CONTEST STRATEGY

REGIONALS:

1- Sort problem set by length and assign to members starting with the fastest

2- Read the problem CAREFULLY, if it is an ACE code it directly on PC and go to step 8

3- Give yourself 5 minutes of thinking even if the problem is hard, you only need to understand the problem statement very well and think in a solution if possible

4- Describe the problem to the person who is better at the problem area, whom should listen very carefully and make sure he understands the problem very well,

5- This small meeting should decide one of the following: 1-the problem should be delayed 2-you should write the solution you came up with 3-you both stay for sometime thinking in a solution 4-you only should stay for sometime thinking in a solution

6- If you both decided that this problem is to be solved, the better of the two at the problem area will read the problem carefully (if not yet) write code on paper and get approval from the backup that the code is COMPLETE

7- Once the PC is free, copy ur code there, make sure you copy the input correctly from the problem statement and debug for the first 10 minutes to match the sample output, if more debugging is needed the backup should join for another 10 minutes, if still print and debug on paper

8- if you submit and got WA or TLE or RTA review the checklist, read the problem again, debug on paper or whatever for another 20 minutes, if u found bug(s)interrupt the man on the PC, write the testcase u suspect, run and make sure u get WA, then take backup of the code apply ur fix, run and make sure the output is correct and submit

9- if the offline debug took 20 minutes the backup should read the problem and the code and spend 10 minutes with you, if u couldn't get it leave the problem immediately and get back to it later

10- In the last hour don't start a new problem (unless you've no wrong submissions), sort problems by most solved and for each wrong submission the author and the backup (and the third if he isn't doing anything) should debug it.

HINTS FOR THE CONTEST

Hints: -Compete with problemset instead of team, use score board only to know which problems are solved

WA bugs:

-CHECK THE SPELLING OF OUTPUT STRINGS (Specillay s for plural and case sensitivity)

-Repeat sample input cases in reverse order

-Read the problem again, specially the input and output

-Make sure you correctly initialize between testcases

-Math operations like mod, floor and ceil works differentelly on positive and negative

-Multiple edges between two nodes -Multiple spaces between input words -truncate or apporximate

-double issues, watch for -0.0 (if the double is near than zero output zero) and don't use (==, <, >) directly

-Multiple input items (same string in the input twice), use set or multiset

-Input terminating condition and output format must equal to what the problem specified

-Copy input correctly from problem statements

-watch for special cases in the input

-Integer and char overflow (multiplications & powers& Cross Products)!!

-Make use you don't use a very large infinity and add things to it which may cause overflow

-If you've a double and want to convert it to integer (multiply by 100000 or so), then add EPS first as the double 0.7 may be stored as. Watch out: "Input is a 32 integer bit" int x; cin>>x; if(x<0) x = -x; do(x); OVERFLOW: -2^31 should not be positived in int var.

0.69999999

-HashSet and HashMap don't sort, TreeSet and TreeMap do (C++ set and map are tree-based) -If the problem can be DPed then do it this way (safer than greedy)

-After all, you may have got the problem the wrong way, let a fresh member read it and hear from him (don't affect him)

-not a number(NAN) which comes from sqrt(-ve), (0/0) ,or acos(1.00000000001) or cos(-1.000000000001) for such case if the value is very close

to -1 or 1 make it 1.

- reading by scanf("%d ",x) to remove '\n' can remove leading spaces on the next line

---

TLE bugs: -Note that Choosing all combination of N items is of order 2^N using recursion and N*(2^N) if using bitmasked loop -Use scanf instead of cin if u got TLE -avoid division, mod and multiplicatio operations if u got TLE -If the problem is DP, make sure you are using the smallest possible number of dimensions for the DP -incorrect input reading/termination (watch for empty lines)

Runtime bugs: -Index out of boundaries -Stack over flow -integer division by zero -Calling Integer.pareseInt with invalid string- incorrect input reading (getline)- empty lines in input

Presentataion error=Output formmat error: 1) Watch out diplayed lists 1 3 7 9 Do not display SPACE after last number(here 9)

2) Make sure from sepreating testscases. 2.1) Display blank line after each test case means there is a line between each test case even after the last test case. 2.2) Display blank line between test casse. --> Means ONLY between testcases

3) In C++ : memcpy and memset don't work normally with very large arrays

---

1- first hour is the hunt for ACES, don't interrupt the team members too much in this hour. if there is an interruption it should be for asking about something not for thinking with you in the idea.

