


UCLA-ve Local Programming Contest 2011
Warm-Up Session
Universidad Centroccidental "Lisandro Alvarado"

Sponsored by:

July 09, 2011

This problem set should contain a rules section and two (02) problems on three (03) numbered pages.
Please inform the Contest Staff immediately if something is missing from your problem set.
Welcome to the Programming Contest and enjoy it!.
http://uclave.acm.org/

Rules

- Each team will be provided with a single computer that have the VM of ACM-ICPC running on VirtualBox. All teams have equivalent computing equipment.
- There are two (02) problems for each team to be completed in one (01) hour.
- All problems require that you read test data from the *standard input* and write results to the *standard output.*
- You have may use any of the following programming languages: C, C++ or JAVA. You may use different programming languages for different problems and even for different submissions of a problem.
- All problems have a source file named with the first word of title of the problem. The extension you must use for your source files are: *.c* for programs written in C, *.cpp or .cc* for those in C++, and *.java* for those in JAVA.
- You may use any of the standard libraries that your chosen programming languages provides. You may not use any other library that requires an extra flag to be passed to the compiler command. If you do this, judges will probably get compilation-linking error in your program.
- Output correspond exactly to the provided sample output format, including (mis)spelling and spacing. Multiple spaces will not be used in any of the judge's output, except where explicitly stated.
- Your solution to any problem should be submitted for judging using PC^2 software only. Once you have submitted a solution, it will reach the judges. The time it takes for your problem to be judged will depend on how busy the judges are. Once your submission has been judged, you will receive a message through PC^2 indicating the judgment. If your solution is accepted the message will be "Yes", if it is not then the message will be "No", along with the type of error the judges encountered, which may be: "Compilation Error", "Runtime Error", "Time-Limit Exceeded", "Wrong Answer" or "Presentation Error".
- Programming style is not considered in this contest. The judges will only test whether the *input / output* behavior of your program is correct or not. However, each problem has an execution time-limit of 60 seconds. That is, if your program takes more that 60 seconds to execute for the given input, it will be judged as incorrect.
- Contestants may bring any printed materials *(books, papers, documentation, source code of programs, etc.)* to the contest area, but no soft copy will be allowed *(diskettes, CD's, DVD's, pendrives, etc).*

Problem A. Old School

Input file: `Standard Input`
Output file: `Standard Output`

A particle has initial velocity and constant acceleration. If its velocity after certain time is v then what will its displacement be in twice of that time?

Input

The input will contain two integers in each line. Each line makes one set of input. These two integers denote the value of $v(-100 \leq v \leq 100)$ and $t(0 \leq t \leq 200)$ (t means at the time the particle gains that velocity)

Output

For each line of input print a single integer in one line denoting the displacement in double of that time.

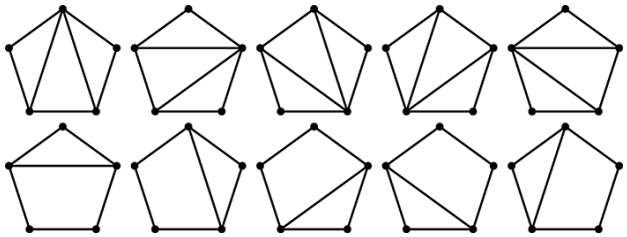
Sample input and output

Standard Input	Standard Output
1 2 5 6 -10 8	4 60 -160

Problem B. Geometric Division

Input file: `Standard Input`
Output file: `Standard Output`

Given a geometric figure like a regular polygon, there are numerous ways to divide it into several triangles and/or quadrangles by adding some diagonals that do not properly intersect each other. For example, the figure as shown below, all ten different divisions of a regular pentagon into triangles and quadrangles.



Divisions of a regular pentagon into triangles and quadrangles.

Given n, the number of sides of the polygon, compute the number of such divisions.

Input

The input contains multiple test cases. Each test case consists of a single integer $n(3 \leq n \leq 5000)$ on a separate line. The input ends where EOF is met.

Output

For each test case, print the answer modulo 2^{64} on a separate line.

Sample input and output

Standard Input	Standard Output
3	1
4	3
5	10
6	38
7	154
8	654
9	2871
10	12925