)
 - index번째 사람을 어떤 팀에 넣을지 결정해야 함
 - 1번 팀과 2번 팀에 속한 사람이 각각 first, second에 들어 있음
- 정답을 찾은 경우
 - $index == n$
- 다음 경우
 - 1번 팀: go(index, first, second)
 - 2번 팀: go(index, first, second)
 - 두 경우 모두 호출 전에 first 또는 second에 index를 넣고, 호출 후에 빼는 과정이 필요



스타트와 링크

<https://www.acmicpc.net/problem/14889>

- 소스: <http://codeplus.codes/251534f6856f40649b867c834d1e0992>

스타트와 링크

807

28

<https://www.acmicpc.net/problem/14889>

- go(index, first, second)
 - index번째 사람을 어떤 팀에 넣을지 결정해야 함
 - 1번 팀과 2번 팀에 속한 사람이 각각 first, second에 들어 있음
- 정답을 찾은 경우
 - index == n
- 불가능한 경우
 - first의 크기 > n/2
 - second의 크기 > n/2
- 다음 경우
 - 1번 팀: go(index, first, second)
 - 2번 팀: go(index, first, second)
 - 두 경우 모두 호출 전에 first 또는 second에 index를 넣고, 호출 후에 빼는 과정이 필요

1 2 3 4 5 6 | 7 8
1 2 1 1 1 1

112 : 507

24 : 307 ↓

스타트와 링크

<https://www.acmicpc.net/problem/14889>

- 소스: <http://codeplus.codes/556157a8419d424a86d3e14e2273600b>

링크와 스타트

<https://www.acmicpc.net/problem/15661>

- N명을 두 팀으로 나누려고 한다. ($4 \leq N \leq 20$)
- 두 팀의 능력치를 구한 다음, 차이의 최소값을 구하는 문제
- $S[i][j] = i$ 번 사람과 j 번 사람이 같은 팀에 속했을 때, 팀에 더해지는 능력치
- 팀의 능력치: 팀에 속한 모든 쌍의 $S[i][j]$ 의 합

링크와 스타트

<https://www.acmicpc.net/problem/15661>

- go(index, first, second)
 - index번째 사람을 어떤 팀에 넣을지 결정해야 함
 - 1번 팀과 2번 팀에 속한 사람이 각각 first, second에 들어 있음
- 정답을 찾은 경우
 - index == n
- 다음 경우
 - 1번 팀: go(index, first, second)
 - 2번 팀: go(index, first, second)
 - 두 경우 모두 호출 전에 first 또는 second에 index를 넣고, 호출 후에 빼는 과정이 필요

링크와 스타트

<https://www.acmicpc.net/problem/15661>

- 소스: <http://codeplus.codes/1e88b5849dc94e2a8fc5c9f2c321ea7e>

부등호

$$A = [>, <, <, >]$$

33

<https://www.acmicpc.net/problem/2529>

- 부등호 기호 <와 >가 나열된 수열 A가 있다
- 기호의 앞 뒤에 한 자리 숫자를 넣어서 모든 부등호 관계를 만족시키려고 한다
- 이 때, 선택된 수는 모두 달라야 한다
- k개의 부등호 관계를 모두 만족시키는 (k+1)개의 자리의 정수 중에서 최대값과 최소값을 구하는 문제

