V3

In my opinion, I don't think we can save our self. I my self struggle to avoid all things that I consider 'sinful' 24/7 (24 hours a day, 7 days a week). How not to be angry, lazy, jealous, lusting, 'stealing', not caring, name all the bad things here... ALL the time. I think my good deeds will not outweighs my bad ones. We all, you and me, are sinners... It is just a matter whether we want to admit it or not... The Christian Bible said that "all have sinned and fall short of the glory of God"... To be continued in volume 4. See previous story in volume 2.

Last updated on: 15 October 2007 08:12:38 PM

Comment on this volume: None

No	Problem Name	*	Algorithm
300-307: Central European Regionals - 1995 (2nd link)			
300	Maya Calendar	6.0	Ad Hoc
301	Transportation	*	WA
302	John's trip	*	Haven't try yet
303	Pipe	*	Haven't try yet
304	Department	*	Haven't try yet
305	Joseph	5.5	Simulation
306	Cipher	5.0	Simulation
307	Sticks	9.9	TLE
308-315: Central European Regionals - 1996 (2nd link)			
308	Tin Cutter	*	Haven't try yet
309	FORCAL	*	Haven't try yet
310	L--system	*	Haven't try yet
311	Packets	7.0	Math
312	Crosswords (II)	5.5	Output-related
313	Intervals	*	Haven't try yet
314	Robot	*	Haven't try yet
315	Network	6.5	Graph (Articulation Point)
316-323: Southwestern European Regionals - 1996			
316	Stars	*	Haven't try yet
317	Hexagon	*	Haven't try yet
318	Domino Effect	*	Haven't try yet
319	Pendulum	*	Haven't try yet
320	Borders	6.5	Output-related
321	The New Villa	*	Haven't try yet, look at Arif's notes
322	Ships	*	Haven't try yet
323	Jury Compromise	*	Haven't try yet
324-330: North Central Regionals - 1993			
324	Factorial Frequencies	5.5	Math (Big Integer + Factorial)
325	Identifying Legal Pascal Real Constant	4.5	Ad Hoc
326	Extrapolation Using a Difference Table	6.5	Math
327	Evaluating Simple C Expressions	4.5	Simulation
328	The Finite State Text Processing Machine	*	Haven't try yet
329	PostScript Emulation	*	Very tedious simulation, Haven't try yet
330	Inventory Maintenance	6.0	Ad Hoc, output-related
331-337: North Central Regionals - 1994			
331	Mapping the Swaps	4.5	Ad Hoc
332	Rational Numbers from Repeating Fraction	4.5	Math
333	Recognizing Good ISBNs	3.5	Simulation
334	Identifying Concurrent Events	5.5	Floyd Warshall (Transitive Hull)
335	Processing MX Records	5.0	Simulation
336	A Node Too Far	5.0	Graph Traversal
337	Interpreting Control Sequences	5.5	Output-related
338-344: North Central Regionals - 1995 (2nd link)			

