

DFS/BFS

최백준 choi@startlink.io

외판원 순회 2

2

<https://www.acmicpc.net/problem/10971>

- 영어로 Travelling Salesman Problem (TSP)
- 1번부터 N번까지 번호가 매겨져있는 도시가 있다
- 한 도시에서 시작해 N개의 모든 도시를 거쳐 다시 원래 도시로 돌아오려고 한다 (한 번 갔던 도시로는 다시 갈 수 없다)
- 이 때, 가장 적은 비용을 구하는 문제
- $W[i][j] = i \rightarrow j$ 비용

외판원 순회 2

<https://www.acmicpc.net/problem/10971>

- $2 \leq N \leq 10$
- $N! = 10! = 3628800$
- 모든 경우를 다해봐도 시간 안에 나온다

외판원 순회 2

<https://www.acmicpc.net/problem/10971>

- $2 \leq N \leq 10$
- $N! = 10! = 3628800$
- 모든 경우를 다해봐도 시간 안에 나온다
- 모든 경우 = $N!$
 - 비용 계산 = N
- 시간복잡도: $O(N * N!)$

외판원 순회 2

<https://www.acmicpc.net/problem/10971>

```
do {  
    bool ok = true;  
    int sum = 0;  
    for (int i=0; i<n-1; i++) {  
        if (w[d[i]][d[i+1]] == 0) ok = false;  
        else sum += w[d[i]][d[i+1]];  
    }  
    if (ok && w[d[n-1]][d[0]] != 0) {  
        sum += w[d[n-1]][d[0]];  
        if (ans > sum) ans = sum;  
    }  
} while (next_permutation(d.begin(), d.end()));
```

외판원 순회 2

<https://www.acmicpc.net/problem/10971>

- $O(N \cdot N!)$
- C++: <https://gist.github.com/Baekjoon/a62f0b1263752c8d1a75>
- Java: <https://gist.github.com/Baekjoon/a5450f44bc19da72f9ac>
- $O(N!)$
- C++: <https://gist.github.com/Baekjoon/3eeee9003b22cffb2a76>
- C++ 2: <https://gist.github.com/Baekjoon/45c47a211c3be61e054a>
- Java: <https://gist.github.com/Baekjoon/88bfb6c2e54bb399beb2>

숨바꼭질

7

<https://www.acmicpc.net/problem/1697>

- 수빈이의 위치: N
 - 동생의 위치: K
 - 동생을 찾는 가장 빠른 시간을 구하는 문제
-
- 수빈이가 할 수 있는 행동 (위치: X)
 1. 걷기: $X+1$ 또는 $X-1$ 로 이동 (1초)
 2. 순간이동: $2*X$ 로 이동 (1초)

숨바꼭질

<https://www.acmicpc.net/problem/1697>

- 수빈이의 위치: 5
- 동생의 위치: 17
- 5-10-9-18-17 로 4초만에 동생을 찾을 수 있다.

