



고급 알고리즘

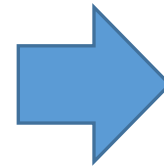
1-비밀지도

(출처) tech.kakao.com

비밀지도 - 2018 카카오 공채 코딩 테스트 문제

- 지도는 한 변의 길이가 n 인 정사각형 배열이며, 각 칸은 공백(" ") 또는 벽("#")의 두 종류로 이루어져 있음
- 전체 지도는 이런 두 장의 지도를 겹쳐서 얻을 수 있음
- 두 장의 지도를 겹쳐서 얻은 전체 지도에서 어느 하나의 지도라도 벽인 경우는 전체 지도에서 벽이고, 두 장 지도 모두 공백인 경우 전체 지도에서 공백임
- 각각의 지도 "지도 1"과 "지도 2"는 각각 정수 배열로 암호화되어 있다.
- 암호화된 배열은 지도의 각 가로줄에서 벽 부분을 1, 공백 부분을 0으로 부호화했을 때 얻어지는 이진수에 해당하는 값의 배열이다.

	#			#	01001(2) = 9
#		#			10100(2) = 20
#	#	#			11100(2) = 28
#			#		10010(2) = 18
	#		#	#	01011(2) = 18
#	#	#	#		11110(2) = 30
				#	00001(2) = 1
#		#		#	10101(2) = 21
#				#	10001(2) = 17
#	#	#			11100(2) = 28



#	#	#	#	#
#		#		#
#	#	#		#
#			#	#
#	#	#	#	#

1-비밀지도

(출처) tech.kakao.com

비밀지도 - 2018 카카오 공채 코딩 테스트 문제

입력	입력으로 지도의 한 변 크기 n 과 2개의 정수 배열 $arr1, arr2$ 가 들어온다. - $1 \leq n \leq 16$ - $arr1, arr2$ 는 길이 n 인 정수 배열로 주어진다. - 정수 배열의 각 원소 x 를 이진수로 변환했을 때의 길이는 n 이하이다. 즉, $0 \leq x \leq 2^n - 1$ 을 만족한다.	
출력	원래의 비밀지도를 해독하여 "#", 공백으로 구성된 문자열 배열로 출력하라.	
입출력 예	<u>입력</u> 5 [9, 20, 28, 18, 11] [30, 1, 21, 17, 28]	<u>출력</u> ["#####", "# # #", "### #", "# ##", "#####"]
	<u>입력</u> 6 [46, 33, 33, 22, 31, 50] [27, 56, 19, 14, 14, 10]	<u>출력</u> ["#####", "### #", "## ##", "#### ", "#####", "### #"]

1-비밀지도

배경지식

비트 연산 (AND, OR, XOR)

1 (배경지식) 비트 연산 (AND, OR, XOR)

		AND	OR	XOR
a	b	a & b	a b	a ^ b
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

```
#include <stdio.h>
int main()
{
    unsigned char num1 = 1; // 0000 0001
    unsigned char num2 = 3; // 0000 0011

    printf("%d\n", num1 & num2); // 0000 0001
    printf("%d\n", num1 | num2); // 0000 0011
    printf("%d\n", num1 ^ num2); // 0000 0010
    return 0;
}
```

1
3
2

1-비밀지도

배경지식

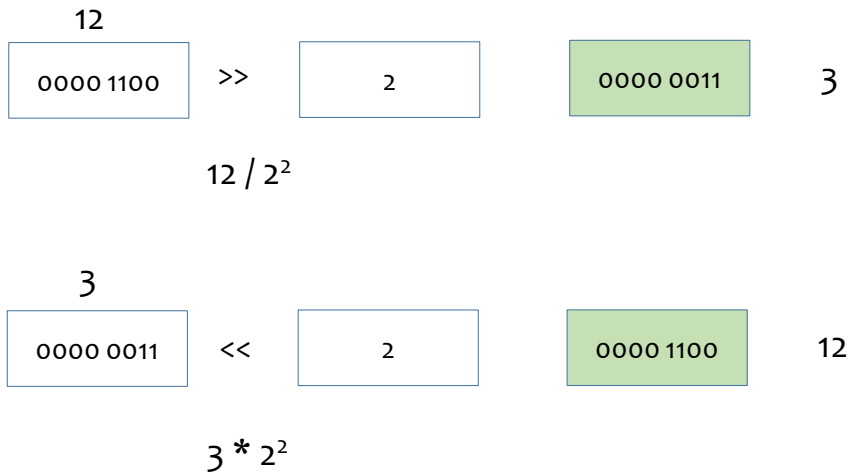
비트 연산 (shift)

