Function call requires implicit overhead when producing the stack frame with copying arguments to parameters. To overcome this issue, we can use inline function to reduce the number of function calls, or use reference parameters to prevent the copy. In addition, we may invoke a function repeatedly with the same argument value for a particular parameter. That is, we can specify each parameter a default argument: a default value to be passed to the corresponding parameter. Whenever a program omits an argument, the compiler rewrites the function call and inserts the default value of that argument to be passed as an argument in the function call.

Your goal is to write a program with a functions containing all of the above features. Specifically, give the following functions:

1. An inline function to calculate the 3D offset $\vec{x}$ in terms of an initial speed $\vec{v}_0$ and an acceleration $\vec{a}$ along with x, y and z coordinates in a given period of time according to the following equation:

$\vec{x} = \vec{v}_0 t + \frac{1}{2}\vec{a}t^2$**,**

where the default values for the initial velocity, acceleration, and time are (1,2,3), (2,5,10), and , 5 respectively.

2. A call-by-reference input function to retrieve each line of the input.

3. A call-by-reference output function to call the function in 1. for different number of input parameters.

Requirement: Use the sample main function and implement the above three functions in two separate files “function_header.h” and “function_header.cpp”. Use C++-style input and output.

Prohibited: Use multiple functions for different numbers of parameters. Use C-style input/output.

Input

The first line contains an integer N, representing the number of cases. In the following N lines, each line contains an integer n, representing the number of parameters to be inputted, followed by n integers which in turn correspond to (if exist) the time, the velocity along with x coordinate, the acceleration along with x coordinate, the velocity along with y coordinate, the acceleration along with y coordinate, the velocity along with z coordinate, and the acceleration along with z coordinate.

Output

For each line of input, output the 3D offsets with end line stream manipulator. All the calculation must be completed using **integer operations**.

Sample Input

```
7
1 4
2 4 5
3 4 5 8
4 4 5 8 3
5 4 5 8 3 7
6 4 5 8 3 7 6
7 4 5 8 3 7 6 4
```

Sample Output

```
20 48 92
36 48 92
84 48 92
84 52 92
84 68 92
84 68 104
84 68 56
```