

# 2022-1 HI-ARC 초급스터디

## 4주차. 백트래킹

이지은 (leeju1013)

# 목차

1. 재귀 함수
2. 백트래킹

# 1. 재귀 함수

- What?
  - 자기 자신을 호출하는 함수
- How?
  - 점차 basecase(기저 조건)에 가까워져야 함

```

1 #include <iostream>
2 using namespace std;
3
4 void func(int level){
5     if(level==0) return;
6     cout<<level<<" Hi~\n";
7     func(level-1);
8     cout<<level<<" Bye~\n";
9 }
10
11 int main() {
12     func(3);
13     return 0;
14 }

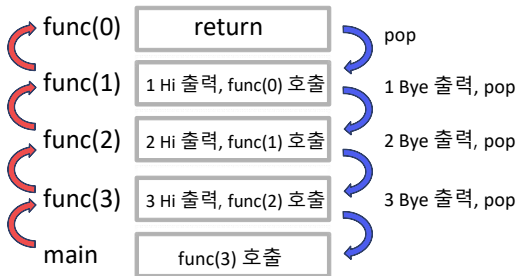
```

stdout

```

3 Hi~
2 Hi~
1 Hi~
1 Bye~
2 Bye~
3 Bye~

```



```

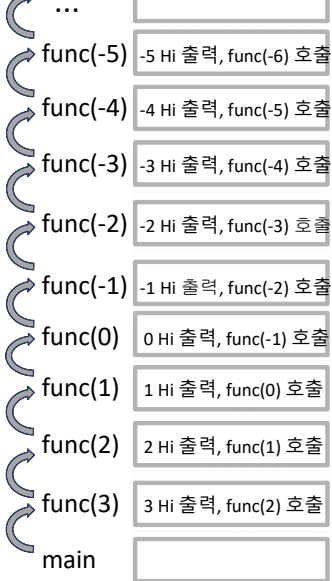
1 #include <iostream>
2 using namespace std;
3
4 void func(int level){
5     //if(level==0) return;
6     cout<<level<<" Hi~\n";
7     func(level-1);
8     cout<<level<<" Bye~\n";
9 }
10
11 int main() {
12     func(3);
13     return 0;
14 }

```

Runtime error

stdout

3 Hi~  
 2 Hi~  
 1 Hi~  
 0 Hi~  
 -1 Hi~  
 -2 Hi~  
 -3 Hi~  
 -4 Hi~  
 -5 Hi~  
 -6 Hi~  
 -7 Hi~  
 -8 Hi~  
 -9 Hi~  
 -10 Hi~



Stack Overflow

## 피보나치 수 5

```

1 #include <iostream>
2 using namespace std;
3
4 int fibo(int n){
5     if(n==0) return 0;
6     else if(n==1) return 1;
7     else return fibo(n-1)+fibo(n-2);
8 }
9 int main(){
10     ios_base::sync_with_stdio(false);
11
12     int n; cin>>n;
13     cout<<fibo(n);
14     return 0;
15 }

```

## 문제

피보나치 수는 0과 1로 시작한다. 0번째 피보나치 수는 0이고, 1번째 피보나치 수는 1이다. 그 다음 2번째 부터는 바로 앞 두 피보나치 수의 합이 된다.

이를 식으로 써보면  $F_n = F_{n-1} + F_{n-2}$  ( $n \geq 2$ )가 된다.

$n=17$ 일때 까지 피보나치 수를 써보면 다음과 같다.

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597

$n$ 이 주어졌을 때,  $n$ 번째 피보나치 수를 구하는 프로그램을 작성하시오.

## 입력

첫째 줄에  $n$ 이 주어진다.  $n$ 은 20보다 작거나 같은 자연수 또는 0이다.