숨바꼭질

<https://www.acmicpc.net/problem/1697>

- 큐에 수빈이의 위치를 넣어가면서 이동시킨다
- 한 번 방문한 곳은 다시 방문하지 않는 것이 좋기 때문에, 따로 배열에 체크하면서 방문

숨바꼭질

10

<https://www.acmicpc.net/problem/1697>

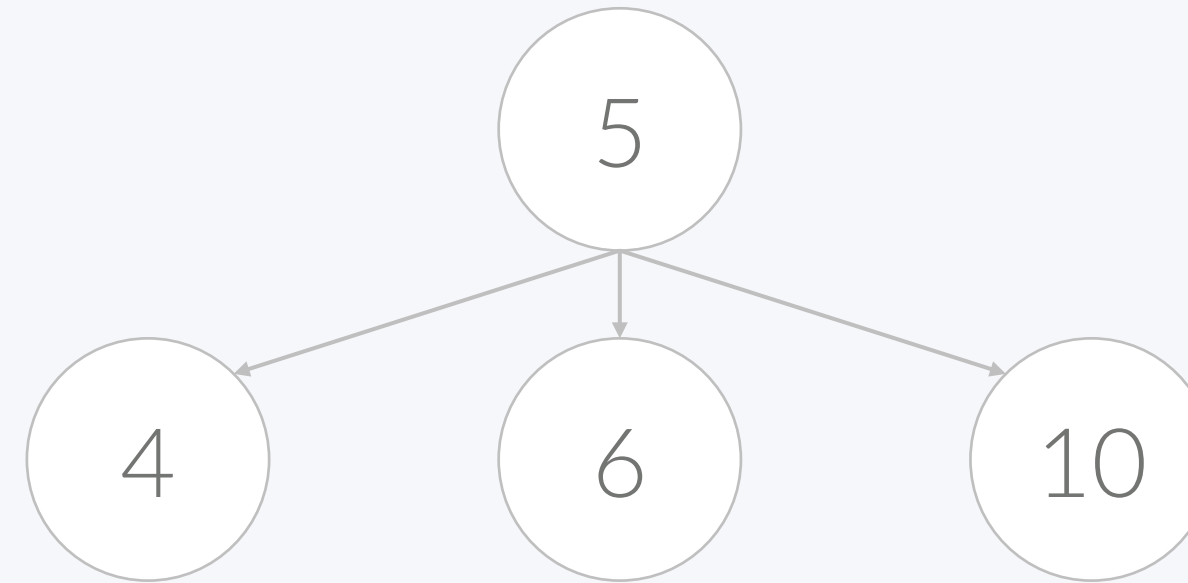
- 가장 처음
- Queue: 5

5

숨바꼭질

<https://www.acmicpc.net/problem/1697>

- 5에서 이동
- Queue: 5 4 6 10