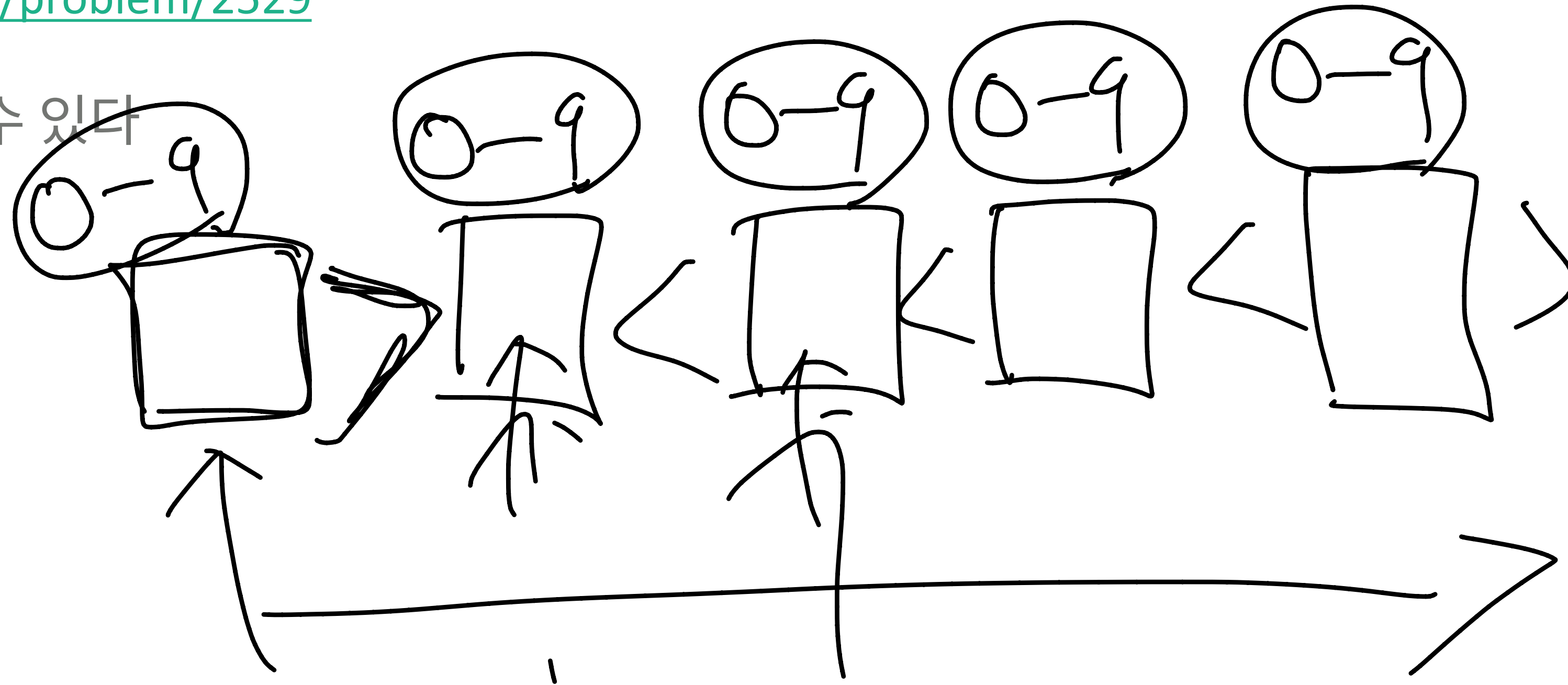
92351
~~2~~

$$\begin{array}{ccccccc} 9 & > & 2 & < & 3 & < & 5 & > & 1 \\ - & & - & & - & & - & & - \\ 9 & > & 5 & < & 6 & < & 7 & > & 2 \end{array}$$

부등호

<https://www.acmicpc.net/problem/2529>

- 재귀함수를 이용할 수 있다



부등호

<https://www.acmicpc.net/problem/2529>

35

```
void go(int index, string num) {  
    if (index == n+1) {  
        if (ok(num)) {  
            ans.push_back(num);  
        }  
        return;  
    }  
    for (int i=0; i<=9; i++) {  
        if (check[i]) continue;  
        check[i] = true;  
        go(index+1, num+to_string(i));  
        check[i] = false;  
    }  
}
```

Index 4227

0-9 4227

오류만 4227 x

부등호

<https://www.acmicpc.net/problem/2529>

- 소스: <http://codeplus.codes/745b86e0551d4d9b90898bd25bd5da7f>

부등호

<https://www.acmicpc.net/problem/2529>

- 부등호 기호를 만족하는지를 가장 마지막에 모든 수를 결정하고 검사하고 있다
- 첫 번째 수가 9이고, 두 번째 수가 8인 경우에는
- $9 < 8$ 을 만족하지 않기 때문에, 뒤에 무엇이 온다고 해도 절대로 정답이 될 수 없다
- 하지만 앞 페이지의 소스는 계속해서 검사를 진행한다.