stdin

4

stdin

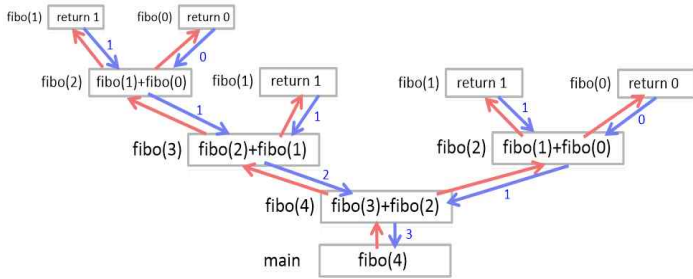
17

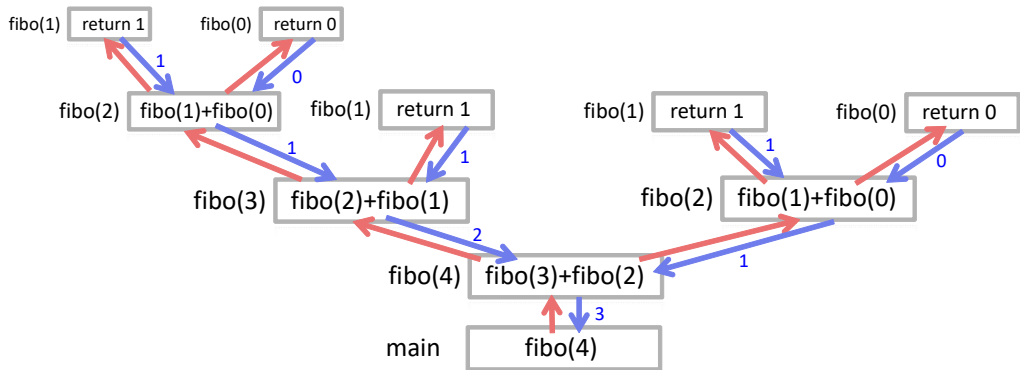
stdout

3

stdout

1597





## 2. 백트래킹(Backtracking)

- What?
  - 해를 찾는 도중 해가 아니어서 막히면,  
되돌아가서(back) 다시 해를 찾아가는(track) 기법
- How?
  - 현재 상태에서 가능한 모든 후보군을 따라 들어가며 탐색
  - 불필요한 부분은 가지치기(Pruning)



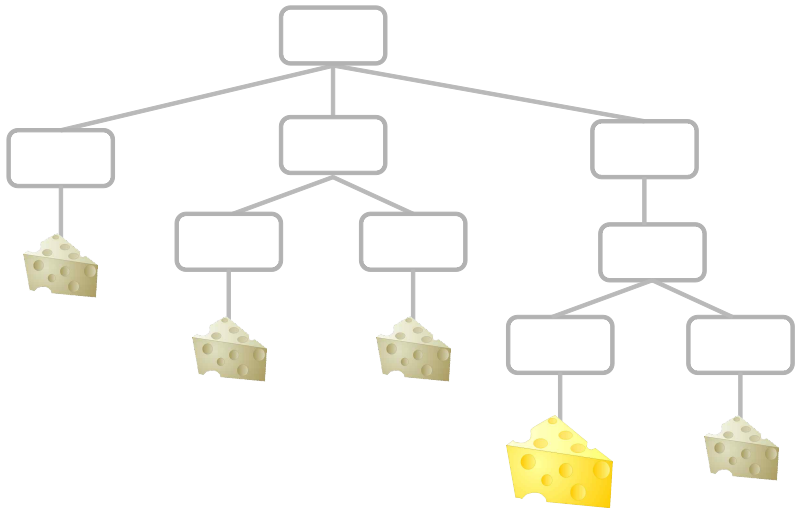


level : 0

level : 1

level : 2

level : 3



N과 M (1) 성공

## 3 실버 III

예제 입력 2 복사

예제 출력 2 복사

시간 제한	메모리 제한	제출	정답
1 초	512 MB	54194	33447

4 2

```

1 2
1 3
1 4
2 1
2 3
2 4
3 1
3 2
3 4
4 1
4 2
4 3

```

## 문제

자연수  $N$ 과  $M$ 이 주어졌을 때, 아래 조건을 만족하는 길이가  $M$ 인 수열을 모두 구하는 프로그램을 작성하시오.

- 1부터  $N$ 까지 자연수 중에서 중복 없이  $M$ 개를 고른 수열

## 입력

첫째 줄에 자연수  $N$ 과  $M$ 이 주어진다. ( $1 \leq M \leq N \leq 8$ )

## 출력

한 줄에 하나씩 문제의 조건을 만족하는 수열을 출력한다. 중복되는 수열을 여러 번 출력하면 안되며, 각 수열은 공백으로 구분해서 출력해야 한다.

수열은 사전 순으로 증가하는 순서로 출력해야 한다.