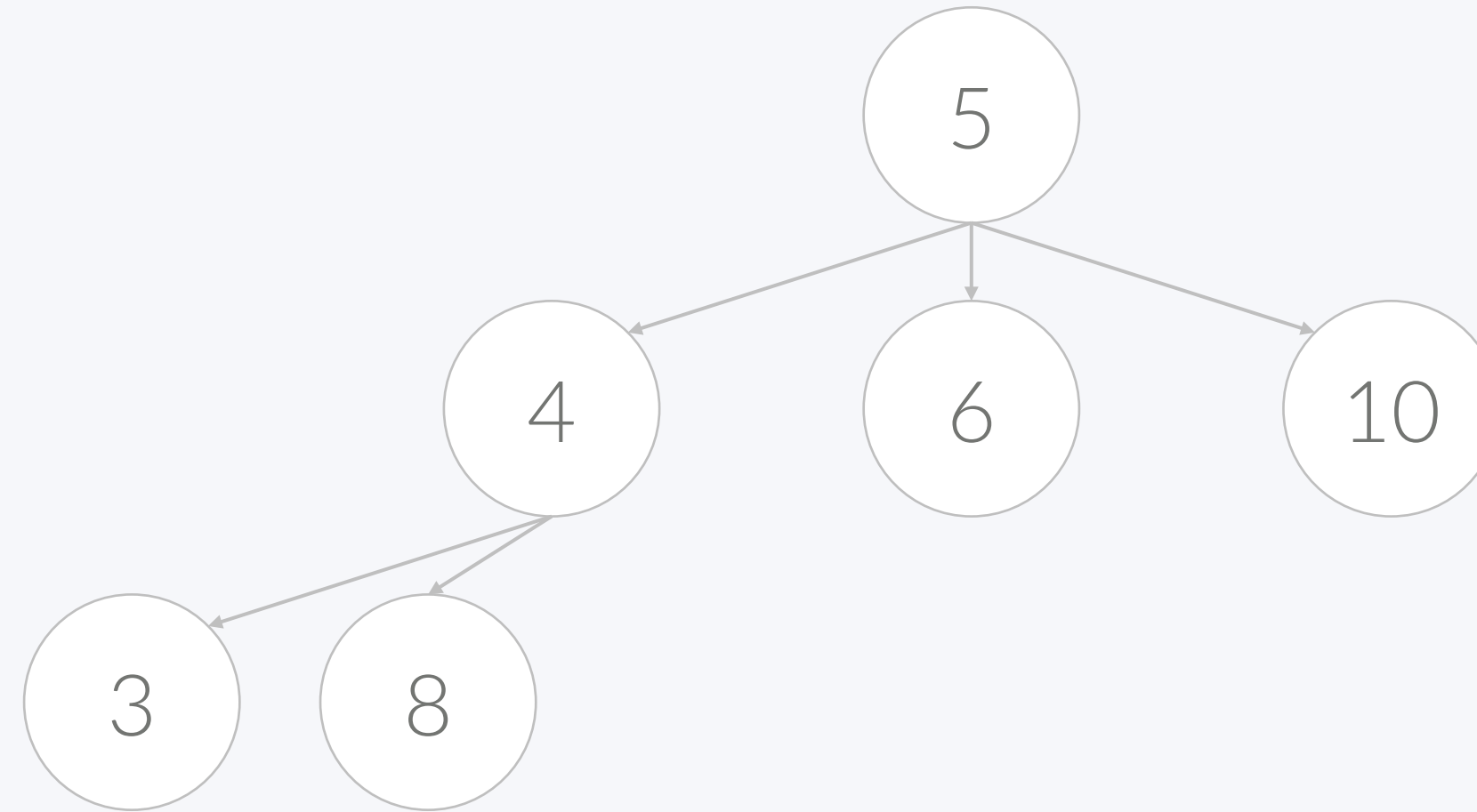


숨바꼭질

12

<https://www.acmicpc.net/problem/1697>

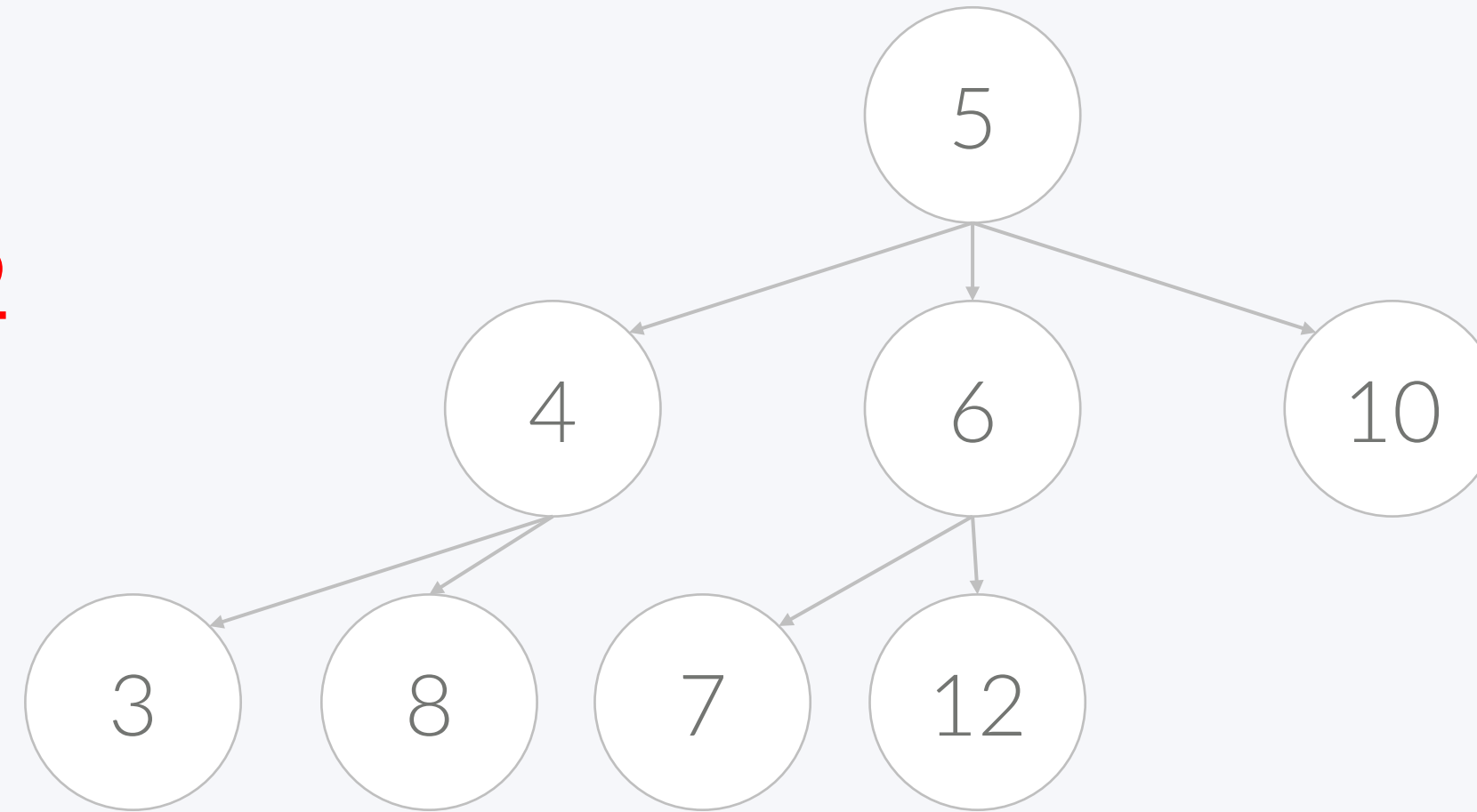
- 4에서 이동
- Queue: 5 4 6 10 3 8



숨바꼭질

<https://www.acmicpc.net/problem/1697>

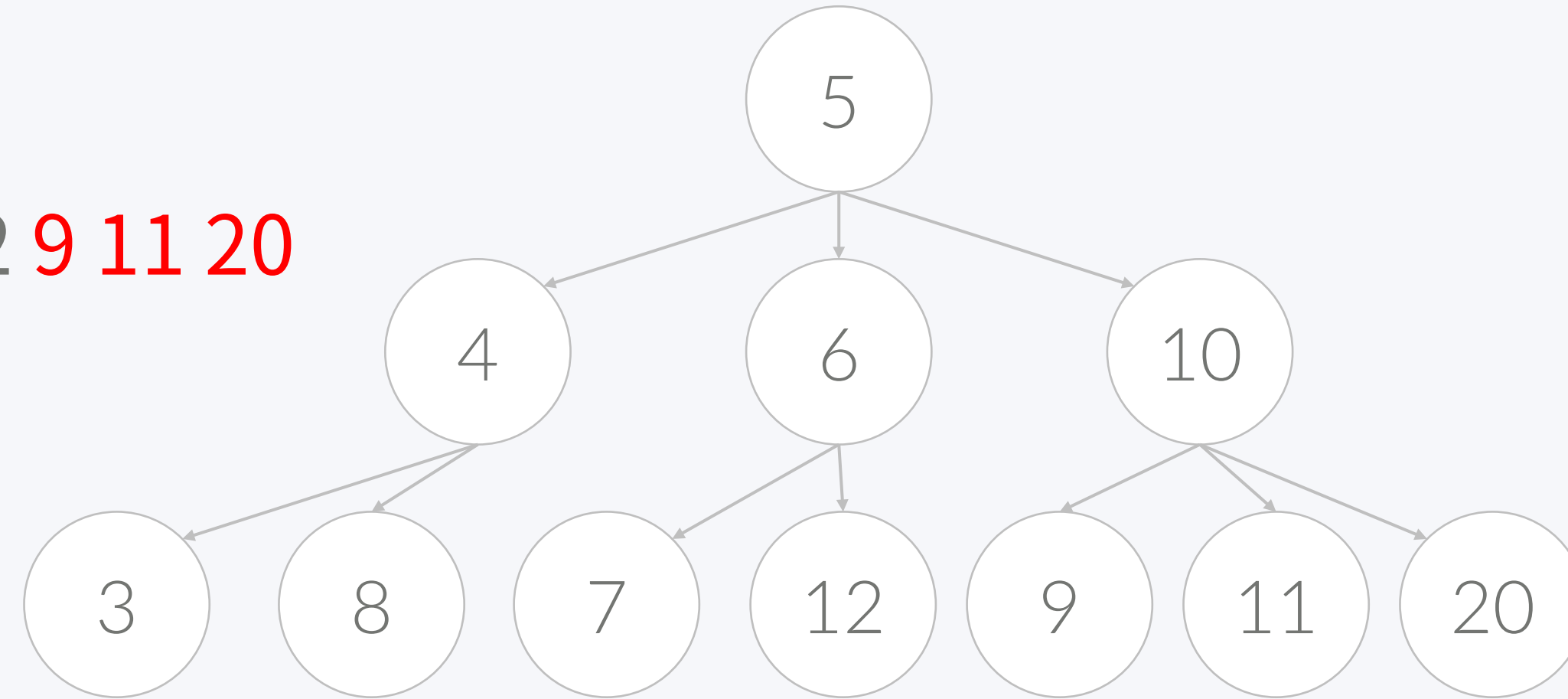
- 6에서 이동
- Queue: 5 4 6 10 3 8 7 12



숨바꼭질

