

Archives of the Sphere Online Judge

Editors:

Adrian Kosowski
Roman Sol
Michal Czuczman
Michal Malafiejski
Piotr Lowiec
Thanh Vy Hua Le
Adam Dzedzej
Adrian Kuegel
Ngo Minh Duc
Pawel Gawrychowski
Simon Gog
Lukasz Kuszner
Csaba Noszaly
Darek Dereniowski
Neal Zane
Patryk Pomykalski
Tomasz Niedzwiecki
Tomek Czajka
Krzysztof Kluczek
Piotr Piotrowski
Adam Nadolski
Konrad Piwakowski
Robin Nittka
Pawel Dobrzycki
Maxim A. Sukhov
Lukasz Wrona
Rafal Nowak

Last updated: 2005-06-01 18:21:23

Preface

This electronic material contains a set of algorithmic problems, forming the archives of the Sphere Online Judge site (<http://spoj.sphere.pl>). The document can be accessed at the following URLs:

- in PostScript format: <http://spoj.sphere.pl/problems.ps>
- in Portable Document Format: <http://spoj.sphere.pl/problems.pdf>

These resources are constantly updated to synchronise with the ever-changing hypertext version of the problems, and to include newly added problems. If you have obtained this document from another source, it is strongly recommended that you should download the current version from one of the aforementioned URLs.

Enjoy problem-solving at the Sphere Online Judge!

Disclaimer from the Editors. Despite our best efforts, it is possible that this document contains errors or that some of the content differs slightly from its original hypertext form. We take no responsibility for any such faults and their consequences. We neither authorise nor approve use of this material for any purpose other than facilitating problem solving at the Sphere Online Judge site; nor do we guarantee its fitness for any purpose whatsoever.

The layout of the problems in this document is the copyright of the Editors named on the cover (as determined by the appropriate footers in the problem description). The content is the copyright of the respective Editor unless the copyright holder is otherwise stated in the 'resource' section. The document as a whole is not protected by copyright, and fragments of it are to be regarded independently. No responsibility is taken by the Editors if use or redistribution of this document violates either their or third party copyright laws. When referring to or citing the whole or a fragment of this document, please state clearly the aforementioned URLs at which the document is to be found, as well as the resources from which the problems you are referring to originally came.

Remarks concerning this document should be sent to the following e-mail address: contact@spoj.sphere.pl.

Table of Contents

1. Problem TEST (1. Life, the Universe, and Everything)
2. Problem PRIME1 (2. Prime Generator)
3. Problem SBSTR1 (3. Substring Check (Bug Funny))
4. Problem ONP (4. Transform the Expression)
5. Problem PALIN (5. The Next Palindrome)
6. Problem ARITH (6. Simple Arithmetics)
7. Problem BULK (7. The Bulk!)
8. Problem CMPLS (8. Complete the Sequence!)
9. Problem DIRVS (9. Direct Visibility)
10. Problem CMEXPR (10. Complicated Expressions)
11. Problem FCTRL (11. Factorial)
12. Problem MMIND (12. The Game of Master-Mind)
13. Problem HOTLINE (13. Hotline)
14. Problem IKEYB (14. I-Keyboard)
15. Problem SHPATH (15. The Shortest Path)
16. Problem TETRA (16. Sphere in a tetrahedron)
17. Problem CRYPTO1 (17. The Bytelandian Cryptographer (Act I))
18. Problem CRYPTO2 (18. The Bytelandian Cryptographer (Act II))
19. Problem CRYPTO3 (19. The Bytelandian Cryptographer (Act III))
20. Problem CRYPTO4 (20. The Bytelandian Cryptographer (Act IV))
21. Problem TRICENTR (22. Triangle From Centroid)
22. Problem PIR (23. Pyramids)
23. Problem FCTRL2 (24. Small factorials)
24. Problem POUR1 (25. Pouring water)
25. Problem BSHEEP (26. Build the Fence)
26. Problem SBANK (27. Sorting Bank Accounts)
27. Problem HMRO (28. Help the Military Recruitment Office!)
28. Problem HASHIT (29. Hash it!)
29. Problem BLINNET (30. Bytelandian Blingors Network)
30. Problem MUL (31. Fast Multiplication)
31. Problem NHAY (32. A Needle in the Haystack)
32. Problem TRIP (33. Trip)
33. Problem RUNAWAY (34. Run Away)
34. Problem EQBOX (35. Equipment Box)
35. Problem CODE1 (36. Secret Code)
36. Problem PROPKEY (37. The Proper Key)
37. Problem LABYR1 (38. Labyrinth)
38. Problem PIGBANK (39. Piggy-Bank)
39. Problem STONE (40. Lifting the Stone)
40. Problem WORDS1 (41. Play on Words)
41. Problem ADDREV (42. Adding Reversed Numbers)
42. Problem BOOKS1 (43. Copying Books)
43. Problem SCYPHER (44. Substitution Cipher)
44. Problem COMMEDIA (45. Commedia dell Arte)
45. Problem SCRAPER (47. Skyscraper Floors)