- $n=4, m=2$ 인 경우 (반복문)

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int n=4, m=2;
6
7     for(int i=1; i<=n; i++){
8         for(int j=1; j<=n; j++){
9             if(i==j) continue;
10            cout<<i<<" "<<j<<'\n';
11        }
12    }
13    return 0;
14 }
```

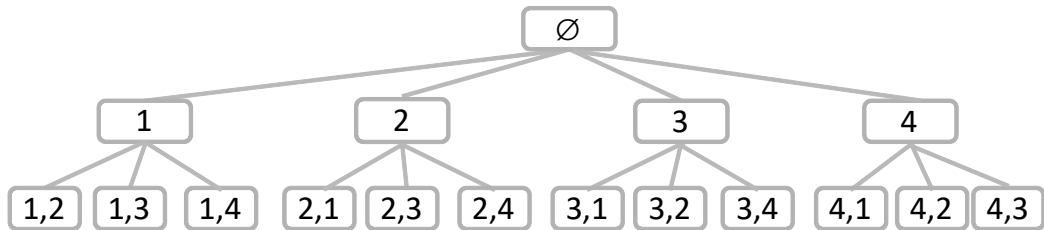
⚙️ stdout

```
1 2
1 3
1 4
2 1
2 3
2 4
3 1
3 2
3 4
4 1
4 2
4 3
```



$m=8$  이라면?? 8중 for문 ..

- $n=4, m=2$ 인 경우 (재귀)



M개가 쌓이면 출력을 하고 되돌아가자!

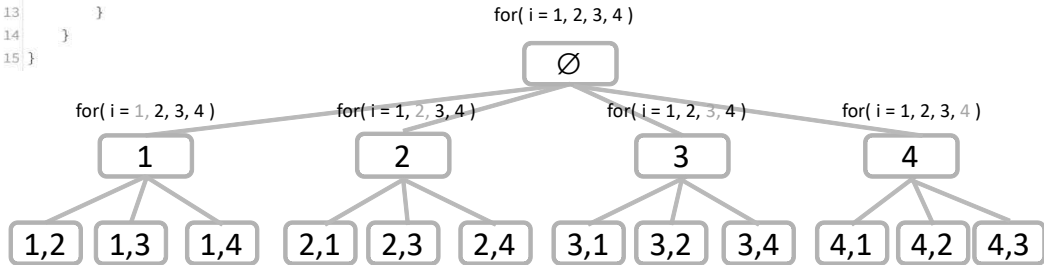
```

1 void bt(){
2     if(v.size()==m){ //base case
3         v의 원소 전부 출력;
4         return;
5     }
6     for(int i=1; i<=n; i++){
7         if(i가 방문하지 않은 숫자라면){
8             i 방문 체크;
9             벡터에 원소 i push;
10            bt(); //재귀 호출
11            i 방문 체크 되돌리기;
12            벡터의 원소 i pop;
13        }
14    }
15 }

```

visited	[0]	[1]	[2]	[3]	[4]
방문 체크 배열	0	0	0	0	0

v	[0]	[1]
현재 고른 원소를 담은 벡터		



```

1 void bt(){
2     if(v.size()==m){ //base case
3         v의 원소 전부 출력;
4         return;
5     }
6     for(int i=1; i<=n; i++){
7         if(i가 방문하지 않은 숫자라면){
8             i 방문 체크;
9             벡터에 원소 i push;
10            bt(); //재귀 호출
11            i 방문 체크 되돌리기;
12            벡터의 원소 i pop;
13        }
14    }
15 }

```

```

1 #include <iostream>
2 #include <vector>
3 using namespace std;
4
5 int n,m;
6 bool visited[9];
7 vector<int> v;
8
9 void bt(){
10     if(v.size()==m){ //base case
11         for(auto &k :v) cout<<k<<" ";
12         cout<<'\\n';
13         return;
14     }
15     for(int i=1; i<=n; i++){
16         if(!visited[i]){
17             visited[i]=1;
18             v.push_back(i);
19             bt();
20             visited[i]=0;
21             v.pop_back();
22         }
23     }
24 }
25 int main() {
26     cin>> n >> m;
27     bt();
28     return 0;
29 }

```

N과 M (2) 성공

## 3 실버 III

## 예제 입력 2 복사

## 예제 출력 2 복사

시간 제한	메모리 제한	제출	정답	맞힌
1 초	512 MB	37093	27811	20:

4 2

