Problem 1. Go to School

(Time Limit: 2 seconds)

Problem Description

Taipei is a busy city. There are many cars on the road in the morning. Many people arrive at schools or companies late because of the traffic jam. Kenny is a lazy high school student and he feels he is spending too much time to get to the school. So he surveyed the traffic status and found that the time to pass each road will be longer when it is later in the morning, because the cars and motors will become more. Now Kenny wants to schedule a best route to travel from home to school. According to Kenny's information, please help Kenny to find the minimum time he needs for the travelling.

Input Format

There is an integer t ($1 \leq t \leq 20$) in the first line, which represents the number of cases to follow . For each case, the first line contains two integers n ($1 \leq n \leq 1000$) and m ($1 \leq \mathrm{m} \leq 10^6$) which indicate there are n+1 road junctions and m roads on Kenny's map. In the next m lines, each contains five integers s, t, c, l, o($0 \leq s, t \leq n$, $0 < c \leq 100$, $0 \leq l$, $o \leq 100$) which indicate that there is a road from junction s to junction t , and before Kenny travels on it , the length of the road will increase l units after each c seconds , and the initial road length is o units . For simplicity, we assume that the length of a road will not change once Kenny has travelled on it. Kenny's home is at junction 0, his school is at junction n , and it takes 1 second for Kenny to travel 1 unit.

Output Format

For each case, output a line the minimum time that Kenny needs to travel from home to school.

Example

Sample Input:	**Sample Output:**
1	4
3 3	
0 1 1 0 2	
1 2 1 0 1	
2 3 1 0 1	

Problem 2. Parsing Tree

(Time Limit: 3 seconds)

Problem Description

In the study of formal logic, the syntax of a logical language is typically defined by its alphabet and formation rules of its formulas. Let us consider a very simple logical language such that its alphabet is comprised of a countable set of propositional symbols Φ and a logical connective $\cup$, and its formulas φ are formed according to the following syntactic rules:

$$\varphi ::= p \mid \varphi \cup \varphi$$

where $p \in \Phi$. Without using auxiliary symbols like parentheses as usual, a formula may have different readings. Each reading can be represented by a parsing tree, which is a rooted binary tree whose leaves are labeled with elements in Φ and internal nodes are labeled with the symbol $\cup$. If the in-order traversal of a parsing tree is a formula φ, then we say that it is a parsing tree for φ. For example, the following tree is a parsing tree for the formula $p \cup q \cup r$.

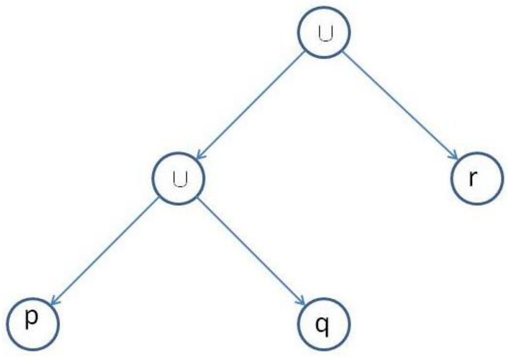

Figure 1. A parsing tree for $p \cup q \cup r$

For each $n \geq 0$, let φ_n denote the formula $p_0 \cup p_1 \cup \ldots \cup p_n$,, where each p_i is a propositional symbol and $p_i \neq p_j$ for any $0 \leq i \neq j \leq n$. The problem is to find the number of different parsing trees for φ_n.

Technical Specification

The index *n* of the formula φ_n is an integer *n* such that $0 \leq n \leq 32$.

Input Format

The first line of the input consists of a single number *k* denoting the number of test cases, followed by *k* lines where each line contains the input *n* of a test case.

Output Format

For each test case with input *n*, compute the number of different parsing trees for φ_n. Output the answers one by one in *k* lines.

Example

Sample Input:	**Sample Output:**
2 2 3	2 5

Problem 3. Permutation Rank

(Time Limit: 3 seconds)

Problem Description

Suppose you are given a word in English, say `app`. You can scramble letters in this word so that you get all permutations of it. There are totally 3 kinds of distinct permutations in this case. They are `app`, `pap` and `ppa`. When you list these words in the lexicographic order, you are required to sort them as if they do appear in a dictionary. That is, words beginning with `a` are always in the front of words beginning with `b`, `c`, etc. This rule also applies to the second positions and so on. After sorting them, you can count the number of permutations before a specific word in this list; this is called its permutation rank. For example, the permutation ranks of `app`, `pap` and `ppa` are 0, 1 and 2, respectively. Your mission is to determine the permutation ranks for a bunch of words written in lowercase English alphabet.

