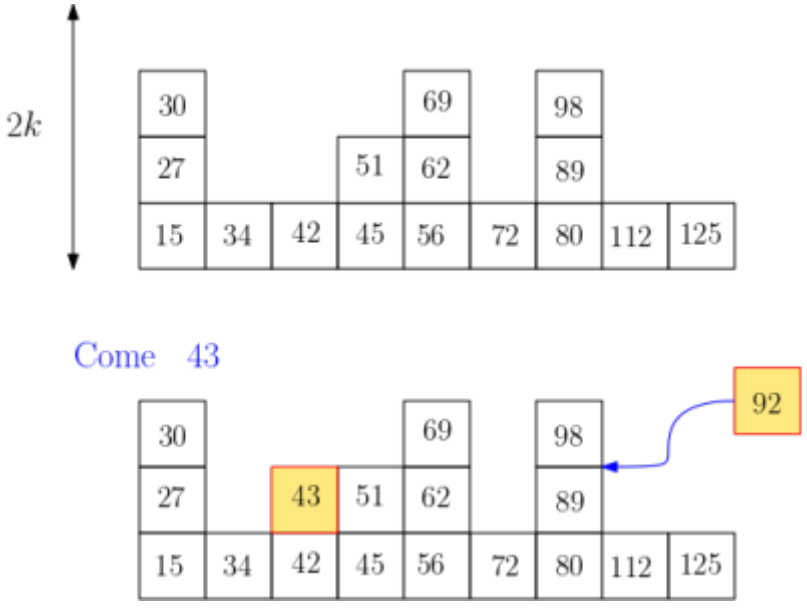
대기실

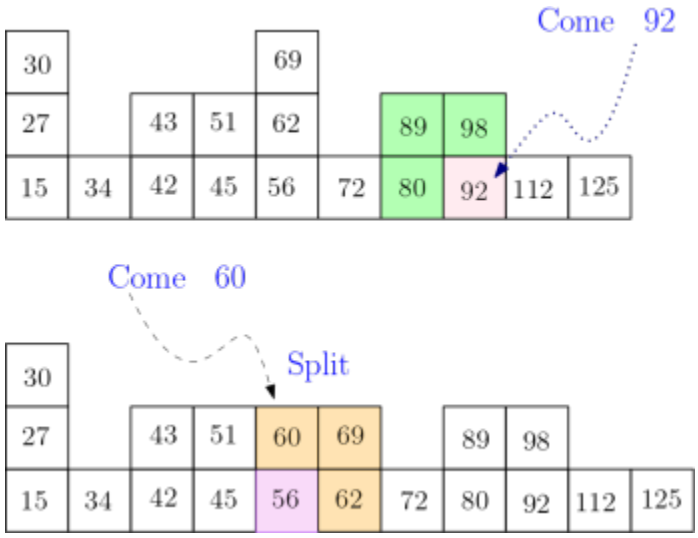
[문제] 벤처 기업 Poogle에서 신입사원 면접을 준비 중이다. 면접실에는 긴 벤치와 같은 모양의 의자가 줄지어 있으며 최대 $2k$ 명이 앉을 수 있다. 이 대기실에 이미 배부된 번호표를 가진 지원자들이 입장하고 퇴장을 한다. 우리는 항상 면접자의 대기 번호 순서가 유지되도록 배정한다. 그런데 한 의자에서 최대 $2k$명이 앉을 수 있기 때문에 특별히 아래와 같은 배정 알고리즘을 운용한다.

1. 모든 지원자는 대기 번호의 오름차순이 유지되도록 배치되어야 한다.
2. 새로 들어온 지원자는 전체 순서를 유지되도록 빈 자리를 찾아 들어간다,
2. 새로운 지원자의 입장으로 의자가 $2k$명으로 꽉 차면 새로운 벤치를 추가 배치하여, k명이 앉는 2개의 의자로 분할(Split), 재배치를 한다.
3. 면접 후 당사자가 퇴장을 하게되면 빠지고 전체 순서는 그대로 유지된다.
4. 만일 퇴장 면접자로 한 의자가 완전히 비면 그 위치의 의자는 삭제하고 뒤 의자를 모두 앞으로 당겨서 빈 곳이 없도록 재조정을 한다.