```
1 2
1 3
1 4
2 3
2 4
3 4
```

## 문제

자연수 N과 M이 주어졌을 때, 아래 조건을 만족하는 길이가 M인 수열을 모두 구하는 프로그램을 작성하시오.

- 1부터 N까지 자연수 중에서 중복 없이 M개를 고른 수열
- 고른 수열은 오름차순이어야 한다.

## 입력

첫째 줄에 자연수 N과 M이 주어진다. ( $1 \leq M \leq N \leq 8$ )

## 출력

한 줄에 하나씩 문제의 조건을 만족하는 수열을 출력한다. 중복되는 수열을 여러 번 출력하면 안되며, 각 수열은 공백으로 구분해서 출력해야 한다.

수열은 사전 순으로 증가하는 순서로 출력해야 한다.

## 예제 입력 2 복사

4 2

## 예제 출력 2 복사

1 2  
1 3  
1 4  
2 3  
2 4  
3 4

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4
5 int n,m;
6 vector<int> v;
7
8 void bt(int num){
9     if(v.size()==m){ //base case
10         for(auto &k :v) cout<<k<<" ";
11         cout<<"\n";
12         return;
13     }
14     for(int i=num; i<=n; i++){
15         v.push_back(i);
16         bt(i+1);
17         v.pop_back();
18     }
19 }
20 int main() {
21     cin>> n >> m;
22     bt(1);
23     return 0;
24 }
```

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4 int n,m;
5 bool visited[9];
6 vector<int> v;
7
8 void bt(){
9     if(v.size()==m){
10         for(auto &k:v)
11             cout << k << ' ';
12         cout << "\n";
13         return;
14     }
15     for (int i=v.back()+1; i<=n; i++) {
16         v.push_back(i);
17         bt();
18         v.pop_back();
19     }
20 }
21
22 int main(){
23     ios_base::sync_with_stdio(false); cin.tie(NULL);
24     cin >> n >> m;
25     for (int i=1; i<=n; i++){
26         v.push_back(i);
27         bt();
28         v.pop_back();
29     }
30     return 0;
31 }
```



### 예제 입력 2 복사

4 2

### 예제 출력 2 복사

1 2  
1 3  
1 4  
2 3  
2 4  
3 4

```
1 #include <iostream>
2 using namespace std;
3 int n,m;
4 int arr[9];
5
6 void solve(int cur, int idx){ //현재 넣을 숫자, 현재 넣을 인덱스
7     if(idx==m){ //m개(arr[0]~arr[m-1])를 다 정했으면 출력
8         for(int i=0; i<m; i++) cout<<arr[i]<<" ";
9         cout<<"\n";
10        return;
11    }
12    if(cur==n+1) return; //모든 숫자 다 봤으면 리턴
13    arr[idx]=cur; //일단 숫자를 넣음
14    solve(cur+1, idx+1); //idx에 cur넣을 경우, 다음 칸을 전달
15    solve(cur+1, idx); //idx에 cur을 넣지 않는 경우, 현재 칸을 (뛴어썅우라고) 전달
16 }
17 int main(){
18     ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);
19
20     cin>>n>>m;
21     solve(1,0);
22
23     return 0;
24 }
```

부분수열의 합 성공

## 2 실버 II

시간 제한

메모리 제한

제출

정답

맞힌 사람

정답 비율

2 초

256 MB

47738

22013

14105

44.191%

## 문제

$N$ 개의 정수로 이루어진 수열이 있을 때, 크기가 양수인 부분수열 중에서 그 수열의 원소를 다 더한 값이  $S$ 가 되는 경우의 수를 구하는 프로그램을 작성하시오.

## 입력

첫째 줄에 정수의 개수를 나타내는  $N$ 과 정수  $S$ 가 주어진다. ( $1 \leq N \leq 20$ ,  $|S| \leq 1,000,000$ ) 둘째 줄에  $N$ 개의 정수가 빈 칸을 사이에 두고 주어진다. 주어지는 정수의 절댓값은 100,000을 넘지 않는다.

## 출력

첫째 줄에 합이  $S$ 가 되는 부분수열의 개수를 출력한다.

## 예제 입력 1 복사

```
5 0
-7 -3 -2 5 8
```

## 예제 출력 1 복사

```
1
```

```

void bt(현재 보고있는 원소의 인덱스 idx, 이전까지 선택한 원소들의 합 sum){
    if(수열의 끝까지 봤으면) return; //범위 초과
    if(sum + arr[idx] == S) 정답++; //조건 성립

