We hope you all enjoyed the problem statements. Since this was our(npsabari and karthikabinav) first time setting problem statements we did have some issues with typos in testcases which led to rejudging. We apologize for that. We will try to give a brief idea about solutions we had in mind while setting the problems.

1. GuruP and his sweetheart(APOWB)

Solution: This question boils to finding, if a number n can be expressed as a^b, where a and b are integers greater than equal to 2.
Two observations were required.

- Given a value b, we can check if there exits a integer a such that $a^b = n$ in $\log(n)$ time using a binary search
- The number of value of b is bounded by $O(\log(n))$. Hence an exhaustive search over all b's can be performed in $O(\log n)$ time.

One must be careful while performing the binary search within the correct limits for a given b to avoid overflow.

Time Complexity: $O(t * \log^3(n))$.

2. Awesome Threesome (AWTHR)

Solution: Cutting out all the story the problem was to find the following. Given a graph, partition into three sets(possibly empty) A, B, C such that A and C were complete sub graphs, while B had no internal edges. Also, for a given vertex in set B, it should have edges to atmost one of the sets A or C.

The following was a modified version of the Chocolates Division problem from Kanpur regionals - 2013.
Remove all isolated vertices. Now check if the graph has atmost 2 components. If not, output NO. If yes, on each component use a similar approach to that above mentioned problem.

A brief idea for the above mentioned problem.
Though, there were solutions which used strongly connected components, a simpler degree based analysis can do the job. Lets call Q as the set with complete subgraph, and T as the set with no internal edges. Any vertex in set T will have degree no

greater than any vertex in Q(easy to see, if not try proving). Hence, just sort the degrees and there are n ways to split the vertices into Q and T. Check if any of them satisfies the required property.

Time Complexity: $O(t * (n^2 \log n))$

3. Night at library (EIGVAL)

Solution: This question had a graph as input and a matrix corresponding to it defined. The problem was to check if the eigen values of the matrix had the property that the largest - the second smallest was atleast 2.

The expected solution doesnt have anything to do with eigen value decomposition algorithm(!). This matrix is well known in spectral graph theory as the normalized laplacian matrix.

Let $\lambda_0 \leq \lambda_1 \leq .. \leq \lambda_{n-1}$ be the n eigen values of the matrix. The following property holds true, $\lambda_1 = 0 \leftrightarrow$ the graph is disconnected. Also, $\lambda_{n-1} \leq 2$ for all graphs.
$\lambda_{n-1} = 2 \leftrightarrow$ there exists atleast one non-trivial bipartite component. Non-trivial implies there exists atleast one edge in the component. Therefore, for graphs which are disconnected and has atleast one non-trivial bipartite component the required difference is atleast 2.

We will let the reader explore the proofs for these above statements. As a reference to topics in spectral graph theory we suggest the monograph Lx=b.

Time Complexity: $O(t * (m + n))$

4. Dreamer's E-trip (GRETHIEF)

Solution:

5. New York Magnets (SHAMASS)

Solution: This was one of the simpler problems. The question was given N bodies starting at various times, the time it takes to reach distance d. The only catch being if it comes next to a body which went before it, it will have the speed of that body. Also, every body has a maximum speed it can reach.

This can be solved using some basic equations of motion.
Note that, the time for a particular object to reach destination is maximum of time for it to reach based on equations of motion and time taken by the object that left before it.

To calculate the time based on equation of motion, there can be two cases:

- The object didint reach its maximum velocity. In this case, time is $\sqrt(\frac{2d}{a})$
- The object reached its maximum velocity mid way. In this case find the distance when it reached its maximum velocity and the time from there till destination using this constant maximum velocity. (use motion equations)

Time Complexity: $O(t * n)$

6. Art of Throwing (SHASFUN)

Solution:

7. Curious Case of vatsan mama(SHASGAME)

Solution:

8. Good boy eshwar (SHASGRA)

Solution:

9. No strings attached(SHASTR)

Solution: The question involved to find given two strings, find the length of the maximum common subsequence such that k substring of both s1 and s2 is removed.

The question involved using dynamic programming approach to precompute longest common subsequences and iterate through all possible substrings that can be removed from string1 and string 2.(There were multiple approaches all of which used some form of dynamic programming).

Define a dp[string1_size][string2_size][2][2] matrix to precompute for all possible lengths of string1 and string2 and directions of both forward and backward(the third and fourth dimension of this matrix). Using this we can query for a given substring of sring1 and string2 removed in constant time.

Time Complexity: $O(t * (n^2 + n^2))$.