2

(배경지식) 비트 연산 (shift)

$a \gg b$ a 값을 b 비트 수 만큼 오른쪽으로 시프트

$a \ll b$ a 값을 b 비트 수 만큼 왼쪽으로 시프트



```
#include <stdio.h>
int main()
{
    unsigned char num1 = 12; // 0000 1100
    unsigned char num2 = 3;  // 0000 0011

    printf("%u\n", num1 >> 2); // 3: 0000 0011
    printf("%u\n", num2 << 2); // 12: 0000 1100
    return 0;
}
```

3
12

1-비밀지도

배경지식

비트 연산자의 활용

3 (배경지식) 비트 연산자의 활용

flag = 1 << 3

0000 0001	<<	3	0000 1000
-----------	----	---	-----------

오른쪽에서 4번째 bit를 1로 setting

a		flag	
0000 1100	&	0000 1000	0000 1000

a의 오른쪽에서 4번째 bit가 1 값인지 여부를 검사

```
#include <stdio.h>
int main()
{
    unsigned char flag;
    unsigned char a = 12;

    flag = 1 << 3;
    if ( a & flag) printf("0000 1000\n"); // 0000 1000
    else printf("0000 0000\n");

    return 0;
}
```

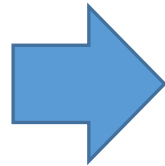
0000 1000

1-비밀지도

(출처) tech.kakao.com

비밀지도 - 2018 카카오 공채 코딩 테스트 문제

	#			#	01001(2) = 9
#		#			10100(2) = 20
#	#	#			11100(2) = 28
#			#		10010(2) = 18
	#		#	#	01011(2) = 18
#	#	#	#		11110(2) = 30
				#	00001(2) = 1
#		#		#	10101(2) = 21
#				#	10001(2) = 17
#	#	#			11100(2) = 28



#	#	#	#	#
#		#		#
#	#	#		#
#			#	#
#	#	#	#	#

입출력 예	입력	출력
	5 [9, 20, 28, 18, 11] [30, 1, 21, 17, 28]	["#####", "# # #", "### #", "# ##", "#####"]
	입력	출력
	6 [46, 33, 33, 22, 31, 50] [27, 56, 19, 14, 14, 10]	["#####", "### #", "## ##", "#### ", "#####", "### #"]

1-비밀지도

알고리즘

4 알고리즘(입력) – IDE <https://ide.geeksforgeeks.org/>

```
#include <stdio.h>
#include <string.h>
int main()
{
    int arr1[] = { 9, 20, 28, 18, 11 };
    int arr2[] = { 30, 1, 21, 17, 28 };
    int n = sizeof(arr1) / sizeof(arr1[0]);

    int i, j;
    int row;
```

```
    for (i = 0; i < n; i++) {
        row = arr1[i] | arr2[i];
```

```
        // row 값을 하위 5bit만 출력
```

```
    }

    return 0;
}
```


1-비밀지도

알고리즘

4 알고리즘(입력) – IDE <https://ide.geeksforgeeks.org/>

```
#include <stdio.h>
#include <string.h>

int main()
{
    int arr1[] = { 9, 20, 28, 18, 11 };
    int arr2[] = { 30, 1, 21, 17, 28 };
    int n = sizeof(arr1) / sizeof(arr1[0]);

    int i, j;
    int row;
```

```
    for (i = 0; i < n; i++) {
        row = arr1[i] | arr2[i];
```

// row 값을 하위 5bit만 출력

```
    }

    return 0;
}
```

```
printf("[");
for (j = n-1; j >= 0; j--) {
    if ( ((row >> j) & 1) == 1)
        printf("#");
    else
        printf(" ");
}
printf("]\n");
```

1-비밀지도

알고리즘

4 알고리즘(입력) – IDE <https://ide.geeksforgeeks.org/>

```
for (i = 0; i < n; i++) {  
    row = arr1[i] | arr2[i];
```

// row 값을 하위 5bit만 출력

```
}
```

```
return 0;
```

```
}
```

```
printf("[");  
for (j = n-1; j >= 0; j--) {  
    if ( ((row >> j) & 1) == 1)  
        printf("#");  
    else  
        printf(" ");  
}  
printf("]\n");
```

row

00010101

#

row

00010101



row >> 4

00000001



(row >> 4) & 1

00000001

&

00000001

00000001



#

row

00010101



row >> 3

00000010



(row >> 3) & 1

00000010

00000001

00000000



' '