46. Problem BEADS (48. Glass Beads)
47. Problem HAREFOX (49. Hares and Foxes)
48. Problem INCARDS (50. Invitation Cards)
49. Problem TOUR (51. Fake tournament)
50. Problem KAMIL (53. Kamil)
51. Problem JULKA (54. Julka)
52. Problem JASIEK (55. Jasiek)
53. Problem DYZIO (56. Dyzio)
54. Problem SUPPER (57. Supernumbers in a permutation)
55. Problem PICAD (58. Crime at Piccadily Circus)
56. Problem BIA (59. Bytelandian Information Agency)
57. Problem DANCE (60. The Gordian Dance)
58. Problem BRCKTS (61. Brackets)
59. Problem IMP (62. The Imp)
60. Problem SQRBR (63. Square Brackets)
61. Problem PERMUT1 (64. Permutations)
62. Problem BALL1 (65. Ball)
63. Problem CRSCNTRY (66. Cross-country)
64. Problem CUTOOT (67. Cutting out)
65. Problem EXPR1 (68. Expression)
66. Problem MOULDS (69. Moulds)
67. Problem RELATS1 (70. Relations)
68. Problem TREE1 (71. Tree)
69. Problem BYTEFOOD (72. Food Shortage in Byteland)
70. Problem BAC (73. Bacterial)
71. Problem DIVSUM (74. Divisor Summation)
72. Problem EDIT1 (75. Editor)
73. Problem EDIT2 (76. Editor Inverse)
74. Problem BRICKS (77. New bricks disorder)
75. Problem MARBLES (78. Marbles)
76. Problem EASYPIE (82. Easy Problem)
77. Problem BUNDLE (83. Bundling)
78. Problem SHORTCUT (84. Shortcut)
79. Problem DICE1 (85. Dice Contest)
80. Problem RAIN1 (86. November Rain)
81. Problem FOOTBALL (87. Football)
82. Problem TREE2 (88. Which is Next)
83. Problem HANGLET (89. Hang or not to hang)
84. Problem MINIMAX (90. Minimizing maximizer)
85. Problem TWOSQRS (91. Two squares or not two squares)
86. Problem CUTSQRS (92. Cutting off Squares)
87. Problem MAYA (94. Numeral System of the Maya)
88. Problem STPAR (95. Street Parade)
89. Problem SHOP (96. Shopping)
90. Problem PARTY (97. Party Schedule)
91. Problem DFLOOR (98. Dance Floor)
92. Problem BUS (99. Bus)

- 93. Problem BABTWR (100. Tower of Babylon)
- 94. Problem FISHER (101. Fishmonger)
- 95. Problem LITEPIPE (102. GX Light Pipeline Inc)
- 96. Problem HIGH (104. Highways)
- 97. Problem ALICEBOB (105. Alice and Bob)
- 98. Problem BINSTIRL (106. Binary Stirling Numbers)
- 99. Problem MAYACAL (107. Calendar of the Maya)
- 100. Problem MORSE (108. Decoding Morse Sequences)
- 101. Problem EXCHNG (109. Exchanges)
- 102. Problem CISTFILL (110. Fill the Cisterns)
- 103. Problem SEGVIS (112. Horizontally Visible Segments)
- 104. Problem FAMILY (115. Family)
- 105. Problem INTERVAL (116. Intervals)
- 106. Problem RHOMBS (118. Rhombs)
- 107. Problem SERVERS (119. Servers)
- 108. Problem SOLIT (120. Solitaire)
- 109. Problem TTABLE (121. Timetable)
- 110. Problem STEVE (122. Voracious Steve)
- 111. Problem PAYING (123. Paying in Byteland)
- 112. Problem SOLSTRAS (126. Solovay-Strassen Inverted)
- 113. Problem JOHNNY (127. Johnny Goes Shopping)
- 114. Problem RENT (130. Rent your airplane and make money)
- 115. Problem SQDANCE (131. Square dance)
- 116. Problem HELPR2D2 (132. Help R2-D2!)
- 117. Problem PHONY (134. Phony Primes)
- 118. Problem MAWORK (135. Men at work)
- 119. Problem TRANS (136. Transformation)
- 120. Problem PARTIT (137. Partition)
- 121. Problem POSTERS (138. Election Posters)
- 122. Problem MAZE (139. The Long and Narrow Maze)
- 123. Problem LONER (140. The Loner)
- 124. Problem GLUE (142. Johnny and the Glue)
- 125. Problem ALIENS (145. Aliens)
- 126. Problem MULTIPLY (146. Fast Multiplication Again)
- 127. Problem TAUT (147. Tautology)
- 128. Problem MLAND (148. Land for Motorways)
- 129. Problem FSHEEP (149. Fencing in the Sheep)
- 130. Problem PLONK (150. Where to Drink the Plonk?)
- 131. Problem COURIER (151. The Courier)
- 132. Problem SCALES (153. Balancing the Stone)
- 133. Problem ROCK (154. Sweet and Sour Rock)
- 134. Problem SOLVING (155. Solving the Puzzle)
- 135. Problem PALSEC (160. Choosing a Palindromic Sequence)
- 136. Problem PAINTTMP (174. Paint templates)
- 137. Problem POLY1 (175. Polygon)
- 138. Problem SUM1SEQ (176. Sum of one-sequence)
- 139. Problem ABWORDS (177. AB-words)