<https://www.acmicpc.net/problem/1697>

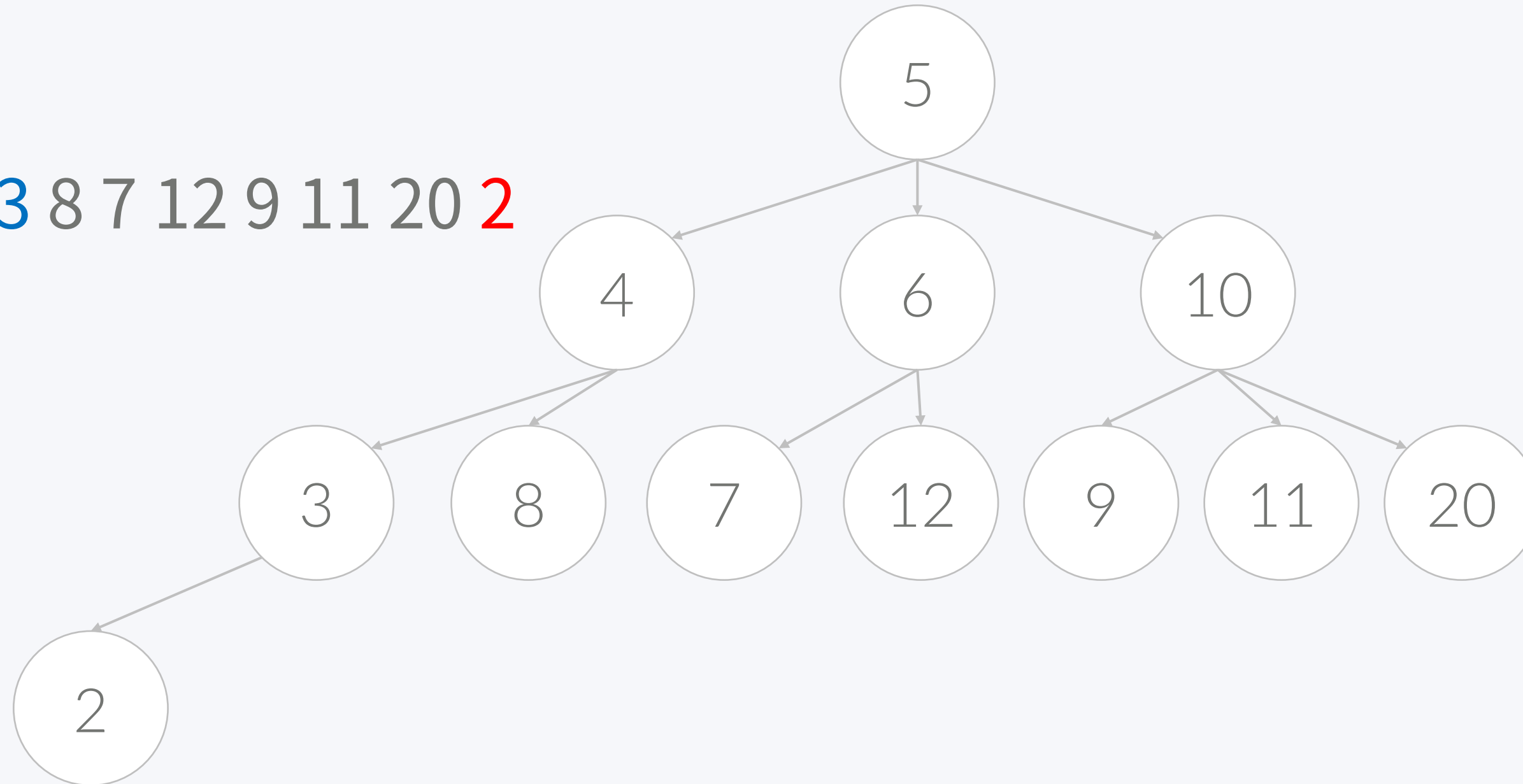
- 10에서 이동
- Queue: 5 4 6 10 3 8 7 12 9 11 20



숨바꼭질

<https://www.acmicpc.net/problem/1697>

- 3에서 이동
- Queue: 5 4 6 10 3 8 7 12 9 11 20 2



숨바꼭질

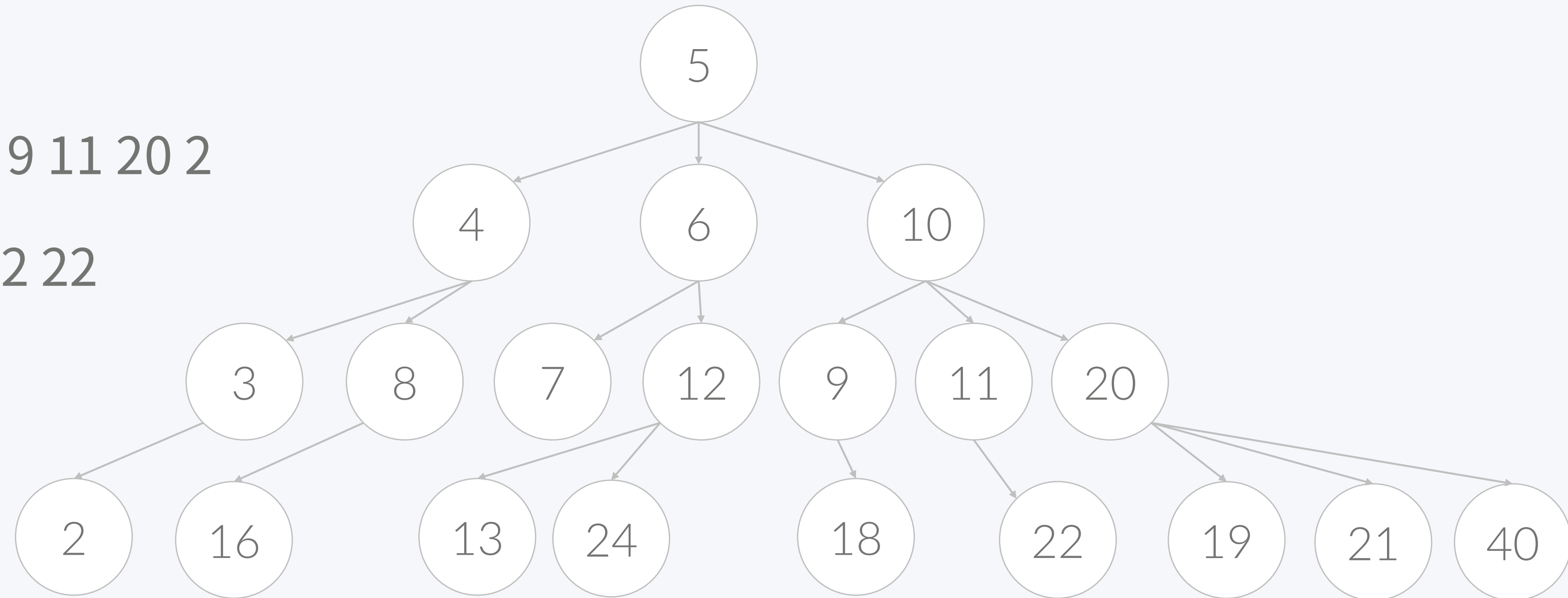
<https://www.acmicpc.net/problem/1697>

- 이런식으로...