5 ~~7~~ 6 < 7 < 8 < 9 > 1

부등호

<https://www.acmicpc.net/problem/2529>

- 함수의 호출 중간에 절대로 정답이 될 수 없는 경우를 발견하면
- 그 뒤의 호출을 더 이상 진행하지 않아도 된다

부등호

39

<https://www.acmicpc.net/problem/2529>

```
void go(int index, string num) {  
    if (index == n+1) {  
        ans.push_back(num);  
        return;  
    }  
    for (int i=0; i<=9; i++) {  
        if (check[i]) continue;  
        if (index == 0 || good(num[index-1], i+'0', a[index-1])) {  
            check[i] = true;  
            go(index+1, num+to_string(i));  
            check[i] = false;  
        }  
    }  
}
```

Handwritten annotations:

- A box with an arrow pointing to the `index` parameter in the `go` function signature.
- The text `index 0` with an arrow pointing to the `index == 0` condition.
- The text `0/22` next to the `for` loop.
- A box around `(23)` next to the `for` loop.
- The text `결론` (Conclusion) next to the `for` loop.
- A box around `i+'0'` in the `good` function call.
- A box around `a[index-1]` in the `good` function call.
- A box around the `if` condition `if (index == 0 || good(num[index-1], i+'0', a[index-1]))`.
- A box around the `check[i] = true;` line.
- A box around the `go(index+1, num+to_string(i));` line.
- A box around the `check[i] = false;` line.

부등호

40

<https://www.acmicpc.net/problem/2529>

```
bool good(char x, char y, char op) {  
    if (op == '<') {  
        if (x > y) return false;  
    }  
    if (op == '>') {  
        if (x < y) return false;  
    }  
    return true;  
}
```


부등호

<https://www.acmicpc.net/problem/2529>

- 소스: <http://codeplus.codes/210b99c6ff4f4e55a84e2f46d8f603b0>

- C++: 920 ms → 8 ms
- Java: 1260 ms → 196 ms

맞춰봐

21개

42

<https://www.acmicpc.net/problem/1248>

- -10부터 10까지 N개의 정수(중복 없음)로 이루어진 수열 A가 있다. ($N \leq 10$)
- $S[i][j] = A[i] + A[i+1] + \dots + A[j]$ 가 0보다 크면 +, 작으면 -, 같으면 0
- S가 주어졌을 때, 가능한 A를 아무거나 찾는 문제