- 140. Problem ROADNET (178. Road net)
- 141. Problem WORDEQ (179. Word equations)
- 142. Problem CONTPACK (180. How to pack containers)
- 143. Problem SCUBADIV (181. Scuba diver)
- 144. Problem WINDOW1 (182. Window)
- 145. Problem ASCIRC (183. Assembler circuits)
- 146. Problem ATMS (184. Automatic Teller Machines)
- 147. Problem CHASE1 (185. Chase)
- 148. Problem LITELANG (186. The lightest language)
- 149. Problem FLBRKLIN (187. Flat broken lines)
- 150. Problem RECTNG1 (188. Rectangles)
- 151. Problem MUSKET (196. Musketeers)
- 152. Problem EMPTY (199. Empty Cuboids)
- 153. Problem MONODIG (200. Monodigital Representations)
- 154. Problem POLYGAME (201. The Game of Polygons)
- 155. Problem ROCKETS (202. Rockets)
- 156. Problem POTHOLE (203. Potholers)
- 157. Problem SLEEP (204. Sleepwalker)
- 158. Problem ICERINK (205. Icerink)
- 159. Problem BITMAP (206. Bitmap)
- 160. Problem THREECOL (207. Three-coloring of binary trees)
- 161. Problem STORE (208. Store-keeper)
- 162. Problem MAP (209. The Map)
- 163. Problem ALTARS (210. The Altars)
- 164. Problem PRIMIT (211. Primitivus recurencis)
- 165. Problem WATER (212. Water among Cubes)
- 166. Problem PANIC (215. Panic in the Plazas)
- 167. Problem SOPARADE (217. Soldiers on Parade)
- 168. Problem PSPHERE (218. Points on a Sphere)
- 169. Problem PAWNS (219. Pawns Gone Wild)
- 170. Problem PHRASES (220. Relevant Phrases of Annihilation)
- 171. Problem BURNCITY (222. The Burning City)
- 172. Problem VONNY (224. Vonny and her dominos)
- 173. Problem HANOI (225. Nightmare in the Towers of Hanoi)
- 174. Problem JEWELS (226. Jewelry and Fashion)
- 175. Problem ORDERS (227. Ordering the Soldiers)
- 176. Problem SHAMAN (228. Shamans)
- 177. Problem SORTING (229. Sorting is easy)
- 178. Problem BYTELE (232. Bytelandian Telecom)
- 179. Problem HOLIDAY1 (234. Getting Rid of the Holidays (Act I))
- 180. Problem VFMUL (235. Very Fast Multiplication)
- 181. Problem ROMAN (236. Converting number formats)
- 182. Problem SUMITR (237. Sums in a Triangle)
- 183. Problem HOLIDAY2 (238. Getting Rid of the Holidays (Act II))
- 184. Problem BTOUR (239. Tour de Byteland)
- 185. Problem SANTA (240. Santa Claus and the Presents)
- 186. Problem BLOCKS (241. Arranging the Blocks)

- 187. Problem STABLEMP (243. Stable Marriage Problem)
- 188. Problem INTSS (244. Internally Stable Sets)
- 189. Problem SQRROOT (245. Square Root)
- 190. Problem CTQUINE (246. Plant a Christmas Tree)
- 191. Problem CHOCOLA (247. Chocolate)
- 192. Problem CTAIN (260. Containers)
- 193. Problem TRIPART (261. Triangle Partitioning)
- 194. Problem CONNECT (262. Connections)
- 195. Problem PERIOD (263. Period)
- 196. Problem CORNET (264. Corporative Network)
- 197. Problem PIVAL (270. Digits of Pi)
- 198. Problem CAVE (272. Cave Exploration)
- 199. Problem DCODE (273. The Modern Dress Code)
- 200. Problem WMELON (274. Johnny and the Watermelon Plantation)
- 201. Problem WATERWAY (275. The Water Ringroad)
- 202. Problem MFENCE (276. Herdkeeper)
- 203. Problem CTGAME (277. City Game)
- 204. Problem BICYCLE (278. Bicycle)
- 205. Problem INUMBER (279. Interesting number)
- 206. Problem LIFTS (280. Lifts)
- 207. Problem MUDDY (282. Muddy Fields)
- 208. Problem NAPTIME (283. Naptime)
- 209. Problem ATSHELT (285. Atomic Shelters)
- 210. Problem SCITIES (286. Selfish Cities)
- 211. Problem NETADMIN (287. Smart Network Administrator)
- 212. Problem PON (288. Prime or Not)
- 213. Problem TMBOX (289. The Turing Music Box)
- 214. Problem POLYEQ (290. Polynomial Equations)
- 215. Problem CUBERT (291. Cube Root)
- 216. Problem ALIBB (292. Alibaba)
- 217. Problem OFBEAT (293. Officers on the Beat)
- 218. Problem HWORK (294. Johnny and the Optimisation of Homework)
- 219. Problem BRIGAMI (295. Bytelandian Origami)
- 220. Problem TWORK (296. Teamwork is Crucial)
- 221. Problem AGGRCOW (297. Aggressive cows)
- 222. Problem DERAILE (298. Closing down Railway Lines)
- 223. Problem CABLETV (300. Cable TV Network)
- 224. Problem BOOK (301. Booklets)
- 225. Problem CANTON (302. Count on Canton)
- 226. Problem UCUBE (303. The Unstable Cube)
- 227. Problem RATTERN (309. The Room Pattern)
- 228. Problem CROSSES (313. The Game of Crosses & Crosses)
- 229. Problem EVAL (314. Digits of e)
- 230. Problem BFORG (315. The Secret Fellowship of Byteland)
- 231. Problem JCROSS (316. Japan Crossword)
- 232. Problem IMGREC1 (317. Simple Image Recognition)
- 233. Problem PITPAIR (318. Pythagorean Legacy)

