대회 규칙

- 사용 가능한 언어는 **C11**, **C++17**, **Java 11**, **Python 3.7.3**입니다.
  - 모든 문제에 대해 제약 조건을 만족하며 정답을 출력하는 C++17 코드가 있음이 보장됩니다.
- 대회는 대회 전용 DomJudge 사이트에서 치뤄지며 문제, 채점 실시간 정보 등을 확인할 수 있습니다.
- 순위는 푼 문제가 많은 순서대로, 푼 문제 수가 같을 경우에는 패널티의 합이 낮은 순으로 정렬됩니다.
  - 문제별 패널티는 '(문제를 풀기까지 걸린 시간(분)) + (그 전까지 제출한 횟수) × 20'입니다.
  - 컴파일 에러는 제출 횟수에 포함되지 않습니다.

금지 / 제한 행위

- 대회가 진행되는 동안 화장실 등을 다녀오는 것은 자유이나, 층 별 이동은 제한됩니다.
- 대회 중도 퇴실은 불가합니다.
- 컴퓨터를 두 대 이상 사용하는 것을 금합니다.
- 운영진에게 질문하는 것 외에 다른 사람과 대화하는 것을 금합니다.
- 사전에 코드를 미리 작성해 와서 사용하는 것을 금합니다.
- 허용된 레퍼런스 페이지를 제외한 메신저, 인터넷 검색, 대화, 이동식 저장 매체를 통한 문제 풀이를 금합니다.
- 문제 제출을 비정상적으로 많이 시도하거나, 의도적으로 대회 웹 서버를 공격하는 행위를 금합니다.

대회 규칙을 어기거나, 운영진이 판단하기에 부정한 행위를 저지를 경우 경고 없이 대회 참가 자격이 박탈될 수 있습니다.

레퍼런스 사이트

다음 레퍼런스 사이트는 열람할 수 있습니다.

- **C/C++** : `https://en.cppreference.com/w/`, `http://cplusplus.com`
- **Java** : `https://docs.oracle.com/en/java/javase/11/docs/api/index.html`
- **Python** : `https://docs.python.org/3/`

DOMjudge 채점

DOMjudge에 코드를 업로드할 때는 다음 조건을 지켜야 합니다.

- 파일 이름은 알파벳 및 숫자로 시작해야 하며, 알파벳 대소문자 / 숫자 / `+._-`만 사용 가능합니다.
- 확장자는 C는 `.c`, C++는 `.cpp` / `cc` / `cxx` / `c++`, Java는 `.java`, Python은 `.py` / `.py3`여야 합니다.
- **제출한 코드는 표준 입출력만으로 통신하여야 합니다 (파일 입출력은 금지됩니다).**
- 제출한 소스코드의 크기는 256 MiB 이하여야 합니다.

DOMjudge 채점 결과

Submit을 한 다음에 Scoreboard 탭에서 제출 결과를 확인할 수 있습니다.

- PENDING : 제출되었으며, 채점 대기중이거나 채점중입니다.
- CORRECT : 제출한 코드가 모든 테스트 케이스에 대해 시간 제한 / 메모리 제한 내에서 올바른 답을 내었고, 정상적으로 종료되었습니다. 이 경우 제출자는 해당 문제를 **풀었습니다**.
- COMPILER-ERROR : 컴파일 과정 중에 에러가 발생하여 채점이 진행되지 않았습니다.
- TIMELIMIT : 프로그램 수행 시간이 제한 시간을 초과하였습니다.
- RUN-ERROR : 프로그램 수행 중 에러가 발생하였습니다. (예시 : 0으로 나누기, 잘못된 주소 참조)
- WRONG-ANSWER : 프로그램이 (일부 테스트 케이스에 대해) 오답을 출력하였습니다.
- OUTPUT-LIMIT : 프로그램이 지나치게 많은 출력을 하였습니다.

대회 중 'request clarification' 탭을 통해 주최진에게 질문을 물을 수 있습니다.

