1 이상의 정수를 받아서 다음의 규칙에 따른 “작업”을 반복하여 결국 1 을 만드는 게임을 하려고 한다. 아래 규칙은 한번의 작업에 대한 것이고, 작업의 결과로 만들어지는 수에 작업을 수행하는 것을 반복한다. 규칙에도 나와 있듯이 현재 수가 1 인 경우는 작업을 하지 않고 멈춘다.
1. 만약 수가 1 이면 작업을 하지 않고 멈춘다. 2. 만약 수가 1 이 아닌 홀수이면 1 을 더한다. 3. 만약 수가 짝수이면 2 로 나눈다.
예를 들어, 받은 수가 2 인 경우는 2→1 로 1 회의 작업 후에 멈춘다. 받은 수가 4 인 경우는 4→2→1 로 2 회의 작업 후에 멈춘다. 받은 수가 3 인 경우는 3→4→2→1 로 3 회의 작업 후에 멈춘다. 받은 수가 6 인 경우는 6→3→4→2→1 로 4 회의 작업 후에 멈춘다.
앞의 예 들에서 볼 수 있듯이, 받은 수가 3 인 경우의 작업 횟수를 알면 받은 수가 6인 경우의 작업 횟수를 바로 계산할 수 있다는 것을 알 수 있다.
두 정수 N_1과 N_2를 입력으로 받아서 $(1 \leq N_1 \leq N_2 \leq 106), N_1, N_1 + 1, N_1 + 2, \cdots, N_2$의 작업 회수를 모두 더한 값을 계산하는 프로그램을 작성하라.
- 제한시간: 전체 테스트 케이스는 10,000개 이하이며, 전체 수행 시간은 1초 이내. (Java 2초 이내)
제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며, 그때까지 수행한 결과에서 테스트 케이스를 1개 그룹 이상 통과하였더라도 점수는 0점이 됩니다. 그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 ’부분 점수(0 < 점수 < 만점)’를 받을 수 있으며, 이를 위해서는, C / C++ 에서 ”printf 함수” 사용할 경우, 프로그램 시작부분에서 ”setbuf(stdout, NULL);”를 한번만 사용하십시오. C++에서는 ”setbuf(stdout, NULL);”와 ”printf 함수” 대신 ”cout”를 사용하고, Java에서는 ”System.out.printIn”을 사용하시면, 제한 시간을 초과하더라도 ’부분 점수’를 받을 수 있습니다.
※ 언어별 기본 제공 소스코드 내용 참고
만약, 제한 시간을 초과하지 않았는데도 ’부분 점수’를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.
- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB - 제출 제한 : 최대 10회 (제출 횟수를 반영하여 순위 결정) 메모리 사용 제한
heap, global, static (총계) : 256MB
stack : 100MB
입력
입력 파일에는 여러 테스트 케이스가 포함될 수 있다.
파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 T가 주어지고,
이후 차례로 T 개의 테스트 케이스가 주어진다. $(1 \leq T \leq 10,000)$
각 테스트 케이스의 첫 줄에는 정수 N_1과 N_2가 주어진다. $(1 \leq N1 \leq N2 \leq 106)$
- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 100점)
주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며, 각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.
丶 그룹 1 (34점) : 이 그룹의 테스트 케이스에서는 $1 \leq N1 \leq N2 \leq 103$.
丶 그룹 2 (66점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

출력

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며,

각 테스트 케이스마다 첫 줄에는 “Case #C”를 출력하여야 한다. 이때 C는 테스트 케이스의 번호이다.

그 다음 줄에, $N_1, N_1 + 1, N_1 + 2, \cdots, N_2$의 작업 회수를 모두 더한 값을 출력한다.

입출력예

입력

2

3 6

6 6

출력

Case #1

14

Case #2

4