흔히

$A = [3, -2, -1, 5]$

S

i \ j	0	1	2	3
0	0	+	+	+
1		X	-	+
2		X	X	+
3		X	X	+

맞춰봐

<https://www.acmicpc.net/problem/1248>

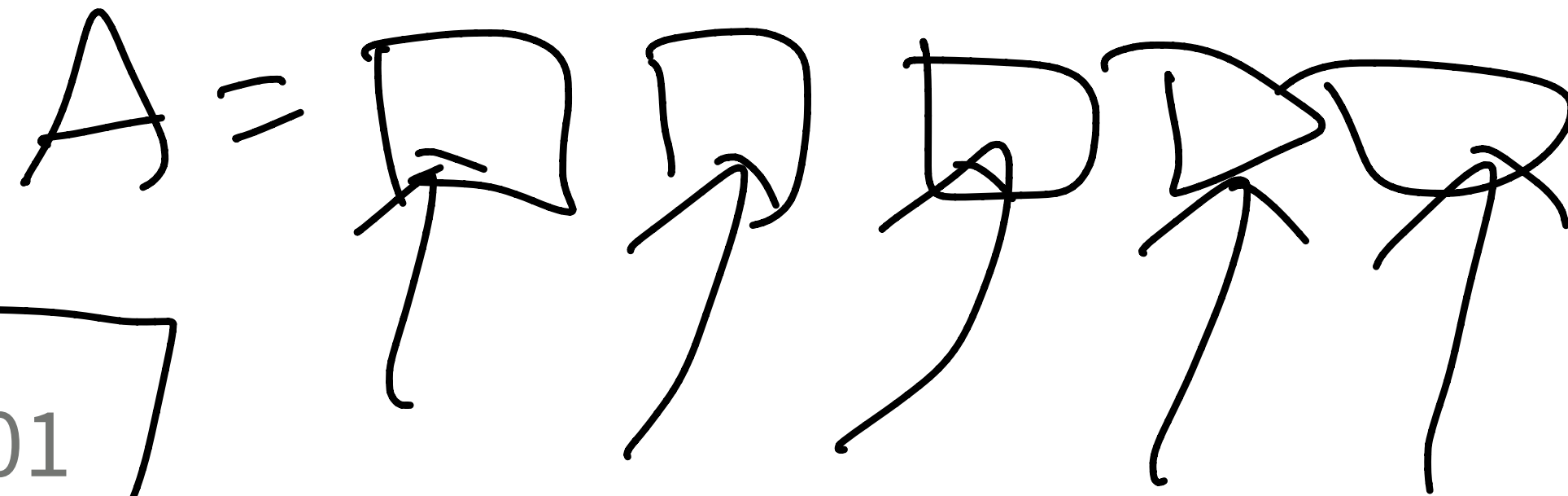
- 21개의 수를 10개의 자리에 넣어야 한다.

• 총 경우의 수: $21^{10} = 16,679,880,978,201$

- 경우의 수가 너무 많다.

- 일단 함수를 작성해보자.

$N \leq 10$

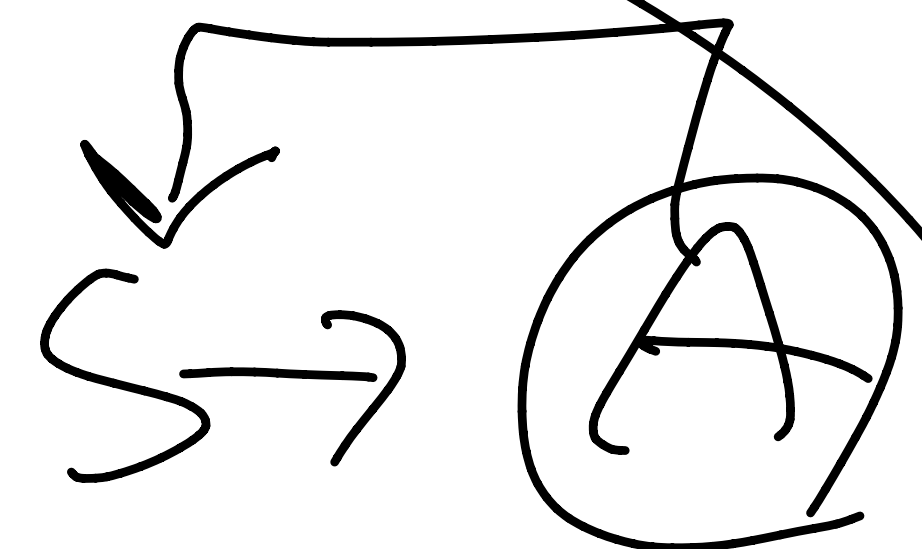


21^{10}

맞춰봐

<https://www.acmicpc.net/problem/1248>

```
bool go(int index) {  
    if (index == n) {  
        return ok();  
    }  
    for (int i = -10; i <= 10; i++) {  
        ans[index] = i;  
        if (go(index+1)) return true;  
    }  
    return false;  
}
```