234. Problem XWORDS (321. X-Words)
235. Problem WINDMILL (325. The Tall Windmills)
236. Problem MGAME (326. Enjoying a Multiplayer Game)
237. Problem BISHOPS (328. Bishops)
238. Problem CALLS (329. Calls)
239. Problem HARDQ (332. Hard Question)
240. Problem PHDISP (334. The Philosophical Dispute)
241. Problem EOPERA (336. Exchange Operations)
242. Problem ROADS (338. Roads)
243. Problem SEQ (339. Recursive Sequence)
244. Problem POKER (344. Poker)
245. Problem MIXTURES (345. Mixtures)
246. Problem COINS (346. Bytelandian gold coins)
247. Problem LAZYCOWS (347. Lazy Cows)
248. Problem EXPEDI (348. Expedition)
249. Problem AROUND (349. Around the world)
250. Problem LANDSCAP (350. Landscaping)
251. Problem HAN01 (351. Ha-noi!)
252. Problem DISPLACE (353. Displace)
253. Problem ACT (359. Alpha Centauri Tennis)
254. Problem BCEASY (360. Bottom Coder (Easy))
255. Problem BCHARD (361. Bottom Coder (Hard))
256. Problem IGARB (362. Ignore the Garbage)
257. Problem COPER (363. Crane Operator)
258. Problem LISA (364. Pocket Money)
259. Problem PHIDIAS (365. Phidias)
260. Problem FARMER (366. Farmer)
261. Problem EMPDIA (367. Empodia)
262. Problem CSTREET (368. Cobbled streets)
263. Problem MATH1 (369. Math I)
264. Problem ONEZERO (370. Ones and zeros)
265. Problem BOXES (371. Boxes)
266. Problem TCONNUM (10000. Simple Numbers Conversion)
267. Problem TMUL (10010. Not So Fast Multiplication)
268. Problem TSHPATH (10011. The Turtles Shortest Path)
269. Problem TDBFS (10013. Searching the Graph)
270. Problem TCNUMFL (10021. Simple Numbers with Fractions Conversion)
271. Problem INTEST (10022. Enormous Input Test)
272. Problem TLCS (10024. Longest Common Subsequence)
273. Problem SUMTRIAN (10025. Sums in a Triangle (tutorial))
274. Problem TPERML (10051. Permutation generator)
275. Problem KMSL4 (10052. Jordan canonical form)
276. Problem KMSL4B (10053. Roots of polynomial)
277. Problem TCONVEX (10055. Convex Hull)
278. Problem TFOSS (10056. Fossil in the Ice)
279. Problem ARMIES (10062. Armies)
280. Problem MMATCH (10063. The Cursed Room)

- 281. Problem TSORT (10072. Turbo Sort)
- 282. Problem PRINT (10075. Prime Intervals)
- 283. Problem LEXISORT (10083. Easy Sorting)
- 284. Problem CLTZ (10087. Collatz)
- 285. Problem ZZPERM (10090. Zig-Zag Permutation)
- 286. Problem DIV (10098. Divisors)
- 287. Problem J4FUN (10099. Just for Fun (Easy))
- 288. Problem DIV2 (10102. Divisors 2)

SPOJ Problem Set

1. Life, the Universe, and Everything

Problem code: TEST

Your program is to use the brute-force approach in order to *find the Answer to Life, the Universe, and Everything*. More precisely... rewrite small numbers from input to output. Stop processing input after reading in the number 42. All numbers at input are integers of one or two digits.

Example

Input :

1
2
88
42
99

Output :

1
2
88

Added by: Michal Malafiejski
Date: 2004-05-01
Time limit [s]: 3
Source limit [B]:50000
Resource: Douglas Adams, The Hitchhiker's Guide to the Galaxy

SPOJ Problem Set

2. Prime Generator

Problem code: PRIME1

Peter wants to generate some prime numbers for his cryptosystem. Help him! Your task is to generate all prime numbers between two given numbers!

Input

The input begins with the number t of test cases in a single line ($t \leq 10$). In each of the next t lines there are two numbers m and n ($1 \leq m \leq n \leq 1000000000$, $n-m \leq 100000$) separated by a space.

Output

For every test case print all prime numbers p such that $m \leq p \leq n$, one number per line, test cases separated by an empty line.

Example

Input :

```
2
1 10
3 5
```

Output :

```
2
3
5
7

3
5
```

Warning: large Input/Output data, be careful with certain languages (though most should be OK if the algorithm is well designed)

Added by: Adam Dzedzej

Date: 2004-05-01

Time limit [s]: 6

Source limit [B]: 50000

SPOJ Problem Set

3. Substring Check (Bug Funny)

Problem code: SBSTR1

Given two binary strings, A (of length 10) and B (of length 5), output 1 if B is a substring of A and 0 otherwise.

*Please note, that the solution may only be submitted in the following languages: Brainf**k, Whitespace and Intercal.*

Input

24 lines consisting of pairs of binary strings A and B separated by a single space.

Output

The logical value of: 'B is a substring of A'.

Example

First two lines of input:

1010110010 10110

1110111011 10011

First two lines of output:

1

0

Added by: Adrian Kosowski

Date: 2004-05-01

Time limit [s]: 15

Source limit [B]:50000

SPOJ Problem Set

4. Transform the Expression

Problem code: ONP

Transform the algebraic expression with brackets into RPN form (Reverse Polish Notation). Two-argument operators: +, -, *, /, ^ (priority from the lowest to the highest), brackets (). Operands: only letters: a,b,...,z. Assume that there is only one RPN form (no expressions like a*b*c).

Input

```
t [the number of expressions <= 100]
expression [length <= 400]
[other expressions]
```

Text grouped in [] does not appear in the input file.

Output

The *expressions* in RPN form, one per line.

Example

```
Input:
3
(a+(b*c))
((a+b)*(z+x))
((a+t)*((b+(a+c))^(c+d)))
```

```
Output:
abc*+
ab+zx+*
at+bac++cd+^*
```

Added by: Michal Malafiejski
Date: 2004-05-01
Time limit [s]: 10
Source limit [B]:50000
Resource: PAL

SPOJ Problem Set

5. The Next Palindrome

Problem code: PALIN

A positive integer is called a *palindrome* if its representation in the decimal system is the same when read from left to right and from right to left. For a given positive integer K of not more than 1000000 digits, write the value of the smallest palindrome larger than K to output. Numbers are always displayed without leading zeros.

Input

The first line contains integer t , the number of test cases. Integers K are given in the next t lines.