Technical Specification

- The alphabet of input words is the set of lowercase English letters.
- The length of a word in the input is at least 1 but no more than 20 letters.

Input Format

The first line of the input file contains an integer indicating the number of test cases. Each test case has a single line which specifies the word that you need to evaluate its permutation rank.

Output Format

For each test case, please output its permutation rank in a line.

Example

Sample Input:	**Sample Output:**
5	0
abc	330
success	1475
failure	0
aaaaaaaaaaaaaaaaaaaa	2432902008176639999

tsrqponmlkjihgfedcba	

Problem 4. Squares

(Time Limit: 2 seconds)

Problem Description

MZ has given too many problems too hard to be solved by many teams. This time, he decides to give a simpler problem about counting the number of squares whose vertices are all in a specified set *S*. A square has four sides of the same length and four right angles (90 degrees). To keep the problem simple, we only consider some grid points, i.e., the coordinates are all integers. In this problem, the set S is specified by three positive integers *n*, *x*, *y*.

$S=\{(p,q): p, q\in\{1,\ldots,n\} \wedge (p,q)\neq(x,y)\}$

Please write an efficient program to compute the number the squares whose vertices are all in *S*. Please note that a square does not necessarily consist of horizontal edges and vertical edges. For example, a square may have (1,2), (2,1), (3,2), (2,3) as its four vertices.

Technical Specification

- The number of test cases: $T \leq 100$
- $1 < n \leq 1000$
- $0 < x \leq n$
- $0 < y \leq n$
- The answer might be greater than 2^{32}.

Input Format

The first line of the input contains an integer *T* indicating the number of test cases. Each test case is a line consisting of three integers *n*, *x*, *y* separated by blanks.

Output Format

For each test case, output the number of squares whose corners are all in *S* in a line.

Example

Sample Input:	**Sample Output:**

```
6
3 1 1
3 1 2
3 2 2
4 1 1
4 1 2
4 2 2
```
	```
	4
	3
	2
	17
	15
	13
	```

Problem 5. Direction Killer

(Time Limit: 4 seconds)

Problem Description

You are a Lv.5 esper "Direction Killer" in Academic City. This is a very powerful ability, however, it does not help you from reaching the grocery stores. As your name suggests, you have no idea which direction is which. So, unfortunately, any effort of your trying to get to the grocery store will be as if you are performing a random walk.

Luckily, as a Lv.5 esper, you are unrivaled in your calculation prowess. Although it does not help you to get to where you want, it does help you estimate how long it will take for you to get there and decide if you'll have time to catch the last episode of your favorite anime.

You have a map of the city which is a rectangle space with N-by-M blocks. Since your ability allows you to traverse any landscape and go through all obstacles, you may just ignore them. For each step, you may move to a block adjacent to you in the north, south, east or west direction with equal probability assuming they exist. That is, you will move to any adjacent block with probability one half, one-third, and one-quarter if you are on the corner blocks, on the edge blocks, and on the other blocks, respectively. Note that sometimes the grocery store is exactly where you started. In that case, since you always didn't realize it until you start moving, you'll have to wait until the first time you return to the starting position. Calculate with your incredible esper prowess, the expected number of steps it takes to get to the grocery store. Hey, even espers need food to survive!

Technical Specification

- The number of test cases $T \leq 50$
- $0 < N \leq 20$ and $0 < M \leq 20$
- The coordinates of the starting location: $0 < x_1 \leq N$ and $0 < y_1 \leq M$
- The coordinates of the grocery store: $0 < x_2 \leq N$ and $0 < y_2 \leq M$

Input Format

The input starts with an integer T, where T represents the number of test cases. Each test case will start with two integers N and M, representing the length and width of the Academic City.

It is followed by two integers x1 and y1 on a line, representing your starting location, and then followed by 2 more integers x2 and y2 on another line representing the location of the grocery store.

Output Format

Output the expected value of the number of steps it takes to get to the grocery store on a single line. Round the output number to 2 digit places.

Example

Sample Input:	**Sample Output:**
`4`	`4.00`
`2 2`	`1.00`
`1 2`	`3.00`
`2 1`	`15.00`
`1 2`	
`1 1`	
`1 2`	
`1 3`	
`1 2`	
`1 1`	
`3 3`	
`2 2`	
`1 1`	