338	Long Multiplication	7.0	WA, what's wrong ??
339	SameGame Simulation	4.5	Simulation
340	Master-Mind Hints	3.5	Simulation
341	Non-Stop Travel	4.0	Floyd Warshall
342	HTML Syntax Checking	*	Cannot be judged yet!!!
343	What Base Is This?	5.5	Math (Base Number)
344	Roman Digititis	4.0	Math (Roman Number)
345-351: North Central Regionals - 1996			
345	It's Ir-Resist-Able!	*	Haven't try yet
346	Getting Chorded	*	Ad Hoc, Haven't try yet
347	Run, Run, Runaround Numbers	4.5	Simulation
348	Optimal Array Multiplication Sequence	5.5	DP (MCM)
349	Transferable Voting (II)	5.5	Simulation
350	Pseudo-Random Numbers	4.0	Math
351	``Cheapest'' Scores	*	Cannot be judged yet!!!
352-376: Source Unknown			
352	The Seasonal War	4.0	Graph (Flood Fill)
353	Pesky Palindromes	4.0	Ad Hoc
354	Crazy Calculator	*	Cannot be judged yet!!!
355	The Bases Are Loaded	5.5	Math (Base Number)
356	Square Pegs And Round Holes	3.0	Math (Geometry)
357	Let Me Count The Ways	5.5	DP (CC) + Math (Big Integer)
358	Don't Have A Cow, Dude	*	Haven't try yet
359	Sex Assignments And Breeding Experiments	*	Haven't try yet
360	Don't Get Hives From This One!	*	Cannot be judged yet!!!
361	Cops and Robbers	*	Haven't try yet
362	18,000 Seconds Remaining	3.0	Simulation
363	Approximate Matches	*	Cannot be judged yet!!!
364	Constitutional Computing	*	Cannot be judged yet!!!
365	Welfare Reform	*	Haven't try yet
366	Cutting Up	*	Haven't try yet
367	Halting Factor Replacement Systems	*	Haven't try yet
368	Indexing Web Pages	*	Cannot be judged yet!!!
369	Combinations	5.0	Math
370	Bingo	*	Haven't try yet
371	Ackermann Functions	3.0	Math (3n+1)
372	WhatFix Notation	*	Cannot be judged yet!!!
373	Romulan Spelling	*	Haven't try yet
374	Big Mod	4.0	Math (Modulo)
375	Inscribed Circles and Isosceles Triangles	*	Haven't try yet, simple math problem ??
376	More Triangles ... THE AMBIGUOUS CASE	*	Haven't try yet, math problem.. difficult?
377-384: Mid-Atlantic Regionals - 1996			
377	Cowculations	5.0	Math (Base Number)
378	Intersecting Lines	5.0	Math (Geometry)
379	HI-Q	4.0	Simulation
380	Call Forwarding	5.0	Backtracking
381	Making the Grade	6.5	Simulation
382	Perfection	2.5	Math
383	Shipping Routes	6.5	Graph Traversal
384	Slurpys	4.5	Backtracking
385-391: Mid-Central Regionals - 1995			
385	DNA Translation	5.5	String processing
386	Perfect Cubes	4.0	Math
387	A Puzzling Problem	*	Haven't try yet
388	Galactic Import	*	Haven't try yet
389	Basically Speaking	5.0	Math (Base Number)
390	Letter Sequence Analysis	*	Haven't try yet
391	Mark-up	4.5	Output-related
393-391: Mid-Central Regionals - 1996			

392	Polynomial Showdown	5.5	Output-related
393	The Doors	*	Haven't try yet
394	Mapmaker	4.0	Ad Hoc
395	Board Silly	*	Haven't try yet
396	Top Dog	*	Haven't try yet
397	Equation Elation	4.5	Ad Hoc
398	18-Wheeler Caravans (aka Semigroups)	*	Haven't try yet
399	Another Puzzling Problem	*	Haven't try yet

Total submit-able problems in this volume: 100
Solved problems: 46
Problems in Wrong Answer list from this volume: 9
Unattempted problems: 45
Total hints in this volume: 51

---

300 - Maya Calendar

Very tedious... I suggest that you don't try this problem unless you have time and patience... This problem is basically a follow-the-given-rule problem.

305 - Joseph (by: Nuno Santos)

A simulation problem... Using pre-calculation, the answers are:
0,2,7,5,30,169,441,1872,7632,1740,93313,459901,1358657,2504881

Some hints from Nuno:

My first approach was also very slow, it took over 5 minutes to find the solution for k=10. My problem was that in order to find the next person that should die, I was iterating m times over a linked list with the live person, with m being the size of the "jump". For this particular problem m gets very big, thereby making this algorithm very slow.

My solution consisted on evaluating the next person to die using modular arithmetic. This can be done if we take in consideration the number of persons that are still alive. So suppose there are n persons and we just killed the person at position p, leaving only 1 persons alive. We must also consider the rank (R) of that person, which is its position among the person alive. So the next person to die has rank Rn = (R + (m-1) % l). After evaluating this value, just find the Rn-th alive person and kill it.

306 - Cipher

The key idea to avoid the time limit is not to simulate everything... it will be too slow... For each index i, there will be a cycle because the array a will be just a permutation of 1 to n, no duplicates... so every k%cycle[i], you will produce the same character in that position. Utilize this property to speed up your simulation a lot...

312 - Crosswords (II)

Easy problem, but very hard in formatting output. Single simple mistake can will cause wrong answer, this problem is really output-related.

