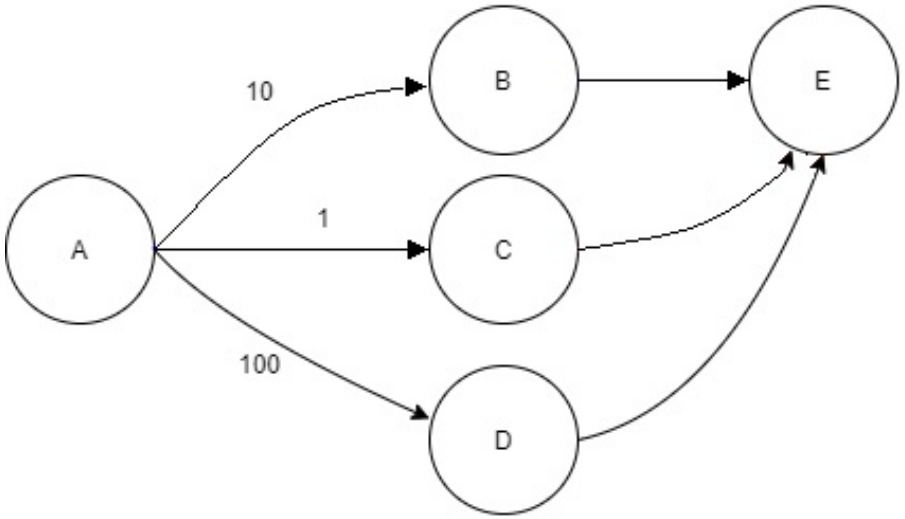
Greedy Algorithm

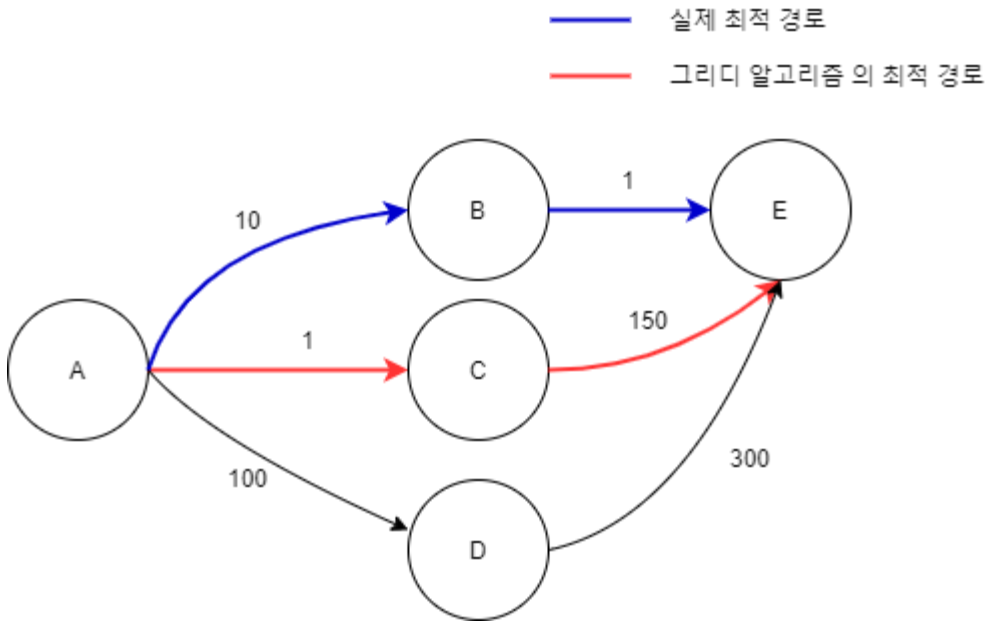
2022 AlgoLive 11th study

Greedy Algorithm이란?

"눈 앞의 이익만을 좇는 알고리즘"

결과가 항상 옳은가?





결과가 항상 옳은가?

-> Local Optimum (지역적 최적) O

-> Global Optimum (최적) X

Greedy Algorithm 사용 조건?

1. 탐욕 선택 속성
2. 최적 부분 구조

탐욕 선택 속성?