2- the ACE problem is the addition, multiplication or sorting problem such that it's not harder than Div2-250 or the lines of code doesn't exceed 20 lines. **it's a must that the problem doesn't need the strategy and the problem can be solved inside the main**.

3- read the problem statement till the end, take your time to check the input and the output, and take care that the sample input and output may have the key to the problem solution.

4- make your code small, simple, smart

---

contest scenario: In the first hour do the following:

1- no interrupts, hunting for aces, and reading problems as much as you can.

2- read the problems to the end including the input and the output section and put a rough estimate for the problem.

3- never not to complete reading a problem to the end.

starting from second hour:

1- all problems codes must be written on papers.

2- the written code should be written in a clean way.

3- the code should be scanned from the papers to the machine and compile.

starting from the third hour:

1- the score board is a good guide to see which problems you should solve.

2- schedule for the next 2 hours which problems to start with and which to delay.

in the last hour:

1- do not start coding a problem in the last hour unless you got accepted in all the other tried problems.

2- do your best to solve all the written problems.

«

WHY WRONG ANSWER

-CHECK THE SPELLING OF OUTPUT STRINGS (Specially s for plural and case sensitivity)

-Repeat sample input cases in reverse order

-Compete with problem set instead of team, use score board only to know which problems are solved

-Read the problem again, specially the input and output

-Make sure you correctly initialize between test cases

-Multiple edges between two nodes

-Multiple spaces between input words

-truncate or approximate

-double issues, watch for -0.0 (if the double is near than zero output zero) and don't use (==, <, >) directly

-Multiple input items (same string in the input twice), use set or multiset

-Input terminating condition and output format must equal to what the problem specified

-Copy input correctly from problem statements

-Watch for special cases in the input

-Integer and char overflow!!

-Make sure you don't use a very large infinity and add things to it which may cause overflow (E.g. in DP)

-Small infinity may be wrong (if it smaller than what u calc)

-overflow: multiplications(cross product) & powers & Base conversions & DP counting problems.

-Check CAREFULLY input stopping conditions. E.g. Input terminate with line START with # or CONTAINS #

-If you've a double and want to convert it to integer (multiply by 100000 or so), then add EPS first as the double 0.7 may be stored as 0.69999999

-HashSet and HashMap don't sort, TreeSet and TreeMap do (C++ set and map are tree-based)

-If the problem can be DPed then do it this way (safer than greedy)

-After all, you may have got the problem the wrong way, let a fresh member read it and hear from him (don't affect him)

-not a number(NAN) which comes from sqrt(-ve), (0/0) ,or cos(1.00000000001) or cos(-1.000000000001) for such case if the value is very close to -1 or 1 make it 1.

-make sure when u are flooring a -ve integer that u floor it to the nearest less integer, for example Floor(-2.3) = -3 but Floor(2.3)= 2.

- Other tricks:

- Word is "sequence of upper/lower case letters". then ali is 1 word, X-Ray is 2 words

- You will operate on string of letters (this do not mean Latin letters a-z, this is bigger)

- Given 2 integers i, j, find number of primes between them. input may be 10 20 OR 20 10

- Given N*M grid, Read N lines each start with M chars. E.g. 3*2

-- 1st line -> ab

-- 2nd line -> cdEXTRA // use to depend on read N, M, as RE may happen

-- 3rd line -> ef

- In multiset insert add new element, but delete removes ALL instances of element

-- if multiset contains (3 3 3 3 6 9) and u delete 3 -->will be (6, 9)

- Use to read input then process it, if u did not, do not BREAK wrongly while reading.

--lp(i, 5) { cin>>x; if(!valid(x)) { ok = 0; break;} --> What about output REMINDER?

- Geometry: Is polygon simple, convex, concave? Is there duplicate points? Does it matter?

- if you are using double the maximum eps you can use is 1e-11, if you need more precision you have to use long double instead.

- if the output is longlong make sure that you use cout not printf