Common Mistake:
1. Forget not to print black squares if they are given at the edges... Black squares at the edges should be removed from the diagram!!!
2. Forget to delete unnecessary spaces at the end of the line.
3. Forget that numbering must be in "nnn" 3-digit format.
4. Other output error.

315 - Networks

This problem is a well known graph problem, finding "articulation points", or finding the weakest link. Search internet for that term and apply the algorithm.

321 - The New Villa (by: Md. Arifuzzaman)

Use a graph of 10*(2^10) = 10240 nodes where, 10*(2^10) = (your room position)*(which lights are on and which are off), then implement a BFS on this graph.

324 - Factorial Frequencies

Max n = 366... 366! will be... very big :), and somehow you need to count the occurence of '0', '1', ... and so on up to '9'... So, check your Big Integer library. Do the factorial multiplication, and then count the digits.

325 - Identifying Legal Pascal Real Constant

Checking for legality. It's very straightforward, just follow the problem specification. If you can pass the sample input => 99% Accepted

327 - Evaluating Simple C Expressions

Huff, even though this problem is not difficult to understand, the actual implementation is rather confusing...

Common Mistake:
1. Input can be like this: "a ++" or "++ a", with spaces beside unary/binary operators.
2. Don't increment/decrement post inc/post dec values first, you'll get troubled because of this. Example: "a - b++" => should be -1, a=1, b=3.
3. Read until you encounter END OF LINE!!!, they didn't say how long the line will be...

330 - Inventory Maintenance (with help from: Syukri)

This problem will be very annoying for most people. You must read the problem description very very carefully. You must not forget even a single space for this problem sensitive output format, be very very patient.

331 - Mapping the Swaps

Enumerate all possible permutation of the input array, but you only allowed to do n swaps, when n is already counted using minimum bubble sort swaps (the problem only want the minimum swap for the swap map), check if this array is "sorted" at the end, if yes then we have one swap maps, accumulate it all, and this total is all possible swap maps.

Yes this method is a total brute force, however, since I manage to get accepted using this technique, I can safely assume that the given test data does not contain big test cases :)

332 - Rational Number from Repeating Fraction

You already given the formula in the problem statement... use it, and then just before output, make sure you reduce the numerator and denominator to the smallest form by using Greatest Common Divisor (GCD).

333 - Recognizing Good ISBNs (test case by: Mazharul Islam)

The internal algorithm for this problem is very easy... what makes this problem seems so difficult (can be seen by low acceptance rate) is the strict input-output... So, instead of discussing the algorithm, I'll clarify the I/O...

Quote from problem description (input): "The input data for this problem will contain one candidate ISBN per line of input, perhaps preceded and/or followed by additional spaces. No line will contain more than 80 characters, but the candidate ISBN may contain illegal characters, and may be more or fewer than the required 10 digits. Valid ISBNs may include hyphens at arbitrary locations. The end of file marks the end of the input data."

Quote from problem description (output): "The output should include a display of the candidate ISBN and a statement of whether it is legal or illegal".

Let's examine the sample input: 0-13-162959-X (Tanenbaum's Computer Networks)

1. One candidate ISBN per line of input (max 80 chars).

0-13-162959-X 0-13-162959-X
0-13-162959-X 0-13-162959-X is incorrect.

2. preceded and/or followed by additional spaces.

0-13-162959-X ==> there is a trailing spaces here
0-13-162959-X is correct.

3. may contain illegal characters.

0abc-13dfasdfa-162959-X
0abc-13dfasdfa-162959-X is correct.
0 -13-162959-X ==> according to my accepted solution, space inside is okay
0-13-162959-X is correct.

4. More or fewer than the required 10 digits.

0-13-162959-X
0-13-162959-X is correct.
0-13-162959-X1
0-13-162959-X1 is incorrect.

5. Hyphens at arbitrary locations

0-1-3-1-6-2-9-5-9-X
0-1-3-1-6-2-9-5-9-X is correct.

6. End of file marks the end of the input data

So, unless you encounter EOF, don't stop processing the data!!!, even though they are all BLANK LINES...

7. display of the candidate ISBN and a statement of whether it is legal or illegal
The output must be the candidate ISBN but REMOVE the starting and trailing additional spaces!