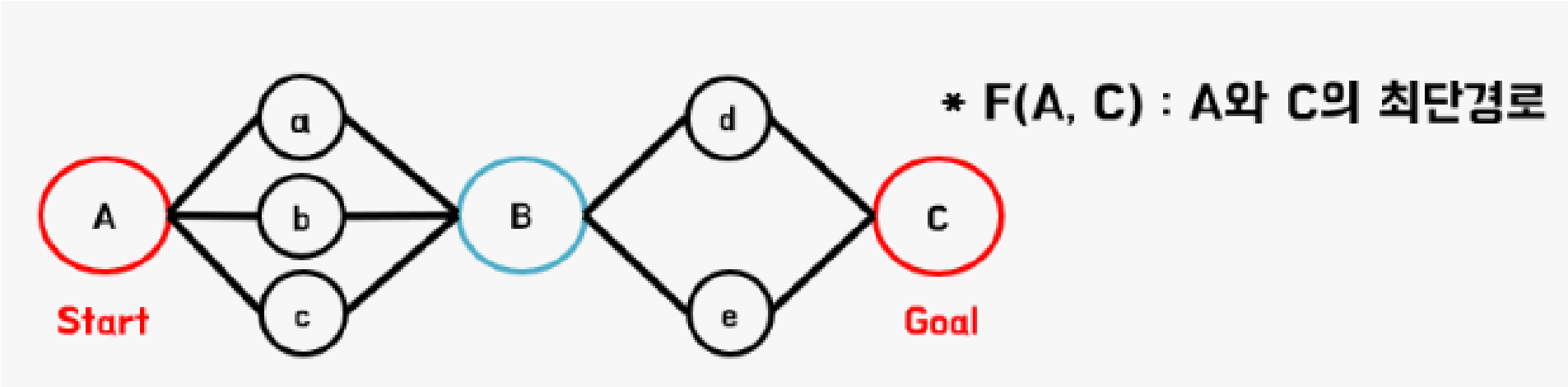
- 탐욕적으로만 선택을 해도 최적해를 구할 수 있다는 뜻
  (현재까지의 최적해)
- 즉, 지역적 최적의 선택이 최적해에 포함되어야 함

최적 부분 구조?

- 전체 문제의 해를 부분 문제로 구할 수 있는 분할 가능한 구조

Greedy Algorithm

2022 AlgoLive 11th study



- 최적 부분 구조

  : F(A,C) = F(A,B) + F(B,C) 로 나타낼 수 있음

- 탐욕적 선택 속성

  : F(A,B)를 구할 때, A의 입장에서 가장 가까운 길을 선택해도,

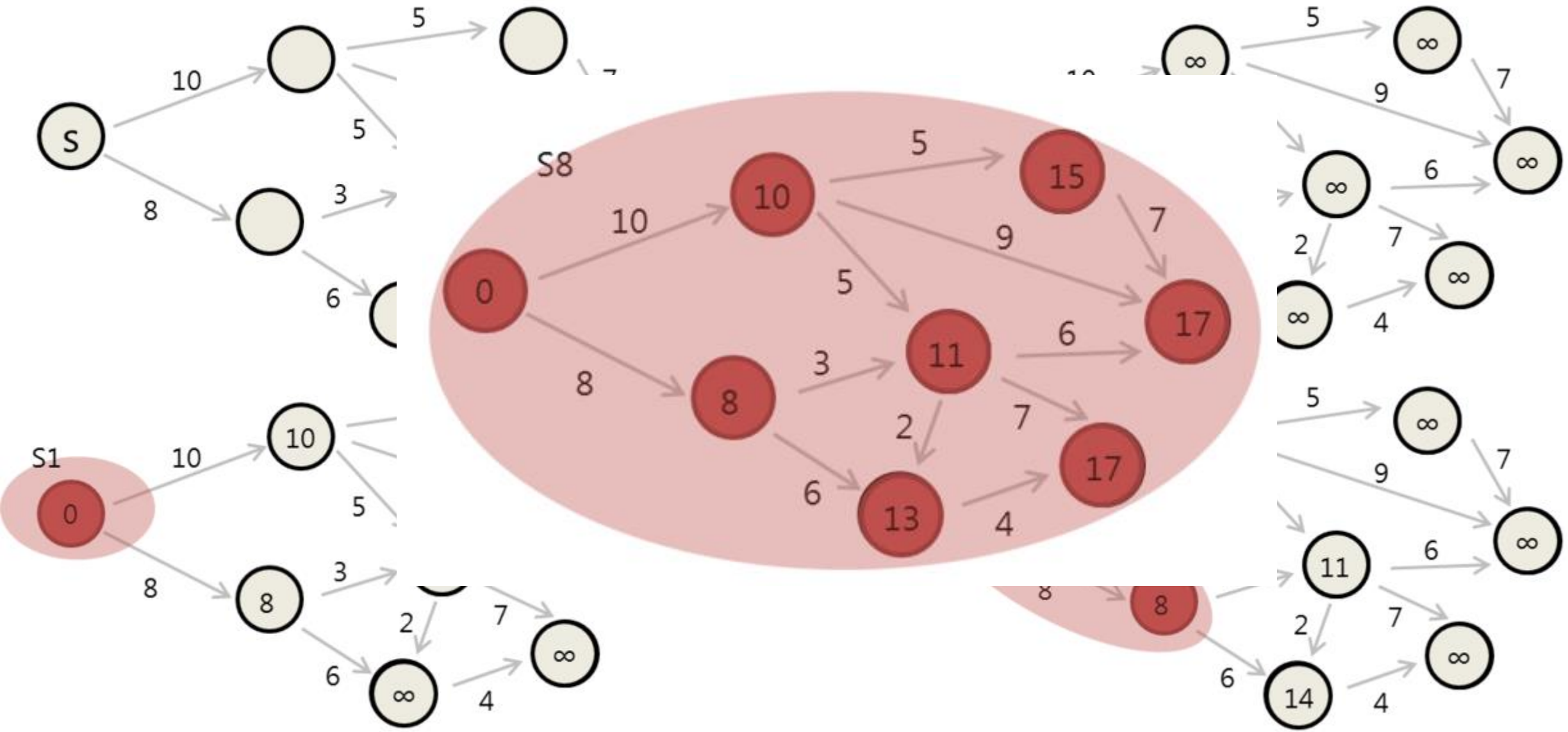
  F(a,B), F(b,B), F(c,B)와 관계 없이 F(A,B)의 해에 포함됨