숨바꼭질

<https://www.acmicpc.net/problem/1697>

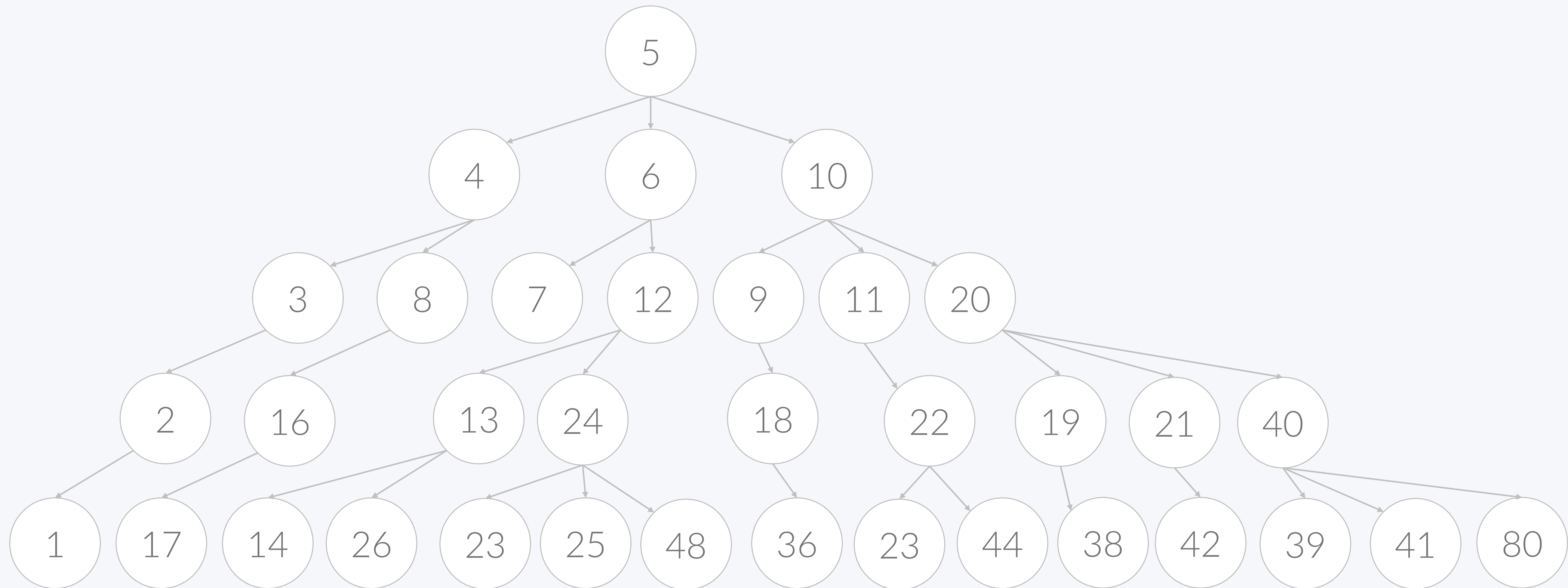
- Queue:
- 5 4 6 10 3 8 7 12 9 11 20 2
- 16 13 15 28 18 12 22
- 19 21 40



숨바꼭질

<https://www.acmicpc.net/problem/1697>

•



숨바꼭질

<https://www.acmicpc.net/problem/1697>

- $check[i] = i$ 를 방문했는지
- $dist[i] = i$ 를 몇 번만에 방문했는지

숨바꼭질

20

<https://www.acmicpc.net/problem/1697>

```
check[n] = true;
dist[n] = 0;
queue<int> q;
q.push(n);
while (!q.empty()) {
    int now = q.front();
    q.pop();
    if (now-1 >= 0) {
        if (check[now-1] == false) {
            q.push(now-1);
            check[now-1] = true;
            dist[now-1] = dist[now] + 1;
        }
    }
}
```

```
if (now+1 < MAX) {
    if (check[now+1] == false) {
        q.push(now+1);
        check[now+1] = true;
        dist[now+1] = dist[now] + 1;
    }
}
if (now*2 < MAX) {
    if (check[now*2] == false) {
        q.push(now*2);
        check[now*2] = true;
        dist[now*2] = dist[now] + 1;
    }
}
```

숨바꼭질

<https://www.acmicpc.net/problem/1697>

- C: <https://gist.github.com/Baekjoon/2505b37c55949c9487904f49b5346111>
- C++: <https://gist.github.com/Baekjoon/5b8924d3aec661746358>
- Java: <https://gist.github.com/Baekjoon/e1abcf6deb6796c1282a>

숨바꼭질 4

22

<https://www.acmicpc.net/problem/13913>

- 숨바꼭질 문제 + 이동하는 방법을 출력하는 문제

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

- now -> next를 갔다고 한다면

```
if (check[next] == false) {  
    q.push(next);  
    check[next] = true;  
    dist[next] = dist[now] + 1;  
}
```

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

- now -> next를 갔다고 한다면

```
if (check[next] == false) {  
    q.push(next);  
    check[next] = true;  
    from[next] = now;  
    dist[next] = dist[now] + 1;  
}
```


숨바꼭질 4

25

<https://www.acmicpc.net/problem/13913>

- $\text{from}[i]$ = 어디에서 왔는지
- 의미: $\text{from}[i] \rightarrow i$
- N에서 K를 가는 문제 이기 때문에
- K부터 from을 통해서 N까지 가야한다.
- 즉, 역순으로 저장되기 때문에, 다시 역순으로 구하는 것이 필요하다.