Input:

```
0-89237-010-6
0-89237-010 -6
-------0-89237-010-6--------
0-89237-010-6XXXX
0-89237-010-6-150
0-89237-010- 6 TEST
XX-0000000000-XX
XX000000XXX0000XXXXX
1234567890
0-89237-010-6
0-89237-010-6 TEST
   ==> notice a blank line
```

Output:

```
0-89237-010-6 is correct.
0-89237-010 -6 is incorrect.
-------0-89237-010-6-------- is correct.
0-89237-010-6XXXX is incorrect.
0-89237-010-6-150 is incorrect.
0-89237-010- 6 TEST is incorrect.
XX-0000000000-XX is incorrect.
XX000000XXX0000XXXXX is incorrect.
1234567890 is incorrect.
0-89237-010-6 is correct.
0-89237-010-6 TEST is incorrect.
 is incorrect.  ==> this is for blank line
```

334 - Identifying Concurrent Events

Use Floyd Warshall to compute transitive hull (if e1->e2 and e2->e3 then e1->e3) in $O(n^3)$

To count the total unconnected directed edges, calculate this:
total possible edges in a graph (n*(n-1)/2) - total directed edges after transitive hull

Then output the first 2 concurrent events.

335 - Processing MX Records

The problem description is very long -_-'. I forgot the details... my suggestion is you read the problem description very carefully. This problem is not difficult.

336 - A Node Too Far

Simple BFS problem, but I almost get very confused with this problem because I made a little mistake... so check the common mistake below:

Common Mistake:

1. Don't forget that the graph can be UNCONNECTED (this make me looks stupid)

Sample input:

```
5
1 2 2 3 3 4 4 5 6 7
1 0
1 1
1 2
1 3
1 4
1 5
1 6
1 7
```

Sample Output:

```
Case 1: 6 nodes not reachable from node 1 with TTL = 0.
Case 2: 5 nodes not reachable from node 1 with TTL = 1.
Case 3: 4 nodes not reachable from node 1 with TTL = 2.
Case 4: 3 nodes not reachable from node 1 with TTL = 3.
Case 5: 2 nodes not reachable from node 1 with TTL = 4.
Case 6: 2 nodes not reachable from node 1 with TTL = 5.
Case 7: 2 nodes not reachable from node 1 with TTL = 6.
Case 8: 2 nodes not reachable from node 1 with TTL = 7.
```

See, from TTL 4 onwards, it's useless since the graph is unconnected (6-7 are unconnected with 1-2-3-4, so these 2 nodes will always unreachable, don't forget that!!!)

2. The node number can be very big, better map the name into small indices.
3. Even with TTL = 0, you can still reach your starting node, isn't it?

337 - Interpreting Control Sequences

This problem belongs to "Output-Related" category. This is the "hard" version of 445 - Marvelous Mazes. You are given control sequences to format the output. And believe me, you will be very very angry if your code got Wrong Answer since this problem is easy but very complex...

-. You need an array to store the output buffer, so create a 10x10 array.
-. Scan the input, character by character, do what they want.
-. You need a flag to store INSERT/OVERWRITE status + a flag to store "^"/Control status.
-. You also need R & C variable to store the value of current row and current column.
-. Don't forget that R & C cannot go outside the boundary (0..9).
-. Don't forget to put a "nice" border for your picture :-)

339 - SameGame Simulation

A brute force simulation again, but you have to be very careful with array index out of bound, etc. This problem is EXACTLY SIMILAR to problem 758 (The Same Game), with some changes. You can solve 2 problems using one source code with some changes (not really minor changes though).

340 - Master-Mind Hints

What we have to do is to count "strong matches" and "weak matches"

Put the secret code to an array
Then do looping for each of line of input and do this

Example:

1 2 3 4 (secret code)
1 3 5 4 (Guess)

Do this to your array, remove strong matches + count them

x 2 3 x
z 3 5 z

Then do this, scan remaining weak matches + count them

x 2 x x
z z 5 z

Finally, output strong matches and weak matches

Don't re-count strong matches as weak matches and don't forget to flag counted items, it will be recounted again if you didn't flag them as counted.

341 - Non-Stop Travel

A simple shortest path problem, and moreover, the maximum number of cities is only 10. You can use brute force search, Djikstra shortest path, or even Floyd Warshall all pairs shortest path (the easiest in my opinion). This problem can be considered easy :)

343 - What Base Is This?