Greedy Algorithm의 예시

Ex. Dijkstra Algorithm

Greedy Algorithm

2022 AlgoLive 11th study



Greedy Algorithm

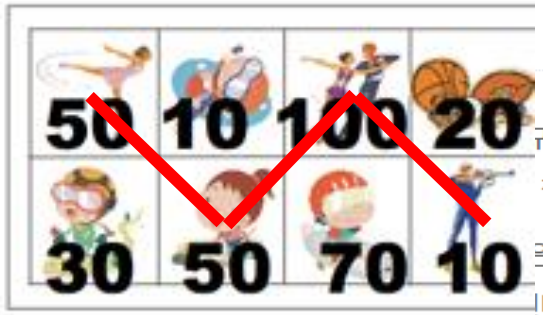
2022 AlgoLive 11th study

스티커 다국어

☆ 한국어

실버 I

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
1 초	256 MB	51053	23740	16163	47.044%

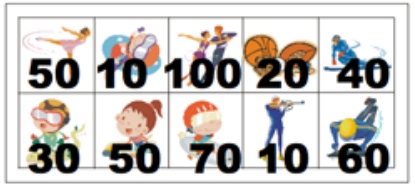


구에서 스티커 2n개를 구매했다. 스티
책상을 꾸미려고 한다.

은 매우 좋지 않다. 스티커 한 장을 떼
다. 즉, 뗀 스티커의 왼쪽, 오른쪽, 위,

다.

(a)



(b)

그래서 언제 사용해야 되나?

4. 사용되는 예시

- AI에 있어서 결정 트리 학습법(Decision Tree Learning)
- 활동 선택 문제(Activity selection problem)
- 거스름돈 문제[5]
- 최소 신장 트리(Minimum spanning tree)
- 제약조건이 많은 대부분의 문제[6]
- 다익스트라 알고리즘
- 허프만 코드
- 크러스컬 알고리즘

단, 동전들에 배수관계가 성립할 때만 한정.

Ex) 1, 5, 10, 25, 50 (O) / 1, 3, 10, 30, 50, (X)

항상 그런 것은 아니지만, 프로그래밍 문제를 풀 때
제약조건이 많다면 대부분 greedy로 풀리는 경우가 많다.
다만 greedy인 줄 알고 풀었다가 피 보는 경우도 있다.

주유소 성공 서브태스크

4 실버 IV

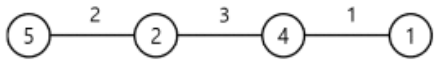
시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
2 초	512 MB	36894	14097	11186	37.939%

문제

어떤 나라에 N개의 도시가 있다. 이 도시들은 일직선 도로 위에 있다. 편의상 일직선을 수평 방향으로 두자. 제일 왼쪽의 도시에서 제일 오른쪽의 도시로 자동차를 이용하여 이동하려고 한다. 인접한 두 도시 사이의 도로들은 서로 길이가 다를 수 있다. 도로 길이의 단위는 km를 사용한다.

처음 출발할 때 자동차에는 기름이 없어서 주유소에서 기름을 넣고 출발하여야 한다. 기름통의 크기는 무제한이어서 얼마든지 많은 기름을 넣을 수 있다. 도로를 이용하여 이동할 때 1km마다 1리터의 기름을 사용한다. 각 도시에는 단 하나의 주유소가 있으며, 도시 마다 주유소의 리터당 가격은 다를 수 있다. 가격의 단위는 원을 사용한다.