$\overline{S} \quad \overline{T}$

\dots

$\overline{n-1} \mid \overline{n}$

맞춰봐

45

<https://www.acmicpc.net/problem/1248>

- 소스: <http://codeplus.codes/d309c55d1c6c44ff99f153eccea218fd>

맞춰봐

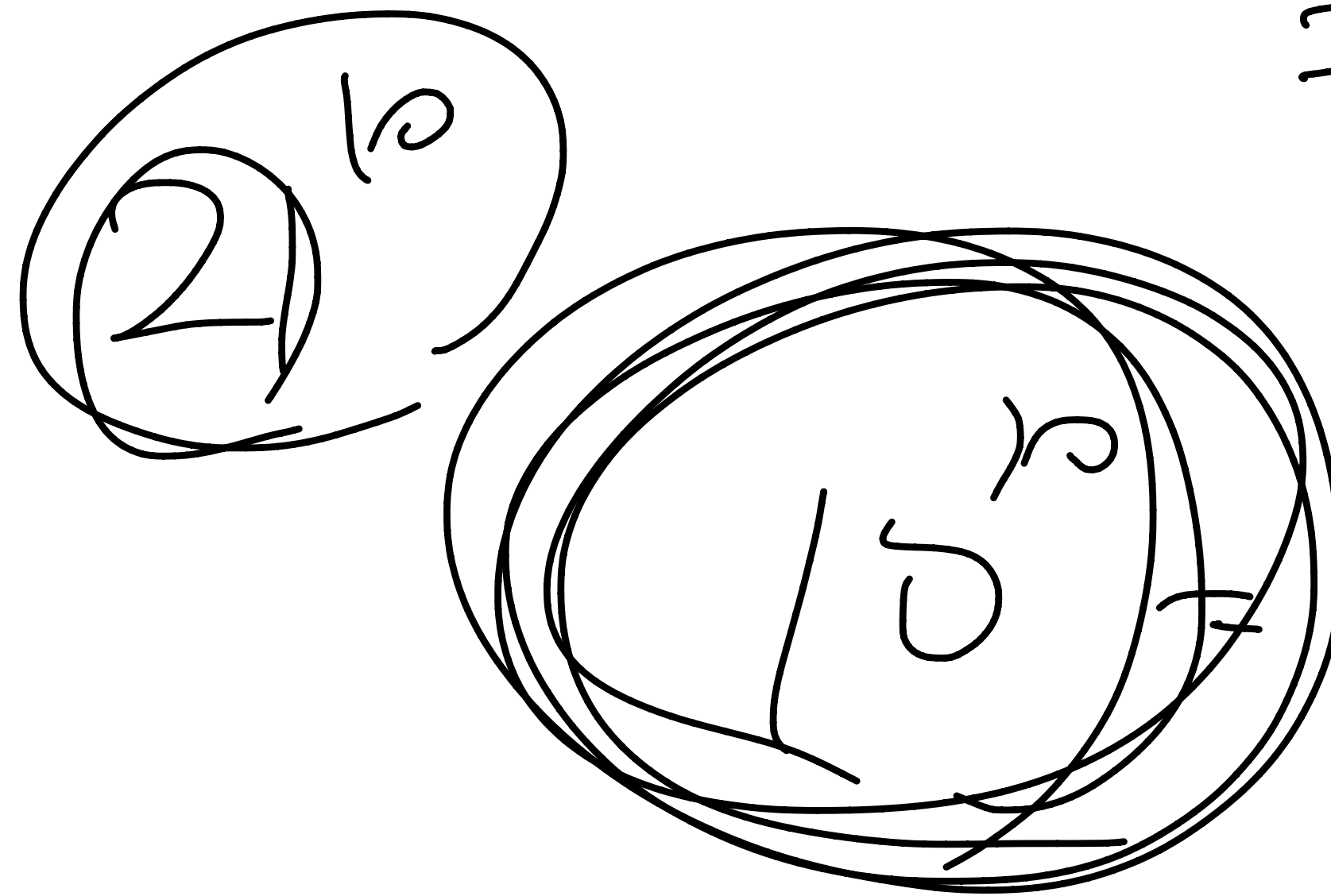
<https://www.acmicpc.net/problem/1248>

- sign[i][i]에는 i번째 수의 부호가 들어있다.
- -10까지 10까지 순회하지 않고
- 양수인 경우에는 1~10
- 음수인 경우에는 -10~-1
- 0인 경우에는 0
- 을 넣는 방식으로 개선해볼 수 있다.