    //현재 idx의 숫자를 더하는 경우
    bt(다음 인덱스, 이전까지 선택한 원소들의 합 + 현재 보고있는 원소의 값);

    //현재 idx의 숫자를 안 더하는 경우
    bt(다음 인덱스, 이전까지 선택한 원소들의 합);
}

```

5 0  
-7 -3 -2 5 8

```

1 #include <iostream>
2 using namespace std;
3
4 int N, S, ans;
5 int arr[21];
6
7 void bt(int idx, int sum){
8     if(idx==N) return; //범위 초과
9     if(sum+arr[idx]==S) ans++; //조건 성립
10
11     bt(idx+1, sum+arr[idx]); //현재 idx숫자 더하는 경우
12     bt(idx+1, sum); //현재 idx숫자 안 더하는 경우
13 }
14
15 int main(){
16     ios_base::sync_with_stdio(false); cin.tie(NULL);
17
18     cin>> N >> S;
19     for(int i=0; i<N; i++) cin>>arr[i];
20
21     bt(0,0);
22     cout<<ans;
23     return 0;
24 }

```

지금까지 선택한 원소들의 합 sum; //전역 선언

```
void bt(현재 보고있는 원소의 인덱스 idx){  
    if(수열의 끝까지 봤으면) return; //범위 초과  
  
    sum+=arr[idx]; //현재 idx의 숫자를 더하는 경우  
    if(sum==S) 정답++; //조건 성립  
    bt(idx+1);  
  
    sum-=arr[idx]; //현재 idx의 숫자를 안 더하는 경우 (값 되돌리기)  
    bt(idx+1);  
}
```

5 0  
-7 -3 -2 5 8

```
1 #include <iostream>  
2 using namespace std;  
3  
4 int N, S, ans, sum;  
5 int arr[21];  
6  
7 void bt(int idx){  
8     if(idx==N) return; //범위 초과  
9  
10    sum+=arr[idx];  
11    if(sum==S) ans++; //조건 성립  
12  
13    bt(idx+1); //현재 idx의 숫자를 더하는 경우  
14    sum-=arr[idx];  
15    bt(idx+1); //현재 idx의 숫자를 안 더하는 경우  
16 }  
17  
18 int main(){  
19     ios_base::sync_with_stdio(false); cin.tie(NULL);  
20  
21     cin>> N >> S;  
22     for(int i=0; i<N; i++) cin>>arr[i];  
23  
24     bt(0);  
25     cout<<ans;  
26     return 0;  
27 }
```

## N-Queen

성공



5 골드 V

시간 제한

메모리 제한

제출

정답

맞힌 사람

정답 비율

10 초

128 MB

60112

29882

19626

49.123%

## 문제

N-Queen 문제는 크기가  $N \times N$ 인 체스판 위에 퀸  $N$ 개를 서로 공격할 수 없게 놓는 문제이다.

$N$ 이 주어졌을 때, 퀸을 놓는 방법의 수를 구하는 프로그램을 작성하시오.

## 입력

첫째 줄에  $N$ 이 주어진다. ( $1 \leq N < 15$ )

## 출력

첫째 줄에 퀸  $N$ 개를 서로 공격할 수 없게 놓는 경우의 수를 출력한다.

예제 입력 1 복사

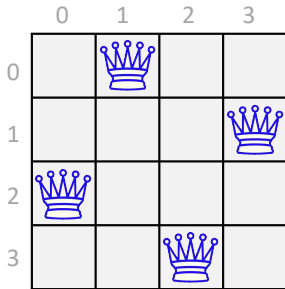
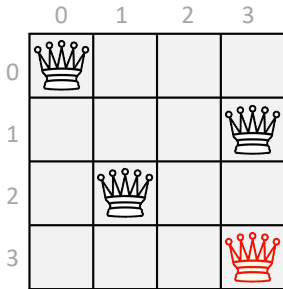
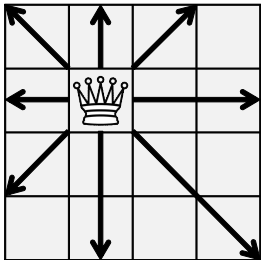
8

예제 출력 1 복사

92

- $n=4$ 인 경우 

:  $4 \times 4$ 인 체스판 위에 퀸 4개를 서로 공격할 수 없게 놓는  
경우의 수를 구하는 문제



상하좌우, 대각선 공격 가능

열

bool check1 [4];

y

y \ x	0	1	2	3
0	0	1	2	3
1	0	1	2	3
2	0	1	2	3
3	0	1	2	3

우상향

bool check2 [7];

$x + y$

y \ x	0	1	2	3
0	0	1	2	3
1	1	2	3	4
2	2	3	4	5
3	3	4	5	6

우하향

bool check3 [7];

$x - y + N - 1$

y \ x	0	1	2	3
0	3	2	1	0
1	4	3	2	1
2	5	4	3	2
3	6	5	4	3