1-비밀지도 (C 예시)

알고리즘

4 알고리즘(입력) – IDE <https://ide.geeksforgeeks.org/>

	#			#	01001(2) = 9
#		#			10100(2) = 20
#	#	#			11100(2) = 28
#			#		10010(2) = 18
	#		#	#	01011(2) = 18
#	#	#	#		11110(2) = 30
				#	00001(2) = 1
#		#		#	10101(2) = 21
#				#	10001(2) = 17
#	#	#			11100(2) = 28



#	#	#	#	#
#		#		#
#	#	#		#
#			#	#
#	#	#	#	#

```
C C
1 #include <stdio.h>
2 #include <string.h>
3
4 int main()
5 {
6     int arr1[] = { 9, 20, 28, 18, 11 };
7     int arr2[] = { 30, 1, 21, 17, 28 };
8     int n = sizeof(arr1)/sizeof(arr1[0]);
9
10    int i, j;
11    int row;
12
13    for (i = 0; i < n; i++) {
14        row = arr1[i] | arr2[i];
15
16        printf("[");
17        for (j = n-1; j >= 0; j--) {
18            if ( ((row >> j) & 1) == 1)
19                printf("#");
20            else
21                printf(" ");
22        }
23        printf("]\n");
24    }
25
26    return 0;
27 }
```

Input Goes Here..

Time(sec) : 0

Output:

```
[#####]
[# # #]
[###]
[# ##]
[#####]
```

1-비밀지도 (Java 예시)

알고리즘

4 알고리즘(입력) – IDE <https://ide.geeksforgeeks.org/>

2-문자열 압축

Problem-2

문자열 압축 - 2020 카카오 공채 코딩 테스트 문제

입력	<p>압축할 문자열 s가 매개변수로 주어질 때, 1개 이상 문자열을 잘라 압축하여 표현한 문자열 중 가장 짧은 것의 길이를 구하시오.</p> <p>단) $1 \leq s$의 길이 ≤ 1000</p> <p>s는 알파벳 소문자로만 이루어져 있습니다.</p> <p>문자열은 제일 앞부터 정해진 길이만큼 잘라야 합니다.</p>		
출력	1개 이상 단위로 문자열을 잘라 압축하여 표현한 문자열 중 가장 짧은 것의 길이를 출력하시오.		
입출력 예	입력 aabbaccc	문자열 1개 단위 분할 2a2ba3c	출력 7
	abababcdcdabababcdcd	문자열 8개 단위 분할 2abababcdcd 2ab2cd2ab2cd	9
	abcabcdede	문자열 3개 단위 분할 2abcdede Abcabcd2de	8
	abcabcabcababcdedede	2개 - abcabcabcabc6de 3개 - 4abcdedede 4개 - abcabcabcabc3dede 6개 - 2abcabc2dedede	14

2-문자열 압축

Problem-2

입출력
예

입력

abbcccdddd

출력

1a2b3c4d

1 (배경지식) RLE (Run Length Encoding)

압축기법 : 문자열에 특정 패턴이 반복 될 경우, 이를 이용하여 문자열을 좀 더 짧게 나타내는 기법

RLE(Run Length Encoding) : 가장 기초적인 압축방식으로 문자와 반복횟수를 저장하는 방식

(예) “abbcccdddd”인 문자열의 경우, “1a2b3c4d”로 저장

문자	a	b	b	c	c	c	d	d	d	d	
출현수	1	1	2	1	2	3	1	2	3	4	
출력		1a		2b			3c				4d

2-문자열 압축

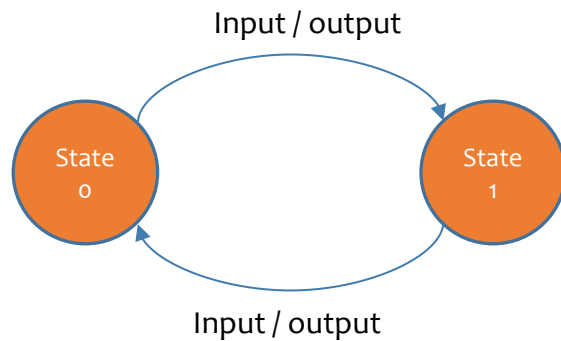
Problem-2

입출력
예

입력
abbcccdddd

출력
1a2b3c4d

1 (배경지식) State Transition Diagram



- 상태와 상태전이를 표현하기 위해 도식화 해서 표현하는 그림
- 프로그램에서도 프로그램 실행이 진행되는 과정상, 상태를 통해 개념화 한 다음 추상화 해서 그림을 그리고 코딩을 하면 오류를 줄이는 문서로써 유효
- 특정 상태에서 입력 값을 받으면 특정 출력을 실행하면서 다른 상태로 전이
- 문제를 해결하기 위한 시스템의 흐름을 상태와 상태전이로 표현