예를 들어, 이 나라에 다음 그림처럼 4개의 도시가 있다고 하자. 원 안에 있는 숫자는 그 도시에 있는 주유소의 리터당 가격이다. 도로 위에 있는 숫자는 도로의 길이를 표시한 것이다.



Greedy Algorithm

2022 AlgoLive 11th study

```python
N = int(input())
roads = list(map(int, input().split()))
cities = list(map(int, input().split()))

minVal = cities[0]
sum = 0
for i in range(N-1):
    if minVal > cities[i]:
        minVal = cities[i]
    sum += (minVal * roads[i])

print(sum)
```

```cpp
#include<bits/stdc++.h>

using namespace std;
typedef long long ll;

vector<ll> cost, dis, prefix;

int main(){
    ios::sync_with_stdio(false);
    cin.tie(0);

    int N; cin >> N;
    cost.resize(N);
    dis.resize(N - 1);
    prefix.resize(N);
    for(int i=0;i<N-1;i++) cin >> dis[i];
    for(int i=0;i<N;i++) {
        prefix[i + 1] = prefix[i] + dis[i];
        cin >> cost[i];
    }
    int R = 0;
    ll answer = 0;
    for(int i=0;i<N-1;){
        while(R < N && cost[i] <= cost[R]) R++;
        if(R == N) --R;
        answer += cost[i] * (prefix[R] - prefix[i]);
        i = R;
    }
    cout << answer;

    return 0;
}

```

Greedy Algorithm

2022 AlgoLive 11th study

```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.StringTokenizer;

public class Main {

	public static void main(String[] args) throws NumberFormatException, IOException {
		BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
		int N = Integer.parseInt(br.readLine());

		long[] roads = new long[N-1];
		StringTokenizer st = new StringTokenizer(br.readLine());
		for(int i=0; i<N-1; i++)
			roads[i] = Long.parseLong(st.nextToken());

		long[] cities = new long[N];
		st = new StringTokenizer(br.readLine());
		for(int i=0; i<N; i++)
			cities[i] = Long.parseLong(st.nextToken());

		long min = Long.MAX_VALUE;
		long sum = 0;
		for(int i=0; i<N-1; i++) {
			if(min > cities[i])
				min = cities[i];
			sum += roads[i] * min;
		}
		System.out.println(sum);
	}
}
```

회의실 배정 성공

☆

실버 I

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
2 초	128 MB	131169	40647	28927	29.490%

문제

한 개의 회의실이 있는데 이를 사용하고자 하는 N개의 회의에 대하여 회의실 사용표를 만들려고 한다. 각 회의에 대해 시작시간과 끝나는 시간이 주어져 있고, 각 회의가 겹치지 않게 하면서 회의실을 사용할 수 있는 회의의 최대 개수를 찾아보자. 단, 회의는 한번 시작하면 중간에 중단될 수 없으며 한 회의가 끝나는 것과 동시에 다음 회의가 시작될 수 있다. 회의의 시작시간과 끝나는 시간이 같을 수도 있다. 이 경우에는 시작하자마자 끝나는 것으로 생각하면 된다.

입력

첫째 줄에 회의의 수 N(1 ≤ N ≤ 100,000)이 주어진다. 둘째 줄부터 N+1 줄까지 각 회의의 정보가 주어지는데 이것은 공백을 사이에 두고 회의의 시작시간과 끝나는 시간이 주어진다. 시작 시간과 끝나는 시간은 2^{31}-1보다 작거나 같은 자연수 또는 0이다.

출력

첫째 줄에 최대 사용할 수 있는 회의의 최대 개수를 출력한다.

Greedy Algorithm

2022 AlgoLive 11th study

```python
import sys

def input():
    return sys.stdin.readline().rstrip()

N = int(input())
arr = []
total = 1

for i in range(N):
    a,b = map(int, input().split())
    arr.append((a,b))

arr.sort(key = lambda x : (x[1],x[0]))
end_time = arr[0][1]

for i in range(1,N):
    if arr[i][0] >= end_time:
        end_time = arr[i][1]
        total += 1

print(total)
```

```cpp
#include<iostream>
#include<stdio.h>
#include<algorithm>
#include<vector>
using namespace std;

int value[10];

int main()
{
	int N, end, begin;

	vector<pair<int, int>> schedule;

	cin >> N ;

	for (int i = 0; i < N; i++)
	{
		cin >> begin >> end;
		schedule.push_back(make_pair(end, begin));
	}

	sort(schedule.begin(), schedule.end());

	int time = schedule[0].first;
	int count = 1;
	for (int i = 1 ;i < N; i++)
	{
		if (time <= schedule[i].second )
		{
			count++;
			time = schedule[i].first;
		}
	}

	cout << count;
}
```