Output

For each K , output the smallest palindrome larger than K .

Example

Input :

2
808
2133

Output :

818
2222

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-05-01

Time limit [s]: 20

Source limit [B]:50000

SPOJ Problem Set

6. Simple Arithmetics

Problem code: ARITH

One part of the new WAP portal is also a calculator computing expressions with very long numbers. To make the output look better, the result is formatted the same way as is it usually used with manual calculations.

Your task is to write the core part of this calculator. Given two numbers and the requested operation, you are to compute the result and print it in the form specified below. With addition and subtraction, the numbers are written below each other. Multiplication is a little bit more complex: first of all, we make a partial result for every digit of one of the numbers, and then sum the results together.

Input

There is a single positive integer T on the first line of input (equal to about 1000). It stands for the number of expressions to follow. Each expression consists of a single line containing a positive integer number, an operator (one of +, - and *) and the second positive integer number. Every number has at most 500 digits. There are no spaces on the line. If the operation is subtraction, the second number is always lower than the first one. No number will begin with zero.

Output

For each expression, print two lines with two given numbers, the second number below the first one, last digits (representing unities) must be aligned in the same column. Put the operator right in front of the first digit of the second number. After the second number, there must be a horizontal line made of dashes (-).

For each addition or subtraction, put the result right below the horizontal line, with last digit aligned to the last digit of both operands.

For each multiplication, multiply the first number by each digit of the second number. Put the partial results one below the other, starting with the product of the last digit of the second number. Each partial result should be aligned with the corresponding digit. That means the last digit of the partial product must be in the same column as the digit of the second number. No product may begin with any additional zeros. If a particular digit is zero, the product has exactly one digit -- zero. If the second number has more than one digit, print another horizontal line under the partial results, and then print the sum of them.

There must be minimal number of spaces on the beginning of lines, with respect to other constraints. The horizontal line is always as long as necessary to reach the left and right end of both numbers (and operators) directly below and above it. That means it begins in the same column where the leftmost digit or operator of that two lines (one below and one above) is. It ends in the column where is the rightmost digit of that two numbers. The line can be neither longer nor shorter than specified.

Print one blank line after each test case, including the last one.

Example

Sample Input:

```
4
12345+67890
324-111
325*4405
1234*4
```

Sample Output:

```
  12345
+67890
-----
  80235

  324
-111
----
  213

    325
  *4405
  ----
    1625
      0
  1300
1300
-----
1431625

1234
 *4
----
4936
```

Warning: large Input/Output data, be careful with certain languages.

Added by: Adrian Kosowski

Date: 2004-05-08

Time limit [s]: 10

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 2000

SPOJ Problem Set

7. The Bulk!

Problem code: BULK

ACM uses a new special technology of building its transceiver stations. This technology is called *Modular Cuboid Architecture (MCA)* and is covered by a patent of Lego company. All parts of the transceiver are shipped in unit blocks that have the form of cubes of exactly the same size. The cubes can be then connected to each other. The MCA is modular architecture, that means we can select preferred transceiver configuration and buy only those components we need.

The cubes must be always connected "face-to-face", i.e. the whole side of one cube is connected to the whole side of another cube. One cube can be thus connected to at most six other units. The resulting equipment, consisting of unit cubes is called *The Bulk* in the communication technology slang.

Sometimes, an old and unneeded bulk is condemned, put into a storage place, and replaced with a new one. It was recently found that ACM has many of such old bulks that just occupy space and are no longer needed. The director has decided that all such bulks must be disassembled to single pieces to save some space. Unfortunately, there is no documentation for the old bulks and nobody knows the exact number of pieces that form them. You are to write a computer program that takes the bulk description and computes the number of unit cubes.

Each bulk is described by its faces (sides). A special X-ray based machine was constructed that is able to localise all faces of the bulk in the space, even the inner faces, because the bulk can be partially hollow (it can contain empty spaces inside). But any bulk must be connected (i.e. it cannot drop into two pieces) and composed of whole unit cubes.

Input

There is a single positive integer T on the first line of input (equal to about 1000). It stands for the number of bulks to follow. Each bulk description begins with a line containing single positive integer F , $6 \leq F \leq 250$, stating the number of faces. Then there are F lines, each containing one face description. All faces of the bulk are always listed, in any order. Any face may be divided into several distinct parts and described like if it was more faces. Faces do not overlap. Every face has one inner side and one outer side. No side can be "partially inner and partially outer".

Each face is described on a single line. The line begins with an integer number P stating the number of points that determine the face, $4 \leq P \leq 200$. Then there are $3 \times P$ numbers, coordinates of the points. Each point is described by three coordinates X,Y,Z ($0 \leq X,Y,Z \leq 1000$) separated by spaces. The points are separated from each other and from the number P by two space characters. These additional spaces were added to make the input more human readable. The face can be constructed by connecting the points in the specified order, plus connecting the last point with the first one.

The face is always composed of "unit squares", that means every edge runs either in X , Y or Z -axis direction. If we take any two neighbouring points X_1,Y_1,Z_1 and X_2,Y_2,Z_2 , then the points will always differ in exactly one of the three coordinates. I.e. it is either $X_1 \neq X_2$, or $Y_1 \neq Y_2$, or $Z_1 \neq Z_2$, other two coordinates are the same. Every face lies in an orthogonal plane, i.e. exactly one coordinate is always the same for all points of the face. The face outline will never touch nor cross

itself.

Output

Your program must print a single line for every test case. The line must contain the sentence The bulk is composed of V units., where V is the volume of the bulk.