2-문자열 압축

Problem - 2

입출력
예

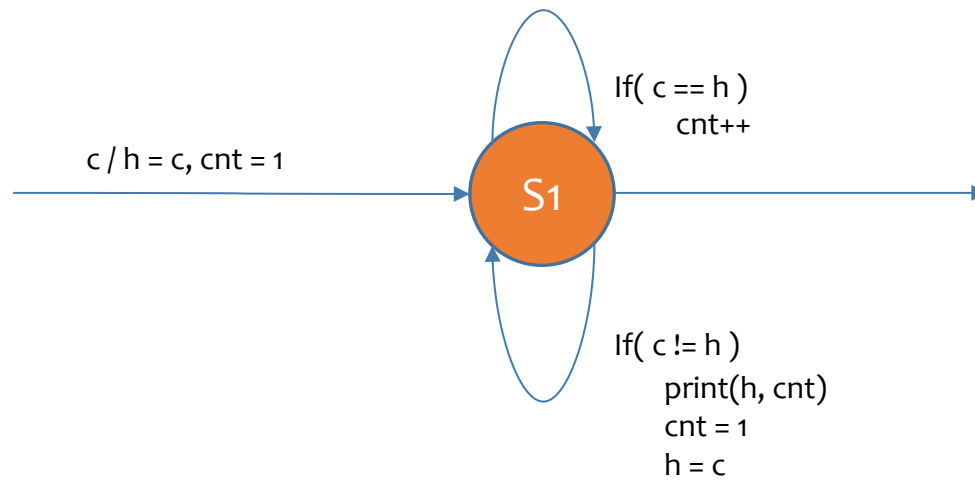
입력

abbcccdddd

출력

1a2b3c4d

4 RLE (Run Length Encoding) state transition diagram 2



문자열1개

문자	a	a	a	a	b	b	a	b	b	a	b	b	
문자열길이	1	2	3	4	1	2	1	1	2	1	1	2	
출력					4a		2b	a		2b	a		2b

문자열2개

문자	a	a	a	a	b	b	a	b	b	a	b	b	
문자열길이		1		2		1		1		1		1	
출력					2aa		bb		ab		ba		bb

문자열 3개

[illegible]

2-문자열 압축

문자열4개

문자	a	a	a	a	b	b	a	b	b	a	b	b	
문자열길이				1				1					
출력					aaaa				bbab				babb

문자열5개

문자	a	a	a	a	b	b	a	b	b	a	b	b	
문자열길이					1					1		1	
출력						aaaab					babba		bb

문자열6개

문자	a	a	a	a	b	b	a	b	b	a	b	b	
문자열길이						1						1	
출력							aaaabb						abbabb

2-문자열 압축 (C++ 예시)

```
소스.cpp* x 13-2.c 13-1.c
Project13
1 #include <string>
2 #include <vector>
3 using namespace std;
4
5 int solution(string s);
6 int main(void) {
7     string str = { "abcabcdede" };
8     int answer = solution(str);
9     printf("%d", answer);
10 }
11
12 int solution(string s) {
13     //int answer = 0;
14     int answer = s.length();
15     for (int i = 1; i <= s.length() / 2; i++) {
16         int len = s.length();
17         for (int j = 0; j < s.length(); j++) {
18             for (int count = 0, z = i; j + z < s.length(); z += i) {
```

```
19                 if (s.substr(j, i) == s.substr(j + z, i)) count++;
20             }
21             len -= i * count;
22             if (count) len += to_string(count + 1).length();
23             j += z - 1;
24             break;
25         }
26         if (j + z + i >= s.length()) {
27             len -= i * count;
28             len += to_string(count + 1).length();
29             j += z;
30         }
31     }
32 }
33 if (len < answer) answer = len;
34 }
35 return answer;
36 }
```

2-문자열 압축 (Java 예시)



2-문자열 압축 (Java 예시)