STCCTCCT = 0

= + ATCTCT

= - ATCTCT

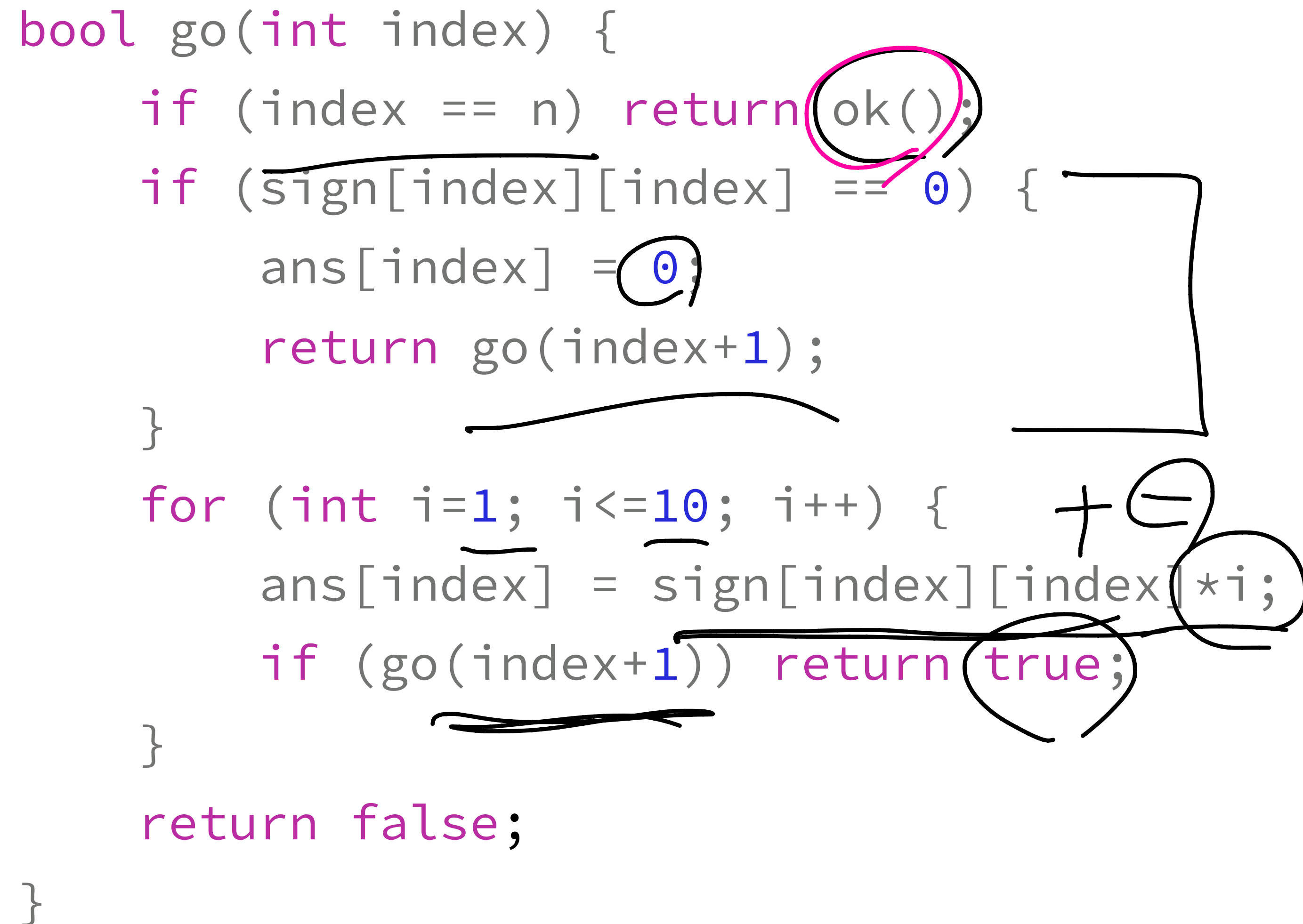


맞춰봐

47

<https://www.acmicpc.net/problem/1248>

```
bool go(int index) {  
    if (index == n) return ok();  
    if (sign[index][index] == 0) {  
        ans[index] = 0;  
        return go(index+1);  
    }  
    for (int i=1; i<=10; i++) {  
        ans[index] = sign[index][index]*i;  
        if (go(index+1)) return true;  
    }  
    return false;  
}
```



맞춰봐

48

<https://www.acmicpc.net/problem/1248>

- 소스: <http://codeplus.codes/aa9a3c07908d48da924049db290183cc>

맞춰봐

<https://www.acmicpc.net/problem/1248>

- index번째 수를 결정하면, 0~index번째 수는 변하지 않는다.
- 따라서, 모든 `sign[k][index]` ($0 \leq k < \text{index}$) 를 `go(index)`에서 검사할 수 있다