컴파일 옵션

사용가능한 언어와 컴파일 옵션은 다음과 같습니다. `"$@"`는 업로드한 코드 및 생성된 프로그램 이름입니다.

- **C11** (gcc 8.3.0)
  컴파일 : `gcc -x c -Wall -O2 -std=c11 -static -pipe -o "$DEST" "$@" -lm`
  실행 : `exec "$@"`
- **C++17** (g++ 8.3.0)
  컴파일 : `g++ -x c++ -Wall -O2 -std=c++17 -static -pipe -o "$DEST" "$@" -lm`
  실행 : `exec "$@"`
- **Java** (Java 11.0.4)
  컴파일 : `javac -encoding UTF-8 -sourcepath . -d . "$@" 2> "$TMPFILE"`
  실행 : `java -Dfile.encoding=UTF-8 -XX:+UseSerialGC -Xss${MEMSTACK}k -Xms${MEMLIMITJAVA}k \ -Xmx${MEMLIMITJAVA}k '$MAINCLASS' "\$@"`
  - MEMSTACK은 65536이며, MEMLIMIT은 문제의 제한에서 128MB를 뺀 값입니다.
- **Python 3** (Python 3.7.3, **PyPy 추가 예정**)
  컴파일 : `python3 -m py_compile "$@"`
  실행 : `python3 "$@"`

이 대회는 DOMjudge 및 채점 환경에 익숙해지기 위한 연습 대회입니다.

문제 목록

대회는 **12시간** 동안 진행되며, 총 **4문제**로 구성되어 있습니다.
총 문제지가 표지를 제외하고 9쪽인지 확인하시길 바랍니다.
문제의 목록은 다음과 같습니다.

A	Hello, KCPC!!	(kcpc)
B	피보나치 수가 아닌 수	(not-fibo)
C	삼각형 세기	(triangle)
D	이항계수 문제로 보이지만...	(binom)

문제 A. Hello, KCPC!! (kcpc)

시간 제한: 1 초
메모리 제한: 512 MB

고려대학교 프로그래밍 경시대회(KCPC)에 참가하게 되신 걸 진심으로 축하드립니다!

"Hello world!"를 처음 출력했던 마음으로 "Hello, KCPC 2019!!"를 따옴표를 제외하고 출력해봅시다.

입력 형식

입력은 없다.

출력 형식

첫 번째 줄에 "Hello, KCPC 2019!!"를 따옴표를 제외하고 출력한다.

예제

표준 입력	표준 출력
	Hello, KCPC 2019!!

문제 B. 피보나치 수가 아닌 수 (not-fibo)

시간 제한: 1 초
메모리 제한: 512 MB

$a_1 = 1$, $a_2 = 1$이라 잡고 $a_n = a_{n-1} + a_{n-2}$ $(n \geq 3)$으로 정의되는 수열 $\{a_k\}$ 를 피보나치 수열이라 한다. 처음 10개의 항을 나열하면 1, 1, 2, 3, 5, 8, 13, 21, 34, 55이다.

피보나치 수열을 나열하는 것은 쉽다. 그럼 피보나치 수열에 포함되지 않는 양의 정수들을 작은 순서대로 나열하는 건 어떨까? 처음 10개의 항을 나열하면 4, 6, 7, 9, 10, 11, 12, 14, 15, 16이다. 피보나치 수열에 없는 수 중 n번째로 작은 수를 구해보자.

입력 형식

첫 번째 줄에는 양의 정수 n이 주어진다. $(1 \leq n \leq 1,000,000)$

출력 형식

첫 번째 줄에 위에서 정의된 피보나치 수열에 포함되지 않는 양의 정수 중 n번째로 작은 수를 출력한다.