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

```
void print(int n, int m) {  
    if (n != m) {  
        print(n, from[m]);  
    }  
    cout << m << ' ';  
}
```

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

```
stack<int> ans;
for (int i=m; i!=n; i=from[i]) {
    ans.push(i);
}
ans.push(n);
while (!ans.empty()) {
    cout << ans.top() << ' ';
    ans.pop();
}
cout << '\n';
```

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

- C: <https://gist.github.com/Baekjoon/3d918ef75f8598f4e73abf0fb5e972ef>
- C++: <https://gist.github.com/Baekjoon/914c1f46da5d4e662758ff6c452dc1e8>
- C++: <https://gist.github.com/Baekjoon/d823eaf585990e75ef5760dd80f232a6>
- Java: <https://gist.github.com/Baekjoon/7dc73e85fe343fc9dbea8b32c1347d7a>
- Java: <https://gist.github.com/Baekjoon/fc8c2fa94562827e8289d59a3fa4b96a>

DSLR

<https://www.acmicpc.net/problem/9019>

- 네 자리 숫자 A와 B가 주어졌을 때
- $A \rightarrow B$ 로 바꾸는 최소 연산 횟수
- $D: N \rightarrow 2 \times N$
- $S: N \rightarrow N-1$
- L: 한 자리씩 왼쪽으로
- R: 한 자리씩 오른쪽으로

DSLR

<https://www.acmicpc.net/problem/9019>

- 앞의 두 문제와 동일하게 풀지만
- 이 문제는 최소값을 구해야 하는건 맞지만
- 어떠한 과정을 거쳐야 하는지를 구해야 한다
- 배열 두 개를 더 이용해서 어떤 과정을 거쳤는지를 저장해야 한다
- $from[i] = i$ 를 어떤 수에서 만들었는지
- $how[i] = i$ 를 어떻게 만들었는지

DSL

<https://www.acmicpc.net/problem/9019>

```
int next = (now*2) % 10000;  
if (check[next] == false) {  
    q.push(next);  
    check[next] = true;  
    dist[next] = dist[now]+1;  
    from[next] = now;  
    how[next] = 'D';  
}
```

DSLR

<https://www.acmicpc.net/problem/9019>

```
next = now-1;
if (next == -1) next = 9999;
if (check[next] == false) {
    q.push(next);
    check[next] = true;
    dist[next] = dist[now]+1;
    from[next] = now;
    how[next] = 'S';
}
```


DSL

<https://www.acmicpc.net/problem/9019>

```
next = (now%1000)*10 + now/1000;
if (check[next] == false) {
    q.push(next);
    check[next] = true;
    dist[next] = dist[now]+1;
    from[next] = now;
    how[next] = 'L';
}
```

DSL R

<https://www.acmicpc.net/problem/9019>

```
next = (now/10) + (now%10)*1000;  
if (check[next] == false) {  
    q.push(next);  
    check[next] = true;  
    dist[next] = dist[now]+1;  
    from[next] = now;  
    how[next] = 'R';  
}
```

DSLR

<https://www.acmicpc.net/problem/9019>

```
string ans = "";  
while (B != A) {  
    ans += how[B];  
    B = from[B];  
}  
reverse(ans.begin(), ans.end());  
cout << ans << '\\n';
```

DSL R

<https://www.acmicpc.net/problem/9019>

```
void print(int A, int B) {  
    if (A == B) return;  
    print(A, from[B]);  
    cout << how[B];  
}
```

DSL R

<https://www.acmicpc.net/problem/9019>

- C++: <https://gist.github.com/Baekjoon/05b6e7a9c0bf6d4742ab>
- C++: <https://gist.github.com/Baekjoon/82ed59e4713f5286001ddcbba644849b>
- Java: <https://gist.github.com/Baekjoon/8c41844868ecb0c58a44>

이모티콘

<https://www.acmicpc.net/problem/14226>

- 화면에 이모티콘은 1개다
- 할 수 있는 연산
 - 이모티콘을 모두 복사해서 클립보드에 저장
 - 클립보드에 있는 모든 이모티콘을 화면에 붙여넣기
 - 화면에 있는 이모티콘 중 하나를 삭제
- S개의 이모티콘을 치는데 걸리는 시간의 최소값을 구하는 문제

이모티콘

<https://www.acmicpc.net/problem/14226>

- 화면에 이모티콘의 개수 s 와 클립보드에 있는 이모티콘의 개수 c 가 중요하다
- 복사: $(s, c) \rightarrow (s, s)$
- 붙여넣기: $(s, c) \rightarrow (s+c, c)$
- 삭제: $(s, c) \rightarrow (s-1, c)$
- $2 \leq S \leq 1,000$ 이기 때문에 BFS 탐색으로 가능하다.

이모티콘

<https://www.acmicpc.net/problem/14226>

- C++: <https://gist.github.com/Baekjoon/195fcf8798fe3b5b9ae8e46765734265>
- Java: <https://gist.github.com/Baekjoon/5fb20656b011a9842b8e68a44328a398>

1, 2, 3 더하기

41

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

- 정수 n 을 1, 2, 3의 조합으로 나타내는 방법의 수를 구하는 문제
- $n = 4$
- $1+1+1+1$
- $1+1+2$
- $1+2+1$
- $2+1+1$
- $2+2$
- $1+3$
- $3+1$

1, 2, 3 더하기

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

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

1, 2, 3 더하기

43

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

- `go(count, sum, goal)`
- 숫자 `count`개로 합 `sum`을 만드는 경우의 수
- 불가능한 경우
 - `count > 10`
 - `sum > goal`
- 가능한 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

45

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

```
int go(int count, int sum, int goal) {  
    if (count > 10) return 0;  
    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;  
}
```

1, 2, 3 더하기

46

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

- C++: <https://gist.github.com/Baekjoon/3235f76fe44c1ad17648>
- Java: <https://gist.github.com/Baekjoon/bdeba307e9e6d1e80fc7>

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

암호 만들기

48

<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
 - aisw
 - aitw
 - astw
 - cist
 - cisw
 - citw
 - istw

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

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
- 언제 답인지 아닌지 확인해야 하나?
 - `n == password.length()`
- 다음
 - `i`번째 알파벳을 사용하는 경우
 - `i`번째 알파벳을 사용하지 않는 경우