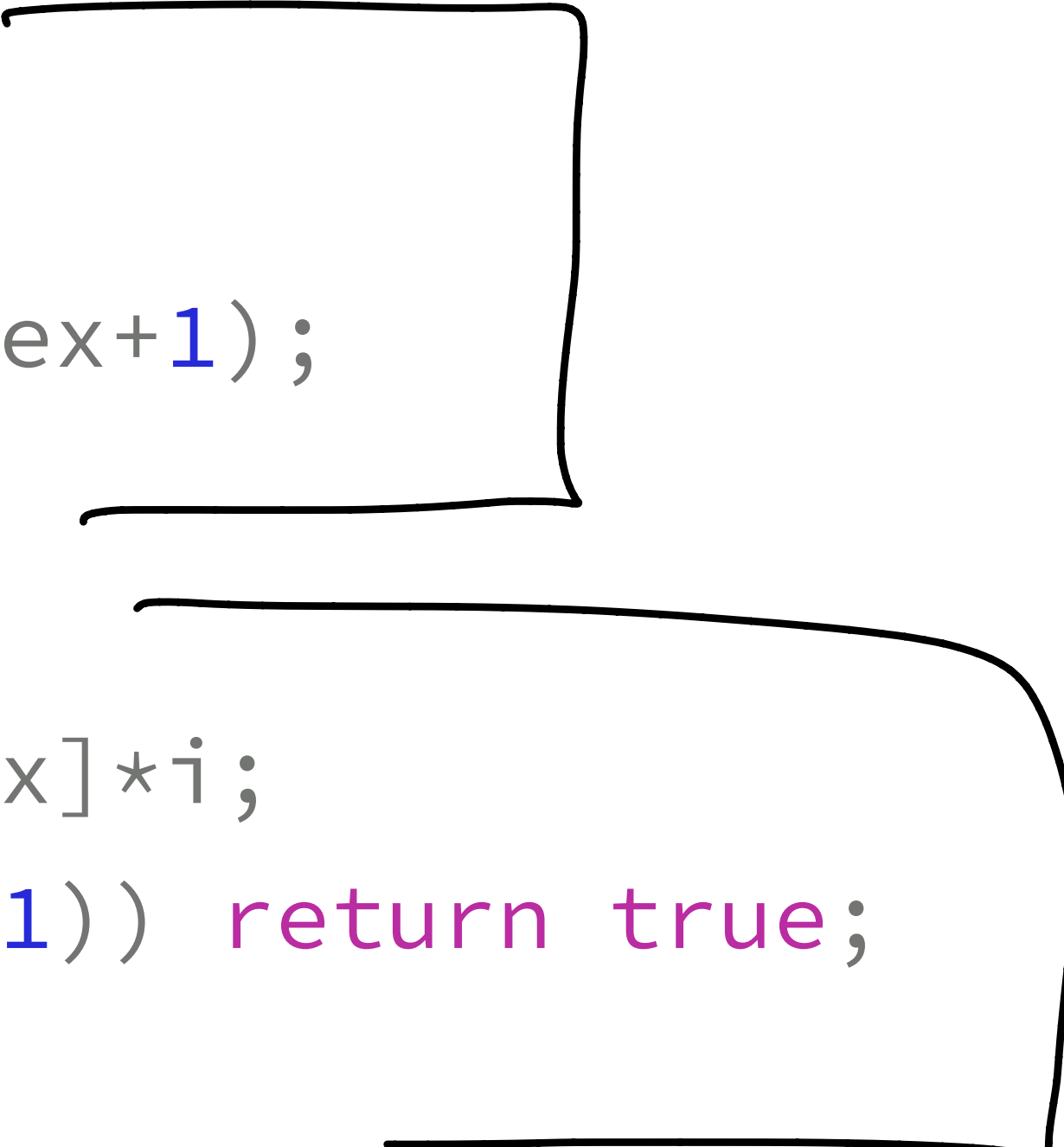
(0 ~ (index-1)) index 번째

맞춰봐

50

<https://www.acmicpc.net/problem/1248>

```
bool go(int index) {  
    if (index == n) return true;  
    if (sign[index][index] == 0) {  
        ans[index] = 0;  
        return check(index) && go(index+1);  
    }  
    for (int i=1; i<=10; i++) {  
        ans[index] = sign[index][index]*i;  
        if (check(index) && go(index+1)) return true;  
    }  
    return false;  
}
```



맞춰봐

51

<https://www.acmicpc.net/problem/1248>

0 ≤ K ≤ Index

```
bool check(int index) {
    int sum = 0;
    for (int i=index; i>=0; i--) {
        sum += ans[i];
        if (sign[i][index] == 0) {
            if (sum != 0) return false;
        } else if (sign[i][index] < 0) {