예제

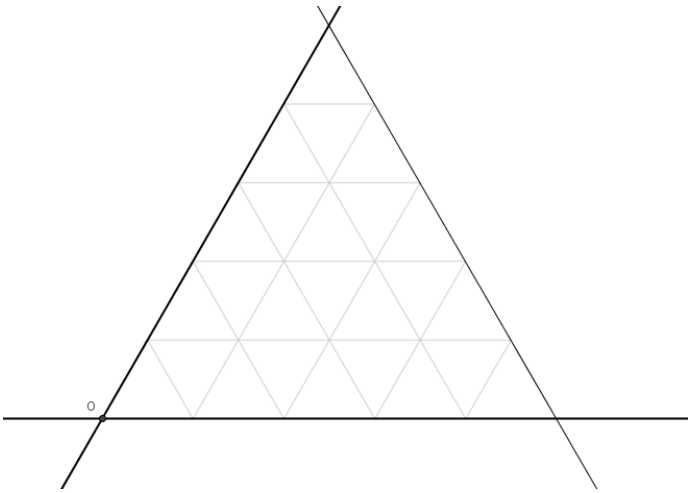
표준 입력	표준 출력
1	4
4	9
12	18

문제 C. 삼각형 세기 (triangle)

시간 제한: 2 초
메모리 제한: 512 MB

수빈이는 원을 좋아한다. 그러나 원과 관련된 문제들이 야기하는 예외 처리 때문에 수빈이는 원에 싫증이 났고, 그 관심을 삼각형으로 옮겼다.

삼각격자(triangular grid, isometric grid)에 대한 설명을 하고 넘어가자. 일반적인 y축이 x축과 직각을 이루는 직각좌표계와는 달리, 삼각격자는 다음 그림과 같이 x축과 y축이 이루는 각도가 60도이다.



60도라는 특성상 정삼각형이 많이 그려진다. $(0,0)$, $(m,0)$, $(0,m)$의 정삼각형을 만드려면 $x=0$, $y=0$, $x+y=m$ 이 세 직선만 있으면 되기 때문이다.

수학적인 호기심이 뛰어난 수빈이는 이 상태에서 세 변 중 하나에 평행하면서 이 삼각형의 내부를 지나가는 직선들을 많이 그렸다. 이를 다 그리고 나서 수빈이는 삼각격자 안에 수많은 정삼각형들이 나타난다는 것을 알아냈고, 여러분을 골탕 먹이려고 정삼각형이 몇 개가 있는지 질문을 하려고 한다.

주어진 정삼각형 안에 완전히 포함되는 정삼각형이 몇 개가 만들어지는지 구하는 프로그램을 짜서 이 위기를 모면하자!

입력 형식

첫 번째 줄에는 정삼각형의 크기인 정수 m과 수빈이가 새로 그은 직선의 개수 q가 공백으로 구분되어 주어진다. $(1 \le m \le 10^9, 0 \le q \le min(500, 3m-3))$

정삼각형의 꼭짓점은 삼각격자에서 $(0,0)$, $(m,0)$, $(0,m)$이다.

그 다음 q개의 줄에는 각각 두 개의 정수 d와 l이 공백으로 구분되어 주어진다. $(0 < l < m)$ d는 x축과 이루는 각도를 의미하며, 이는 0, 60, 120 중 하나이다.

- d가 0일 경우, 직선 $y=l$이 추가된다.
- d가 60일 경우, 직선 $x=l$이 추가된다.
- d가 120일 경우, 직선 $x+y=l$이 추가된다.

한 직선이 2번 이상 입력되는 경우는 없다.

출력 형식

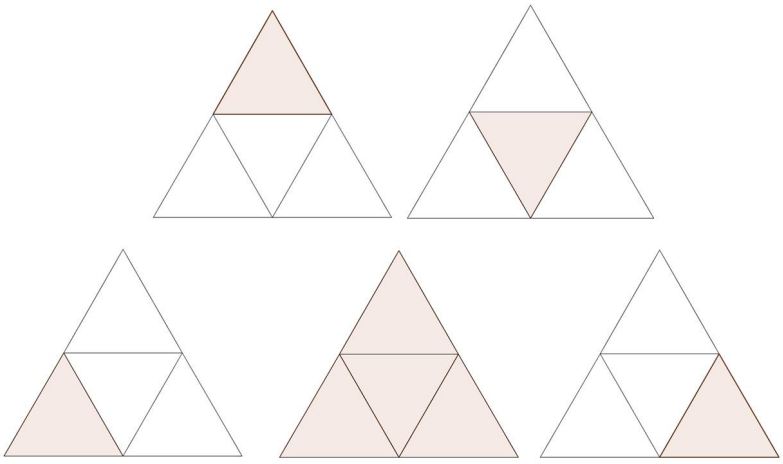
첫 번째 줄에 정삼각형들의 총 개수를 나타내는 자연수를 출력한다.