Example

Sample Input:

```
2
12
4 10 10 10 10 10 20 10 20 20 10 20 10
4 20 10 10 20 10 20 20 20 20 20 10
4 10 10 10 10 10 20 20 10 20 20 10 10
4 10 20 10 10 20 20 20 20 20 20 10
4 10 10 10 10 20 10 20 20 10 20 10 10
5 10 10 20 10 20 20 20 20 20 20 15 20 20 10 20
4 14 14 14 14 14 16 14 16 16 14 16 14
4 16 14 14 16 14 16 16 16 16 16 14
4 14 14 14 14 14 16 16 14 16 16 14 14
4 14 16 14 14 16 16 16 16 16 16 14
4 14 14 14 14 16 14 16 16 14 16 14 14
4 14 14 16 14 16 16 16 16 16 14 16
12
4 20 20 30 20 30 30 30 30 30 20 30
4 10 10 10 10 40 10 40 40 10 40 10 10
6 10 10 20 20 10 20 20 30 20 30 30 20 30 40 20 10 40 20
6 20 10 20 20 20 20 30 20 20 30 40 20 40 40 20 40 10 20
4 10 10 10 40 10 10 40 10 20 10 10 20
4 10 40 10 40 40 10 40 40 20 10 40 20
4 20 20 20 30 20 20 30 20 30 20 20 30
4 20 30 20 30 30 20 30 30 30 20 30 30
4 10 10 10 10 40 10 10 40 20 10 10 20
4 40 10 10 40 40 10 40 40 20 40 10 20
4 20 20 20 20 30 20 20 30 30 20 20 30
4 30 20 20 30 30 20 30 30 30 30 20 30
```

Sample Output:

```
The bulk is composed of 992 units.
The bulk is composed of 10000 units.
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-05-08

Time limit [s]: 15

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 2000

SPOJ Problem Set

8. Complete the Sequence!

Problem code: CMPLS

You probably know those quizzes in Sunday magazines: given the sequence 1, 2, 3, 4, 5, what is the next number? Sometimes it is very easy to answer, sometimes it could be pretty hard. Because these "sequence problems" are very popular, ACM wants to implement them into the "Free Time" section of their new WAP portal.

ACM programmers have noticed that some of the quizzes can be solved by describing the sequence by polynomials. For example, the sequence 1, 2, 3, 4, 5 can be easily understood as a trivial polynomial. The next number is 6. But even more complex sequences, like 1, 2, 4, 7, 11, can be described by a polynomial. In this case, $\frac{1}{2}.n^2 - \frac{1}{2}.n + 1$ can be used. Note that even if the members of the sequence are integers, polynomial coefficients may be any real numbers.

Polynomial is an expression in the following form:

$$P(n) = a_D . n^D + a_{D-1} . n^{D-1} + \dots + a_1 . n + a_0$$

If $a_D \neq 0$, the number D is called a *degree* of the polynomial. Note that constant function $P(n) = C$ can be considered as polynomial of degree 0, and the zero function $P(n) = 0$ is usually defined to have degree -1.

Input

There is a single positive integer T on the first line of input (equal to about 5000). It stands for the number of test cases to follow. Each test case consists of two lines. First line of each test case contains two integer numbers S and C separated by a single space, $1 \leq S < 100$, $1 \leq C < 100$, $(S+C) \leq 100$. The first number, S , stands for the length of the given sequence, the second number, C is the amount of numbers you are to find to complete the sequence.

The second line of each test case contains S integer numbers X_1, X_2, \dots, X_S separated by a space. These numbers form the given sequence. The sequence can always be described by a polynomial $P(n)$ such that for every i , $X_i = P(i)$. Among these polynomials, we can find the polynomial P_{min} with the lowest possible degree. This polynomial should be used for completing the sequence.

Output

For every test case, your program must print a single line containing C integer numbers, separated by a space. These numbers are the values completing the sequence according to the polynomial of the lowest possible degree. In other words, you are to print values $P_{min}(S+1), P_{min}(S+2), \dots, P_{min}(S+C)$.

It is guaranteed that the results $P_{min}(S+i)$ will be non-negative and will fit into the standard *integer* type.

Example

Sample Input:

```
4
6 3
1 2 3 4 5 6
8 2
1 2 4 7 11 16 22 29
10 2
1 1 1 1 1 1 1 1 1 2
1 10
3
```

Sample Output:

```
7 8 9
37 46
11 56
3 3 3 3 3 3 3 3 3
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-05-08

Time limit [s]: 10

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Prague 2000

SPOJ Problem Set

9. Direct Visibility

Problem code: DIRVS

Building the GSM network is a very expensive and complex task. Moreover, after the *Base Transceiver Stations (BTS)* are built and working, we need to perform many various measurements to determine the state of the network, and propose effective improvements to be made.

The ACM technicians have a special equipment for measuring the strength of electro-magnetic fields, the transceivers' power and quality of the signal. This equipment is packed into a huge knapsack and the technician must move with it from one BTS to another. Unfortunately, the knapsack have not enough memory for storing all of the measured values. It has a small cache only, that can store values for several seconds. Then the values must be transmitted to the BTS by an infrared connection (IRDA). The IRDA needs direct visibility between the technician and the BTS.

Your task is to find the path between two neighbouring BTSes such that at least one of those BTSes is always visible.

Input

There is a single positive integer T on the first line of input (equal to about 500). It stands for the number of test cases to follow. Each test case consists of a town description. For simplicity, a town is modelled as a rectangular grid of $P \times Q$ square fields. Each field is exactly 1 metre wide. For each field, a non-negative integer $Z_{i,j}$ is given, representing the height of the terrain in that place, in metres. That means the town model is made of cubes, each of them being either solid or empty. There are no "half solid" cubes.