Greedy Algorithm

2022 AlgoLive 11th study

```java
import java.util.Scanner;
import java.util.Arrays;
import java.util.Comparator;

public class Main {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);

        int N = in.nextInt();


        /*
          time[][0] 은 시작시점을 의미
          time[][1] 은 종료시점을 의미
        */
        int[][] time = new int[N][2];


        for(int i = 0; i < N; i++) {
            time[i][0] = in.nextInt();  // 시작시간
            time[i][1] = in.nextInt();  // 종료시간
        }

        // 끝나는 시간을 기준으로 정렬하기 위해 compare 재정의
        Arrays.sort(time, new Comparator<int[]>() {

            @Override
            public int compare(int[] o1, int[] o2) {

                // 종료시간이 같을 경우 시작시간이 빠른순으로 정렬해야한다.
                if(o1[1] == o2[1]) {
                    return o1[0] - o2[0];
                }

                return o1[1] - o2[1];
            }

        });

        int count = 0;
        int prev_end_time = 0;

        for(int i = 0; i < N; i++) {

            // 직전 종료시간이 다음 회의 시작 시간보다 작거나 같다면 갱신
            if(prev_end_time <= time[i][0]) {
                prev_end_time = time[i][1];
                count++;
            }
        }

        System.out.println(count);
    }

}
```

행복 유치원 성공

골드 V

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
1 초	512 MB	3553	1925	1576	56.225%

문제

행복 유치원 원장인 태양이는 어느 날 N명의 원생들을 키 순서대로 일렬로 줄 세우고, 총 K개의 조로 나누려고 한다. 각 조에는 원생이 적어도 한 명 있어야 하며, 같은 조에 속한 원생들은 서로 인접해 있어야 한다. 조별로 인원수가 같을 필요는 없다.

이렇게 나뉘어진 조들은 각자 단체 티셔츠를 맞추려고 한다. 조마다 티셔츠를 맞추는 비용은 조에서 가장 키가 큰 원생과 가장 키가 작은 원생의 키 차이만큼 든다. 최대한 비용을 아끼고 싶어 하는 태양이는 K개의 조에 대해 티셔츠 만드는 비용의 합을 최소로 하고 싶어한다. 태양이를 도와 최소의 비용을 구하자.

Greedy Algorithm

2022 AlgoLive 11th study

```python
# Authored by : cieske
# Co-authored by : -
# Link : http://boj.kr/cfb369b6a5134cfa9c0859eab5464c47
import sys
def input():
    return sys.stdin.readline().rstrip()

n, k = map(int, input().split())
lst = list(map(int, input().split())) # 정렬된 상태로 들어옴
sub = sorted([lst[i+1] - lst[i] for i in range(n-1)]) # 원생 간 키 차이 정렬
print(sum(sub[:(n-k)])) # Greedy하게 n-k개만 선택
```

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int main(){
    ios_base::sync_with_stdio(false);
    cin.tie(0);

    int N, K; cin>>N>>K;
    vector<long long int> v(N), cost(N-1);

    for(int i=0; i<N; i++) cin>>v[i];

    sort(v.begin(), v.end());

    for(int i=1; i<N; i++) cost[i-1] = v[i] - v[i-1];

    sort(cost.begin(), cost.end());

    long long int ans = 0;
    for(int i=0; i<N-K; i++) ans += cost[i];

    cout<<ans;
}
```

Greedy Algorithm

2022 AlgoLive 11th study

```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.*;


public class Main {

    static int n, k, result;
    static int[] arr;
    static List<Integer> list = new ArrayList<>();

    public static void main(String[] args) throws IOException {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        String[] s1 = br.readLine().split(" ");
        n = Integer.parseInt(s1[0]);
        k = Integer.parseInt(s1[1]);
        arr = new int[n];
        String[] s = br.readLine().split(" ");
        for (int i = 0; i < n; i++) {
            arr[i] = Integer.parseInt(s[i]);
        }
        Arrays.sort(arr);
        solve();
        System.out.println(result);
    }
    public static void solve() {
        for (int i = 1; i < n; i++) {
            list.add(arr[i] - arr[i - 1]);
        }

        Collections.sort(list);

        for (int i = 0; i < n - k; i++) {
            result+=list.get(i);
        }
    }
}
```

Thank you