예제

표준 입력	표준 출력
2 3 0 1 60 1 120 1	5
10 3 120 2 0 1 60 8	6

설명

첫 번째 예제에 대한 설명은 다음 그림에 나와 있다.



문제 D. 이항계수 문제로 보이지만... (binom)

시간 제한: 1 초
메모리 제한: 512 MB

${}_nC_r$로도 표현되는 이항 계수 $\binom{n}{r}$이 $\frac{n!}{r!(n-r)!}$으로 정의된다는 점은 익히 들어보았을 것이다. 수학에 관심이 있다면, 파스칼의 삼각형을 통해 다음 점화식이 성립한다는 것도 알 것이다.

$$\binom{n}{r} = \binom{n-1}{r-1} + \binom{n-1}{r}$$

때문에 이항 계수의 계산은 동적 계획법에서 memoization의 대표적인 예시로 쓰인다. 의사 코드를 작성해보면 다음과 같다.

```
def binom(n, r):
    if dp[n][r] > 0:
        return dp[n][r]
    if n == 0:
        return dp[n][r] = 1
    if r == 0:
        return dp[n][r] = binom(n-1, r)
    if r == n:
        return dp[n][r] = binom(n-1, r-1)
    return dp[n][r] = binom(n-1, r) + binom(n-1, r-1)
```

여기서 `dp[n][r]`은 $\binom{n}{r}$을 저장해놓는 배열로, 처음에는 0으로 초기화되어 있으나 값이 한 번 계산되면 양의 정수로 갱신된다. 때문에 첫 번째 조건문에 의해, 한 번 조사된 값은 조사되거나 갱신되지 않는다. 또, 이 함수는 초기 n, r이 $0 \le r \le n$을 만족하면 호출되는 모든 `binom`도 해당 조건을 만족한다.

상헌이는 이항 계수를 여러 번 질문할 때마다, 몇 개의 이항 계수들이 `dp` 배열에 새롭게 계산되어 갱신되는지 알고 싶어한다. 이항 계수 문제는 아니지만 한 번 풀어보자.

입력 형식

첫 번째 줄에는 `binom` 함수의 호출 횟수인 정수 q가 주어진다. $(1 \le q \le 80,000)$

이후 q개의 줄에 걸쳐 두 정수 n, r이 공백으로 구분되어 주어진다. $(0 \le r \le n \le 80,000)$ 이는 `binom(n, r)`을 호출한다는 뜻이다.

출력 형식

각 줄마다 `binom(n, r)`을 호출했을 때 갱신되는 `dp` 배열의 칸 개수를 출력한다. 매 `binom` 함수의 결과로 갱신된 `dp`는 그 이후로도 계속 적용된다.

예제

표준 입력	표준 출력
4 2 1 3 1 6 3 3 2	4 2 10 0
3 4 2 5 4 6 3	9 4 5

설명

첫 번째 예제에 대한 설명은 다음과 같다.

- `binom(2, 1)`은 `(2,1)`, `(1,0)`, `(0,0)`, `(1,1)`을 갱신한다.
- `binom(3, 1)`은 `(3,1)`, `(2,0)`을 갱신한다.
- `binom(6, 3)`은 `(6,3)`, `(5,2)`, `(4,1)`, `(3,0)`, `(4,2)`, `(3,2)`, `(2,2)`, `(5,3)`, `(4,3)`, `(3,3)`이 갱신한다.
- `binom(3, 2)`은 어느 칸도 갱신하지 않는다. 3번째 호출에서 `(3,2)`가 갱신되었기 때문이다.