Given 2 number (unknown base), you must determine the smallest base so that these 2 numbers equals, or if it is not impossible, output an appropriate message.

Use base 10 as the comparison base.

```
Loop from i = minimum base for number1 until 36.
  Loop from j = minimum base for number2 until 36.
    compare the value of number1 in base i and number2 in base j (compare in base 10)
    if they equal, output the result
```

if they are still not equal, output the corresponding base

Common Mistake:

There are so many clever traps here...

1. NO BASE 1, You cannot have base 1 in your output.
Tricky test case:

0 0
Wrong output:
0 (base 1) = 0 (base 1)
Correct output:
0 (base 2) = 0 (base 2)

2. You must use the minimum base for each number as starting base
Tricky test case:
12 5
Wrong output:
12 (base 3) = 5 (base 5)
Correct output:
12 (base 3) = 5 (base 6)
See, the minimum base for 5 is 6 (why? consult your mathematic book)
start looping from 6, and not from 5

3. Use unsigned long!!!, the number can be VERY big, and make sure you handle this big numbers correctly, the implementation of this big number is up to you
4. Be careful with BLANK lines, the input file can have white spaces scattered everywhere.

344 - Roman Digititis (with help from: Yudha Irsandy)

Given an integer, convert it to roman numeral and count the frequency of 'i','v','x','l','c', straightforward but tricky.

346 - Getting Chorded

Actually, I'm very weak in music. However, this problem is just an ad hoc, follow the rule problem. Read the problem description carefully. You don't really need to know music...

347 - Run, Run, RunAround Numbers

Based on the information that a digit won't appear more than once in the number, and each digit is between 1 and 9 inclusive, then at most 9 digits in each test case, rite? At maximum, you do 1+2+...+8+9 = 45 iterations per test case (remember, all digits are different). So, brute force is "feasible" for this problem... pre-calculate all possible runaround numbers from 1 to 999.999.999 and store all these numbers in an array. Whenever you are asked which runaround number is greater than input number n, simply scan through your pre-calculated array and print the first one greater than n :).

348 - Optimal Array Multiplication Sequence

Dynamic Programming solution available, refer to programming section/your algorithm books.

349 - Transferable Voting (II)

There is nothing much I can say here except that you need to read the problem description of the voting system carefully, and then simulate it.

350 - Pseudo Random Numbers

You must compute the cycle length of a Pseudo Random Number.

I want to share a trick regarding how to speed up modulo operation.
L = (Z * L + I) % M is equal to L = ((Z % M * L % M) % M + I) % M

I'm not sure if this is faster or not, if Z, L are Big but M is small, then this formula will reduce the value to be quite small (faster). if Z,L,M are all small, then this formula will slower the computation a bit.

To determine a cycle, put a big array (size 10000->this is the max possible number for 4 digits), give a tag to if the array element already visited.

352 - The Seasonal War

Very easy problem if you know Graph Flood Fill algorithm =). Refer to my programming section regarding this Flood Fill algorithm.

353 - Pesky Palindromes

Based on the observation that maximum n is 80, most likely not all generated words are palindrome, and no repetitions allowed, we can safely use brute force. Generate all possible sub string and check whether they are palindrome (simple task, I believe). For similar palindrome, don't count twice... simple.

355 - The Bases Are Loaded

Base number conversion problem. This one plus illegal number checking. This problem is EXACTLY SIMILAR to problem 389 (Basically Speaking). You can solve 2 problems using one source code with some changes.

356 - Square Pegs And Round Holes

Phytagoras :-). Draw some pictures in a paper and you'll figure out how to determine which segments are contained in the circle and which segments are completely contained in the circle.

357 - Let Me Count The Ways

Another Coin Changing problem. A dynamic programming solution available for this problem. However they have changed the problem size up to 30.000, which requires you to use Big Integer library, since the number of ways will already exceed standard data type limit.

Note: Coin Changing problem also found in problem 147 (Dollars) and problem 674 (Coin Change). You can solve 3 problems using one similar source code.

362 - 18,000 Seconds Remaining

Very simple, just simulate the process. Maybe you can use this problem to test your coding skill. I think this problem should be solve-able within 15 minutes coding after you read the problem description... want to try?