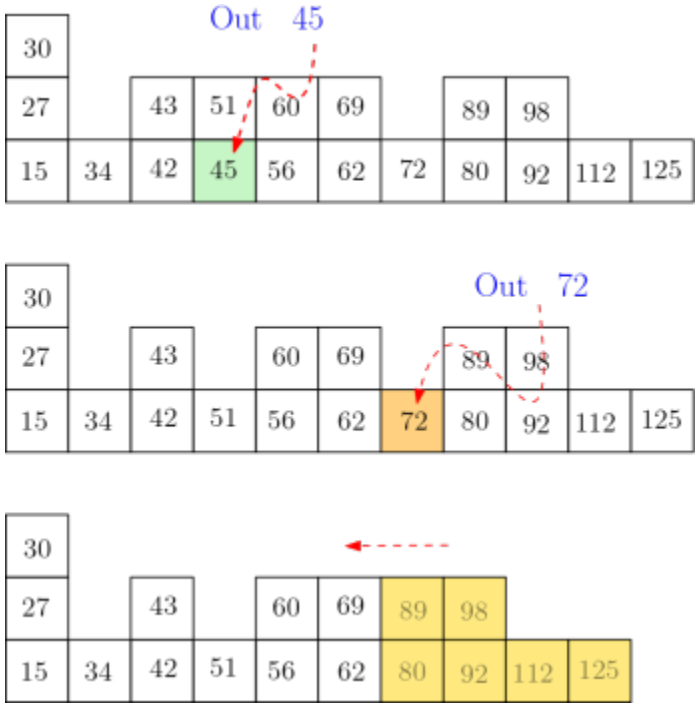
다음 예를 통하여 이 과정을 설명해보자. $2k=4$, 즉 한 의자에 최대 4명까지 앉을 수 있다. 아래 첫 그림은 한 상태를 보여주고 있다. 각 박스는 앉아있는 고객을 나타내고 그 안의 숫자는 대기번호를 의미한다. 대기실의 의자는 더 앞쪽으로 같은 의자에서는 왼쪽(아랫쪽) 에 더 빠른 번호의 고객을 배치한다. 이 상태에서 '43'번 고객이 입장하면 3번째 의자의 끝에 앉게된다. 그 이후 '92'번이 입장하면 이 사람은 7번째 의자 끝에 앉아야 한다. 그런데 이 사람을 배치하면 의자가 꽉 차게 되므로 이런 상황이 되면 split이 발생한다. 즉 [80, 89, 92, 98]이 분할(split) 하여 새로운 2개의 의자에 배치된다.



이 상황이 다음 그림에 표시되어 있다. 그리고 이 상황에서 “60”이 추가하면 다시 SPLIT을 하여 그 아래와 같이 조정된다.



만일 이 상황에서 ‘45’, ‘72’가 퇴장을 하면 아래와 같이 변화한다. 즉 72가 퇴장하면 그 혼자 앉아 있든 의자는 비게 되므로 앞서 설명한대로, 이 의자는 삭제하고 뒤에 있는 모든 의자를 는 앞쪽으로 한 칸씩 이동하게 된다. 즉 의자 사이에 빈 공간이 없도록 compaction을 한다.



여러분에게는 N명의 면접자들이 입장하고 퇴장한 순서가 주어진다. 이 순서로 진행할 때 최종 상황에서 각 의자의 첫 번째 번호(제일 아래 자리)를 순서대로 출력해야 한다. 순서는 앞에서부터 이므로 이 번호는 반드시 오름차순으로 구성되어야 한다. 위와 같은 경우라면 제일 아래 쪽에 제시된 상황이 그 끝이므로 정답 순서는 **`[15, 34, 42, 51, 56, 62, 80, 92, 112, 125]`**가 되어야 한다.

[입출력] 표준 입출력을 사용한다. 첫 줄에 입출입 사건(transaction)의 수 N과 k가 주어진다. [1] 그 범위는 $5 \le N \le 10{,}000$, $2 \le k \le 10$ 이다. 이어지는 N개의 각 줄(line)에는 들어오고 나간 면접자의 번호(대기표 번호) p_i가 주어진다. $1 \le p_i \le 1{,}000$.

입장의 경우에는 '$+\ p_i$' 로 표시되고, 면접을 마치고 나간 경우에는 '$-\ p_i$'로 표시된다. 단 퇴장의 경우 p_i가 대기실에 없는 상황이라면 대기실의 변화는 없다. 경우에 따라서 $p_i = p_j$일 수도 있다.

여러분은 stdout에 제시된 N건의 입출입 사건이 모두 종료된 경우 각 대기실 의자의 첫 번째 고객 번호를 순서대로 출력해야 한다. 즉 위 그림의 경우라면 제일 마지막 상태에서 각 의자를 표시한 slot의 제일 밑바닥에 있는, 즉 [15, 34, 42, 51, 56, 62, 80, 92, 112, 125] 를 출력해야 한다. 한 줄에 하나씩 순서대로 출력한다. 단 마지막 상황에서 대기실에는 반드시 1명 이상의 고객이 있다고 가정한다.

[예제]

stdin	**stdout**
10 2 //N=10, $k=2$ + 80 + 34 + 54 + 78 + 101 - 54 + 32 + 501 - 34 - 200	32 78 101

[제한조건] 프로그램 이름은 **waiting.{c,cpp,py}**이다. 이번 문제는 STL vector를 사용하기를 권고한다. 즉 원소가 vector인 벡터, 즉 이차원 구조의 벡터, 즉 **vector < vector<int> > Waitroom ;** 를 사용하면 좀 더 간력하게 구현할 수 있다.

1) 대부분의 코딩 테스트와 같이 특별한 언급이 없을 경우 모든 정수는 32비트 int 범위이다.