The first line of each test case contains two integer numbers P and Q , separated by a single space, $1 \leq P, Q \leq 200$. Then there are P lines each containing Q integer numbers separated by a space. These numbers are $Z_{i,j}$, where $1 \leq i \leq P$, $1 \leq j \leq Q$ and $0 \leq Z_{i,j} \leq 5000$. After the terrain description, there are four numbers R_1, C_1, R_2, C_2 on the last line of each test case. These numbers represent position of two BTSes, $1 \leq R_1, R_2 \leq P$, $1 \leq C_1, C_2 \leq Q$. The first coordinate (R) determines the row of the town, the second coordinate determines the column.

The technician is moving in steps (*steps* stands for *Standard Technician's Elementary Positional Shift*). Each step is made between two neighbouring square fields. That means the step is always in North, South, West or East direction. It is not possible to move diagonally. The step between two fields A and B (step from A to B) is allowed only if the height of the terrain in the field B is not very different from the height in the field A . The technician can climb at most 1 metre up or descend at most 3 metres down in a single step.

At the end of each step, at least one of the two BTSes must be visible. However, there can be some point "in the middle of the step" where no BTS is visible. This is OK and the data is handled by the cache. The BTS is considered visible, if there is a direct visibility between the unit cube just above the terrain on the BTSes coordinates and the cube just above the terrain on the square field, where the technician is. Direct visibility between two cubes means that the line connecting the centres of the two cubes does not intersect any solid cube. However, the line can touch any number of solid cubes. In

other words, consider both the BTS and the technician being points exactly half metre above the surface and in the centre of the appropriate square field.

Note that the IRDA beam can go between two cubes that touch each other by their edge, although there is no space between them. It is because such a beam touches both of these two cubes but does not intersect any of them. See the last test case of the sample input for an example of such a situation.

Output

You are to find the shortest possible path meeting the above criteria. All steps must be done between neighbouring fields, the terrain must not elevate or descend too much, and at the end of each step, at least one BTS must be visible.

For each test case, print one line containing the sentence The shortest path is M steps long., where M is the number of steps that must be made. If there is no such path, output the sentence Mission impossible!.

Example

Sample Input:

```
4
5 5
8 7 6 5 4
2 2 2 2 2
2 2 2 2 2
2 2 2 2 2
2 2 2 2 2
1 1 5 1
5 8
2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2
9 9 9 9 9 9 9 2
2 2 2 2 2 2 2 2
1 2 5 1
5 8
2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2
9 9 9 9 9 9 9 2
2 2 2 2 2 2 2 2
1 5 5 1
6 12
5 5 5 5 1 5 5 5 5 5 5 5
5 5 5 5 1 5 5 5 5 5 5 5
5 5 5 5 9 5 5 5 5 5 5 5
5 9 1 5 5 5 5 5 5 5 5 5
5 5 9 5 5 5 5 5 5 5 5 5
5 5 9 5 5 5 5 5 5 5 5 5
6 1 3 12
```

Sample Output:

The shortest path is 10 steps long.
Mission impossible!
The shortest path is 14 steps long.
The shortest path is 18 steps long.

Added by: Adrian Kosowski
Date: 2004-05-08
Time limit [s]: 30
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Prague 2000

SPOJ Problem Set

10. Complicated Expressions

Problem code: CMEXPR

The most important activity of ACM is the GSM network. As the mobile phone operator, ACM must build its own transmitting stations. It is very important to compute the exact behaviour of electro-magnetic waves. Unfortunately, prediction of electro-magnetic fields is a very complex task and the formulas describing them are very long and hard-to-read. For example, Maxwell's Equations describing the basic laws of electrical engineering are really tough.

ACM has designed its own computer system that can make some field computations and produce results in the form of mathematic expressions. Unfortunately, by generating the expression in several steps, there are always some unneeded parentheses inside the expression. Your task is to take these partial results and make them "nice" by removing all unnecessary parentheses.

Input

There is a single positive integer T on the first line of input (equal to about 10000). It stands for the number of expressions to follow. Each expression consists of a single line containing only lowercase letters, operators (+, -, *, /) and parentheses ((and)). The letters are variables that can have any value, operators and parentheses have their usual meaning. Multiplication and division have higher priority than subtraction and addition. All operations with the same priority are computed from left to right (operators are left-associative). There are no spaces inside the expressions. No input line contains more than 250 characters.

Output

Print a single line for every expression. The line must contain the same expression with unneeded parentheses removed. You must remove as many parentheses as possible without changing the semantics of the expression. The semantics of the expression is considered the same if and only if any of the following conditions hold:

- The ordering of operations remains the same. That means " $(a+b)+c$ " is the same as " $a+b+c$ ", and " $a+(b/c)$ " is the same as " $a+b/c$ ".
- The order of some operations is swapped but the result remains unchanged with respect to the addition and multiplication associativity. That means " $a+(b+c)$ " and " $(a+b)+c$ " are the same. We can also combine addition with subtraction and multiplication with division, if the subtraction or division is the second operation. For example, " $a+(b-c)$ " is the same as " $a+b-c$ ".

You cannot use any other laws, namely you cannot swap left and right operands and you cannot replace " $a-(b-c)$ " with " $a-b+c$ ".

Example

Sample Input:

```
8
(a+(b*c))
((a+b)*c)
(a*(b*c))
(a*(b/c)*d)
((a/(b/c))/d)
(x)
(a+b)-(c-d)-(e/f)
(a+b)+(c-d)-(e+f)
```

Sample Output:

```
a+b*c
(a+b)*c
a*b*c
a*b/c*d
a/(b/c)/d
x
a+b-(c-d)-e/f
a+b+c-d-(e+f)
```

Added by: Adrian Kosowski

Date: 2004-05-09

Time limit [s]: 10

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 2000

SPOJ Problem Set

11. Factorial

Problem code: FCTRL

The most important part of a GSM network is so called *Base Transceiver Station (BTS)*. These transceivers form the areas called *cells* (this term gave the name to the cellular phone) and every phone connects to the BTS with the strongest signal (in a little simplified view). Of course, BTSes need some attention and technicians need to check their function periodically.

