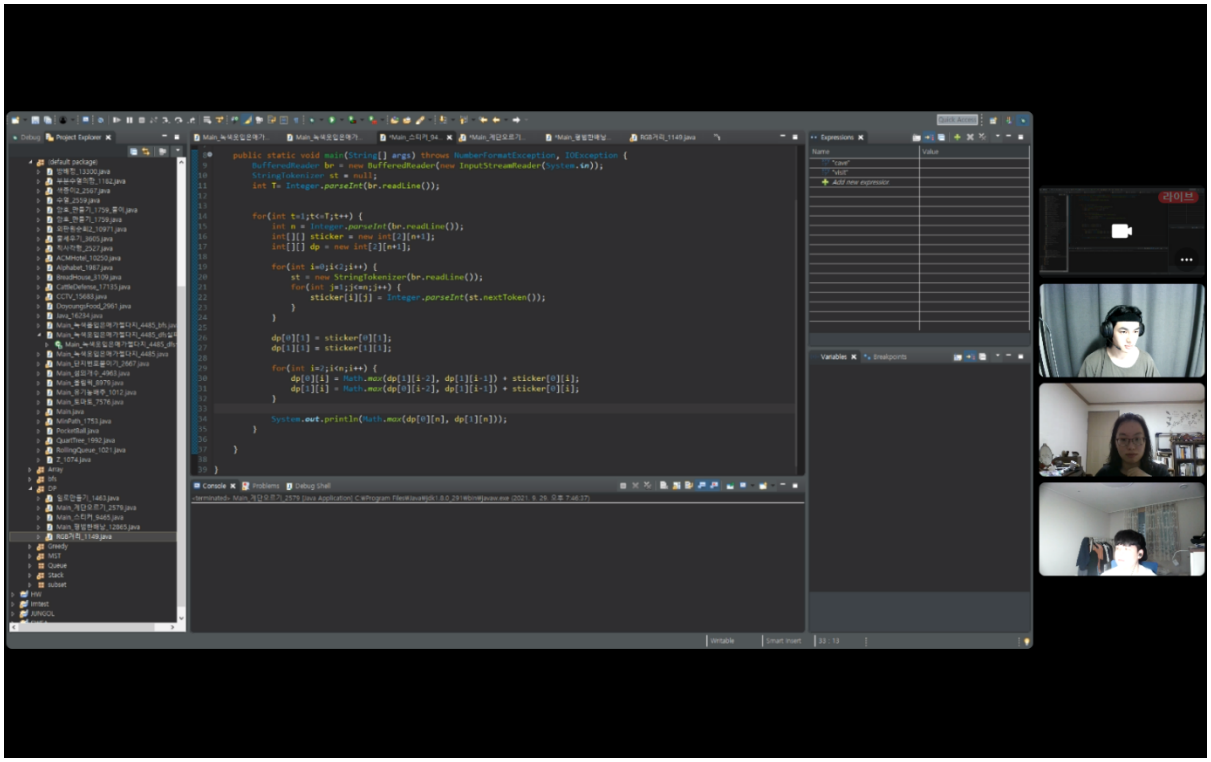
9월 3주차 스터디 팀 발표 자료

3조

김응철 배나영 유우식



1. 백준 9465 스티커

유우식님의 풀이

```java
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int N = sc.nextInt();

    int[] stair = new int[N+1];
    int[] dp = new int[N+1];

    for(int i=1; i<N+1; i++) {
        stair[i] = sc.nextInt();
    }
    //계단이 한칸일 때 그 값이 최대값이므로
    dp[1] = stair[1];
    //연속으로 두칸을 이동할 수 있으므로 처음 두칸을 더한 값이 무조건 최대값
    if(N>=2) {
        dp[2] = stair[1]+stair[2];
    }
    //3번째 계단부터 계산 시작
    for(int i=3; i<N+1; i++) {
        //현 계단의 전 계단 + 3칸 전 계단의 값과 2칸전 계단(연속으로 3칸을 넘어갈 수 없으므로)의 값에서 최대값을 도출한 후
        //현재 계단의 가중치에 더한다.
        dp[i] = Math.max(dp[i-2], (dp[i-3]+stair[i-1])) + stair[i];
    }
    System.out.println(dp[N]);
}
```

배나영님의 풀이

```java
public static void main(String[] args) {

    Scanner sc=new Scanner(System.in);
    int TC=sc.nextInt();

    for(int tc=0;tc<TC;tc++) {
        // 입력 시작
        n=sc.nextInt();
        sticker=new int[2][n];

        for(int i=0;i<2;i++) {
            for(int j=0;j<n;j++) {
                sticker[i][j]=sc.nextInt();
            }
        }
        // 입력 끝

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

            // 0행-자기 자신이 포함되는 경우: 자기자신+앞 열에서 가능한 값
            int include=sticker[0][i]+sticker[1][i-1];

            // 0행-자기 자신이 포함되지 않는 경우 : 자신의 앞 열 두개 중에서 더 큰 것 자유롭게 선택
            int exclude=Math.max(sticker[0][i-1], sticker[1][i-1]);

            // 자기 자신을 포함하는 경우랑 포함하지 않는 경우 중 더 큰 것 선택함
            sticker[0][i]=(include>exclude)?include:exclude;

            // 1행-자기 자신이 포함되는 경우: 자기자신+앞 열에서 가능한 값
            include=sticker[1][i]+sticker[0][i-1];

            // 1행-자기 자신이 포함되지 않는 경우: 앞의 exclude와 같음

            // 자기 자신을 포함하는 경우랑 포함하지 않는 경우 중 더 큰 것 선택함
            sticker[1][i]=(include>exclude)?include:exclude;
        }

        System.out.println(Math.max(sticker[0][n-1], sticker[1][n-1]));
    }
```