```

1 #include <iostream>
2 using namespace std;
3 bool check1[14]; // N은 최대 14
4 bool check2[27]; //13*2 +1
5 bool check3[27]; //13*2 +1
6 int N,ans;
7
8 void bt(퀸을 배치할 행의 인덱스){ // 0번째 ~ N-1번째 행
9
10     if(퀸 N개를 놓는데 성공했다면){
11         정답++; // 조건 성립
12         return;
13     }
14
15     for(int i=0; i<N; i++){ // 퀸을 배치할 열의 인덱스 i
16
17         if(현재 칸에서 공격가능한 위치에 퀸이 있다면) // check 배열을 이용
18             continue;
19
20         // (level,i)에 퀸을 배치하기 (상하,대각선에 이제 퀸 못놓는다고 체크해주기)
21         check1,2,3 배열에 현재 칸 기준으로 체크
22
23         bt(다음 행의 인덱스); // 다음 행에 퀸 놓으러 가자
24
25         // 아까 체크한거 다시 원상태로 되돌리기
26         check1,2,3 배열에 현재 칸 기준으로 체크한거 원상태로
27     }
28 }

```

(출처)

<https://blog.encrypted.>

	0	1	2	3
0				
1				
2				
3				

```

29 int main() {
30     ios_base::sync_with_stdio(false);
31     cin.tie(NULL); cout.tie(NULL);
32
33     cin>>N;
34     bt(0);
35     cout<<ans;
36     return 0;
37 }

```



```

1 #include <iostream>
2 using namespace std;
3 bool check1[14]; // N은 최대 14
4 bool check2[27]; //13*2 +1
5 bool check3[27]; //13*2 +1
6 int N,ans;
7
8 void bt(int level){ // level번째 행에 퀸을 배치할 예정 (0번째 ~ N-1번째 행)
9
10     if(level==N){ // 퀸 N개를 놓는데 성공했다면
11         ans++;
12         return;
13     }
14
15     for(int i=0; i<N; i++){ // (level,i)에 퀸을 배치할 예정
16
17         if((check1[i])||(check2[level+i])||(check3[level-i+N-1]))
18             continue; // 공격가능한 위치에 퀸이 있다면 패스
19
20         // (level,i)에 퀸을 배치하기 (상하,대각선에 이제 퀸 못놓는다고 체크해주기)
21         check1[i]=true;
22         check2[level+i]=true;
23         check3[level-i+N-1]=true;
24
25         bt(level+1); // 다음 행에 퀸 놓으러 가자
26
27         // 아까 체크한거 다시 원상태로 되돌리기
28         check1[i]=false;
29         check2[level+i]=false;
30         check3[level-i+N-1]=false;
31     }
32 }

```

```

33 int main() {
34     ios_base::sync_with_stdio(false);
35     cin.tie(NULL); cout.tie(NULL);
36
37     cin>>N;
38     bt(0);
39     cout<<ans;
40     return 0;
41 }
42

```

# 감사합니다

- 필수 문제

3 15649번

N과 M (1)

3 15650번

N과 M (2)

2 1182번

부분수열의 합

- 연습 문제

5 17478번

재귀함수가 뭔가요?

N과 M 시리즈

N과 M (3) ~ N과 M (12)

3 18429번

근손실

5 9663번

N-Queen

2 6603번

로또

5 1759번

암호 만들기

- 필수문제 기한 : 4월 26일 (화요일) 저녁 8시까지

- 다음 강의 일정 : 5월 3일(화요일) 저녁 8시

일	월	화	수	목	금	토
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
1	2	3	4	5	6	7