암호 만들기

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

- 다음
 - 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>

```
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);
}
```

암호 만들기

53

<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>

- C++: <https://gist.github.com/Baekjoon/dff42ddf0ae028f6b7f1>
- Java: <https://gist.github.com/Baekjoon/e92cfec2c020cd62b8ef>

스도쿠

<https://www.acmicpc.net/problem/2580>

- 스도쿠를 푸는 문제

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

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

스도쿠

<https://www.acmicpc.net/problem/2580>

- $go(z)$: z 번째 칸을 채우는 함수
- $(x, y) \rightarrow 9 * x + y$ 번째 칸

0	1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16	17
18	19	20	21	22	23	24	25	26
27	28	29	30	31	32	33	34	35
36	37	38	39	40	41	42	43	44
45	46	47	48	49	50	51	52	53
54	55	56	57	58	59	60	61	62
63	64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79	80

- `c2[i][j]` = i열에 숫자 j가 있으면 true

[illegible]

스도쿠

<https://www.acmicpc.net/problem/2580>

- $c3[i][j] = i$ 번 작은 정사각형에 숫자 j 가 있으면 true
- (x, y) 는 $(x/3)*3+(y/3)$ 번째 칸

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

스도쿠

<https://www.acmicpc.net/problem/2580>

```
for (int i=0; i<n; i++) {  
    for (int j=0; j<n; j++) {  
        cin >> a[i][j];  
        if (a[i][j] != 0) {  
            c[i][a[i][j]] = true;  
            c2[j][a[i][j]] = true;  
            c3[square(i,j)][a[i][j]] = true;  
        }  
    }  
}  
  
go(0);
```

스도쿠

61

<https://www.acmicpc.net/problem/2580>

```
void go(int z) {
    if (z == 81) {
        // check
        exit(0);
    }
    int x = z/n, y = z%n;
    if (a[x][y] != 0) {
        go(z+1);
    } else {
        // next
    }
}
```

스도쿠

62

<https://www.acmicpc.net/problem/2580>

```
// check
for (int i=0; i<n; i++) {
    for (int j=0; j<n; j++) {
        cout << a[i][j] << ' ';
    }
    cout << '\n';
}
exit(0);
```

<https://www.acmicpc.net/problem/2580>

```
// next
```

```
for (int i=1; i<=9; i++) {  
    if (c[x][i] == 0 && c2[y][i] == 0 && c3[square(x,y)][i]==0) {  
        c[x][i] = c2[y][i] = c3[square(x,y)][i] = true;  
        a[x][y] = i;  
        go(z+1);  
        a[x][y] = 0;  
        c[x][i] = c2[y][i] = c3[square(x,y)][i] = false;  
    }  
}
```

스도쿠

<https://www.acmicpc.net/problem/2580>

- C++: <https://gist.github.com/Baekjoon/fbf225181c5946773106>
- C++: <https://gist.github.com/Baekjoon/23b7f9e8cd454d94d2a7>
- Java: <https://gist.github.com/Baekjoon/687e776019684b5aca54>

부분집합의 합

65

<https://www.acmicpc.net/problem/1182>

- 서로 다른 N개의 정수로 이루어진 집합이 있을 때, 이 집합의 공집합이 아닌 부분집합 중에서 그 집합의 원소를 다 더한 값이 S가 되는 경우의 수를 구하는 문제
- $1 \leq N \leq 20$

부분집합의 합

<https://www.acmicpc.net/problem/1182>

- 모든 집합의 개수 = 2^N
- 모든 집합을 구해보면 된다!

부분집합의 합

67

<https://www.acmicpc.net/problem/1182>

- 전체 집합 = $(1 \ll N) - 1$

```
for (int i=0; i<(1<<n); i++) {  
  
}
```

부분집합의 합

<https://www.acmicpc.net/problem/1182>

- 전체 집합 = $(1 \ll N) - 1$
- 공집합은 제외해야 한다

```
for (int i=1; i<(1<<n); i++) {  
  
}
```

부분집합의 합

<https://www.acmicpc.net/problem/1182>

- 전체 집합 = $(1 \ll N) - 1$
- 공집합은 제외해야 한다
- 집합에 무엇이 포함되어 있는지 확인하기

```
for (int i=1; i<(1<<n); i++) {  
    for (int k=0; k<n; k++) {  
        if (i&(1<<k)) {  
            }  
        }  
    }  
}
```

부분집합의 합

<https://www.acmicpc.net/problem/1182>

```
for (int i=1; i<(1<<n); i++) {  
    int sum = 0;  
    for (int k=0; k<n; k++) {  
        if (i&(1<<k)) {  
            sum += a[k];  
        }  
    }  
    if (sum == s) {  
        ans += 1;  
    }  
}
```