            if (sum >= 0) return false;
        } else if (sign[i][index] > 0) {
            if (sum <= 0) return false;
        }
    }
    return true;
}
```

맞춰봐

52

<https://www.acmicpc.net/problem/1248>

- 소스: <http://codeplus.codes/424e62fb723a4d79b780c7e580a731e1>

끝

코드 플러스

<https://code.plus>

- 슬라이드에 포함된 소스 코드를 보려면 "정보 수정 > 백준 온라인 저지 연동"을 통해 연동한 다음, "백준 온라인 저지"에 로그인해야 합니다.
- 강의 내용에 대한 질문은 코드 플러스의 "질문 게시판"에서 할 수 있습니다.
- 문제와 소스 코드는 슬라이드에 첨부된 링크를 통해서 볼 수 있으며, "백준 온라인 저지"에서 서비스됩니다.
- 슬라이드와 동영상 강의는 코드 플러스 사이트를 통해서만 볼 수 있으며, 동영상 강의의 녹화와 다운로드, 배포와 유통은 저작권법에 의해서 금지되어 있습니다.
- 다른 경로로 이 슬라이드나 동영상 강의를 본 경우에는 codeplus@startlink.io 로 이메일 보내주세요.
- 강의 내용, 동영상 강의, 슬라이드, 첨부되어 있는 소스 코드의 저작권은 스타트링크와 최백준에게 있습니다.