2. 백준 2579 계단

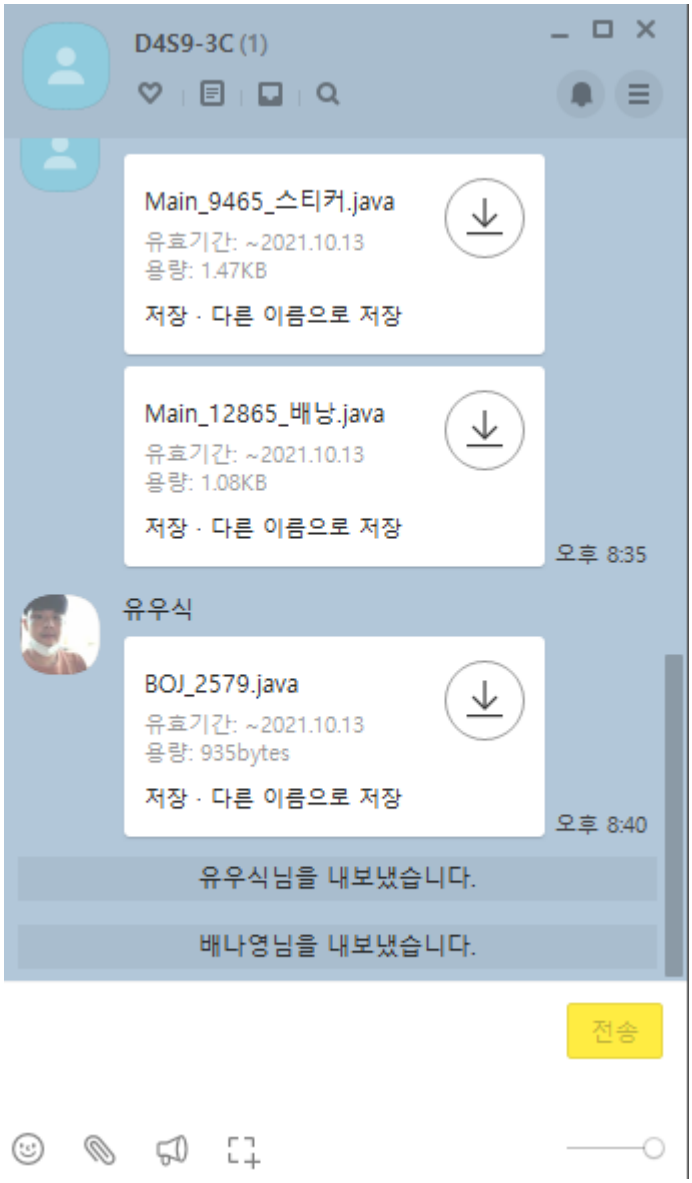
김응철님의 풀이

```java
public class Main_계단오르기_2579 {
    public static void main(String[] args) {


        Scanner sc = new Scanner(System.in);

        int N = sc.nextInt();
        int[] stairs = new int[N+1];
        int[] dp = new int[N+1];

        for(int n=1;n<=N;n++) {
            stairs[n] = sc.nextInt();
        }
        //계단은 한칸 아니면 두칸 이동할 수 있으므로 3칸째는 모두 평등.
        dp[1] = stairs[1];
        dp[2] = stairs[1]+stairs[2];
        dp[3] = Math.max(stairs[1], stairs[2]) + stairs[3];

        for(int i=4;i<=N;i++) {
            dp[i] = Math.max(stairs[i-1] + dp[i-3],dp[i-2])+stairs[i];
        }

        System.out.println(dp[N]);
    }
}
```

3. 백준 12865 평범한 배낭

배나영님의 풀이

```java
static int N, K;
static int[] weights; // 물건의 무게
static int[] values; // 물건의 가치
static int[][] arr;

public static void main(String[] args) {

    Scanner sc=new Scanner(System.in);
    N=sc.nextInt();
    K=sc.nextInt();
    weights=new int[N+1];
    values=new int[N+1];
    arr=new int[N+1][K+1]; // 열이 부분배낭의 용량이라서 K+1개

    for(int i=1;i<=N;i++) {
        weights[i]=sc.nextInt();
        values[i]=sc.nextInt();
    }

    for(int i=0;i<=N;i++) {
        for(int j=0;j<arr[0].length;j++) {
            if(i==0||j==0) arr[i][j]=0;
            else if(weights[i]>j) arr[i][j]=arr[i-1][j];
            else
                arr[i][j]=Math.max(arr[i-1][j], arr[i-1][j-weights[i]]+values[i]); // 선택하지 않는 경우, 선택하는 경우
        }
    }

    print();
    System.out.println(arr[N][K]);
}
```


D4S9-3C (1)
Main_9465_스티커.java
유효기간: ~2021.10.13
용량: 1.47KB
저장 · 다른 이름으로 저장
Main_12865_배낭.java
유효기간: ~2021.10.13
용량: 1.08KB
저장 · 다른 이름으로 저장
오후 8:35
유우식
BOJ_2579.java
유효기간: ~2021.10.13
용량: 935bytes
저장 · 다른 이름으로 저장
오후 8:40
유우식님을 내보냈습니다.
배나영님을 내보냈습니다.
전송