부분집합의 합

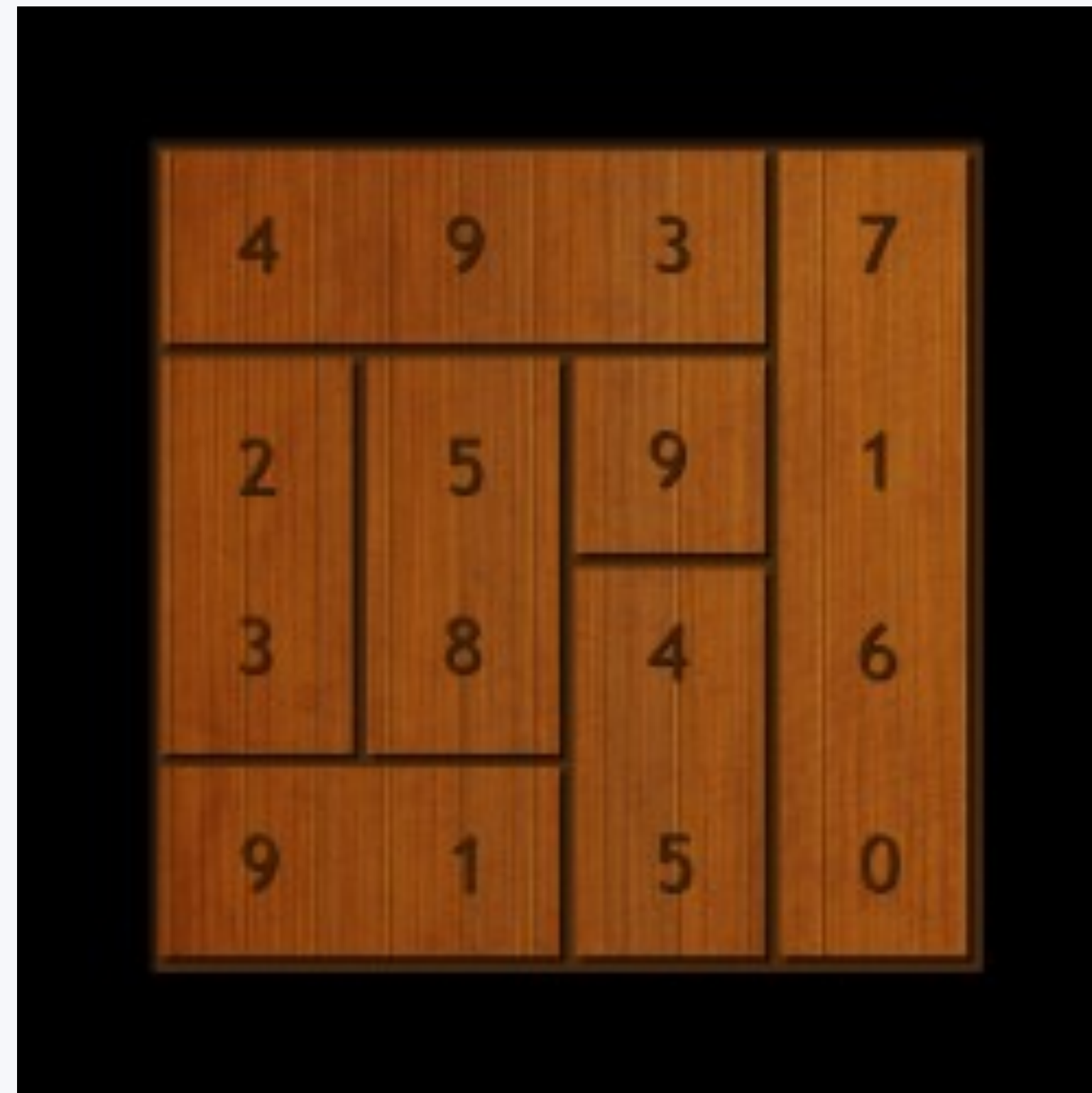
<https://www.acmicpc.net/problem/1182>

- C++: <https://gist.github.com/Baekjoon/f4154089addcd1adacc5>
- Java: <https://gist.github.com/Baekjoon/bddda372acf45d698817>

종이 조각

<https://www.acmicpc.net/problem/14391>

- 종이를 조각으로 잘라서 합의 최대값을 구하는 문제



종이 조각

<https://www.acmicpc.net/problem/14391>

- 각각의 칸에 대해서, 가로(-)인지 세로(|)인지 정하면 된다.

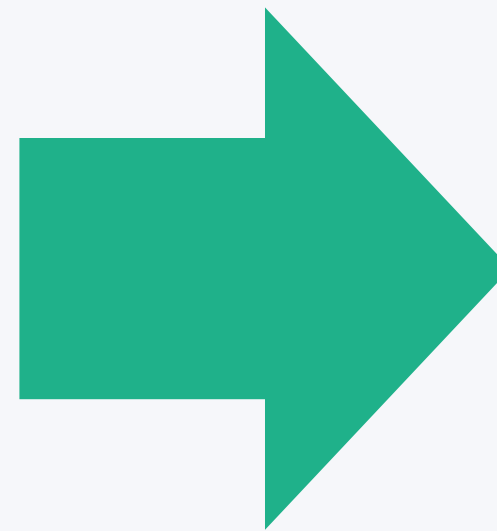
-	-		
-	-	-	
	-		
	-		-

종이 조각

<https://www.acmicpc.net/problem/14391>

- 각각의 칸에 대해서, 가로(-)인지 세로(|)인지 정하면 된다.

-	-		
-	-	-	
	-		
	-		-



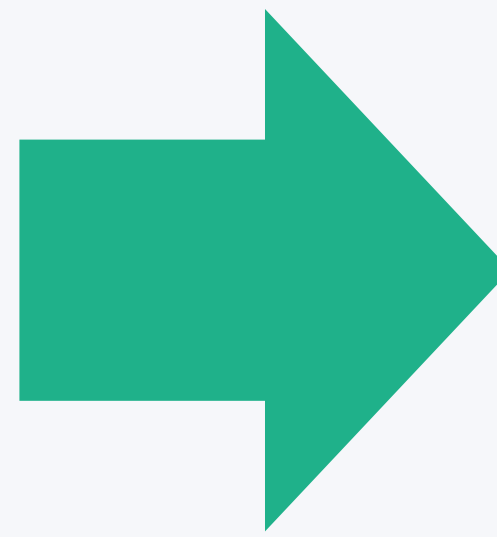
0	0	1	1
0	0	0	1
1	0	1	1
1	0	1	0

종이 조각

<https://www.acmicpc.net/problem/14391>

- 각각의 칸에 대해서, 가로(-)인지 세로(|)인지 정하면 된다.

-	-		
-	-	-	
	-		
	-		-



0	0	1	1
0	0	0	1
1	0	1	1
1	0	1	0

0	0	1	1	0	0	0	1	1	0	1	1	1	0	1	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

종이 조각

76

<https://www.acmicpc.net/problem/14391>

- $2^{(NM)}$ 으로 상태를 만들고 나누어보면 된다

종이 조각

<https://www.acmicpc.net/problem/14391>

- C++: <https://gist.github.com/Baekjoon/39614927a2cae287cfc304615de1d7f5>
- Java: <https://gist.github.com/Baekjoon/85e9bdc01ae9e741a331875a23726c7f>