369 - Combinations (by: Felix Halim)

The key idea is to prevent overflow... Only multiply if needed...

371 - Ackermann Functions

Another 3n+1 problem. Do a brute force calculating, modify your program (problem no 100) to calculate which one from a given interval, creates the longest 3n+1 sequences.

374 - Big Mod (with help from: Ilham Winata Kurnia)

You have to calculate the module of a very big number.
Use this formula:
R = (B^P) % M is equal to R = (B%M * B%M * (repeat B%M - P times...) * B%M) % M
Don't forget, use Divide & Conquer to compute this.

377 - Cowculations (with help from: Ilham Winata Kurnia)

This is a base-4 calculation problem. Just simulate it.

378 - Intersecting Lines

The line defined with two points is an infinite line. You can't assume the line terminate at those two endpoints. The best way to solve this problem is to use well known Line-Intersection algorithm. You may want to search the net for this term "line intersection" or "computational geometry".

379 - HI-Q

Simply simulate the game. Read the rules carefully.

380 - Call Forwarding

Given a list of Call Forwarding rules, determine to which number a call will be forwarded. Use backtracking.

381 - Making The Grade

This problem is for a real hard-worker programmers. This problem has so many restrictions, rules, etc. Believe me, you will not do it unless you really have time...

382 - Perfection

They want you to determine whether a number is Perfect, Deficient, or Abundant based on the definition.

The efficient way to determine whether a number N is Perfect, Deficient, or Abundant is to direct summing and checking like this:

Start from Counter=1, store the value to variable SUM
Next is Counter=2, if 2 can divide N, add the value to SUM
Next is Counter=3, if 3 can divide N, add the value to SUM
and so on...
Stop when Counter=N/2, because nothing bigger than N/2 can properly divide N.

If SUM>N => Abundant
If SUM=N => Perfect
If SUM<N => Deficient

Simple isn't it?

383 - Shipping Routes

This is a graph problem. Convert the input data into a graph and then find the shortest path distance, BFS will do.

384 - Slurpys

Read the problem carefully and construct a Recursive solution based on that.
No trap, No trick, just construct the recursive solution.

385 - DNA Translation

The underlying problem is "just" a string processing problem. If you have biology background, it will help you. But if you don't you can just read the problem description carefully do what they want. Just remember that DNA strand given in input can be in 4 forms: primary, complimentary, primary (reversed), or complimentary (reversed). You need to check these 4 possibilities. Don't worry if there exist multiple valid output. Problem 385 will be judged using special correction program (yellow label).

Tip: since computer is not biology, you may benefit by not bothering the existence of Uracil U, but simply use Thymine T inside the code. This will will simplify your task by removing the need of translating primary DNA strand to m-RNA. :)

386 - Perfect Cubes (with help from: Syukri)

A 4-nested loop ==> $O(n^4)$ brute force can be used to solve this problem.

389 - Basically Speaking

A base number conversion. To convert number from any base X to another base Y, you can use base 10 as intermediate base. Convert number from base X to base 10, and then convert the base 10 representation to base Y. Converting number from/to base 10 is usually taught in high school or simply refer to your mathematic books. This problem is EXACTLY SIMILAR to problem 355 (The Bases Are Loaded), with some changes. You can solve 2 problems using one source code with some changes.

391 - Mark-up

Believe me..., this problem is not easy... Even though this problem only related with output, because of the problem complexity, it is not solvable without careful debugging.

392 - Polynomial Showdown

Actually, this problem is not a mathematic problem. It is more like output-related problem. Just format the polynomial according to universal mathematic format.

394 - Mapmaker

Any n-dimensional array is stored in computer memory as single dimensional array. I'm sure Computer Science student must have learnt about this in their Programming Language class. In this problem, you are asked to calculate that address, quite simple.

397 - Equation Elation

The difficult part is only in parsing the input. Then just simulate the standard mathematical process and output the result on the screen.

Try if you can get this weird input correct (with unary + and - operators):
+1 * +2 + 6 / -2 - 1 = test

Output:
1 * 2 + 6 / -2 - 1 = test
2 + 6 / -2 - 1 = test
2 + -3 - 1 = test
-1 - 1 = test
-2 = test