ACM technicians faced a very interesting problem recently. Given a set of BTSes to visit, they needed to find the shortest path to visit all of the given points and return back to the central company building. Programmers have spent several months studying this problem but with no results. They were unable to find the solution fast enough. After a long time, one of the programmers found this problem in a conference article. Unfortunately, he found that the problem is so called "Travelling Salesman Problem" and it is very hard to solve. If we have N BTSes to be visited, we can visit them in any order, giving us $N!$ possibilities to examine. The function expressing that number is called factorial and can be computed as a product $1.2.3.4....N$. The number is very high even for a relatively small N .

The programmers understood they had no chance to solve the problem. But because they have already received the research grant from the government, they needed to continue with their studies and produce at least *some* results. So they started to study behaviour of the factorial function.

For example, they defined the function Z . For any positive integer N , $Z(N)$ is the number of zeros at the end of the decimal form of number $N!$. They noticed that this function never decreases. If we have two numbers $N_1 < N_2$, then $Z(N_1) \leq Z(N_2)$. It is because we can never "lose" any trailing zero by multiplying by any positive number. We can only get new and new zeros. The function Z is very interesting, so we need a computer program that can determine its value efficiently.

Input

There is a single positive integer T on the first line of input (equal to about 100000). It stands for the number of numbers to follow. Then there are T lines, each containing exactly one positive integer number N , $1 \leq N \leq 1000000000$.

Output

For every number N , output a single line containing the single non-negative integer $Z(N)$.

Example

Sample Input:

6
3
60
100
1024
23456
8735373

Sample Output:

0
14
24
253
5861
2183837

Added by: Adrian Kosowski
Date: 2004-05-09
Time limit [s]: 15
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Prague 2000

SPOJ Problem Set

12. The Game of Master-Mind

Problem code: MMIND

If you want to buy a new cellular phone, there are many various types to choose from. To decide which one is the best for you, you have to consider several important things: its size and weight, battery capacity, WAP support, colour, price. One of the most important things is also the list of games the phone provides. Nokia is one of the most successful phone makers because of its famous Snake and Snake II. ACM wants to make and sell its own phone and they need to program several games for it. One of them is Master-Mind, the famous board logical game.

The game is played between two players. One of them chooses a *secret code* consisting of P ordered pins, each of them having one of the predefined set of C colours. The goal of the second player is to guess that secret sequence of colours. Some colours may not appear in the code, some colours may appear more than once.

The player makes guesses, which are formed in the same way as the secret code. After each guess, he/she is provided with an information on how successful the guess was. This feedback is called a *hint*. Each hint consists of B black points and W white points. The black point stands for every pin that was guessed right, i.e. the right colour was put on the right position. The white point means right colour but on the wrong position. For example, if the secret code is "white, yellow, red, blue, white" and the guess was "white, red, white, white, blue", the hint would consist of one black point (for the white on the first position) and three white points (for the other white, red and blue colours). The goal is to guess the sequence with the minimal number of hints.

The new ACM phone should have the possibility to play both roles. It can make the secret code and give hints, but it can also make its own guesses. Your goal is to write a program for the latter case, that means a program that makes Master-Mind guesses.

Input

There is a single positive integer T on the first line of input. It stands for the number of test cases to follow. Each test case describes one game situation and you are to make a guess. On the first line of each test case, there are three integer numbers, P , C and M . P ($1 \leq P \leq 10$) is the number of pins, C ($1 \leq C \leq 100$) is the number of colours, and M ($1 \leq M \leq 100$) is the number of already played guesses.

Then there are $2 \times M$ lines, two lines for every guess. At the first line of each guess, there are P integer numbers representing colours of the guess. Each colour is represented by a number G_i , $1 \leq G_i \leq C$. The second line contains two integer numbers, B and W , stating for the number of black and white points given by the corresponding hint.

Let's have a secret code S_1, S_2, \dots, S_P and the guess G_1, G_2, \dots, G_P . Then we can make a set H containing pairs of numbers (I, J) such that $S_I = G_J$, and that any number can appear at most once on the first position and at most once on the second position. That means for every two different pairs from that set, (I_1, J_1) and (I_2, J_2) , we have $I_1 \neq I_2$ and $J_1 \neq J_2$. Then we denote $B(H)$ the number of pairs in the set, that meet the condition $I = J$, and $W(H)$ the number of pairs with $I \neq J$.

We define an ordering of every two possible sets H_1 and H_2 . Let's say $H_1 \leq H_2$ if and only if one of the following holds:

- $B(H_1) < B(H_2)$, or
- $B(H_1) = B(H_2)$ and $W(H_1) \leq W(H_2)$

Then we can find a maximal set H_{max} according to this ordering. The numbers $B(H_{max})$ and $W(H_{max})$ are the black and white points for that hint.

Output

For every test case, print the line containing P numbers representing P colours of the next guess. Your guess must be valid according to all previous guesses and hints. The guess is valid if the sequence could be a secret code, i.e. the sequence was not eliminated by previous guesses and hints.

If there is no valid guess possible, output the sentence `You are cheating!`. If there are more valid guesses, output the one that is lexicographically smallest. I.e. find such guess G that for every other valid guess V there exists such a number I that:

- $G_J = V_J$ for every $J < I$, and
- $G_I < V_I$.

Example

Sample Input:

```
3
4 3 2
1 2 3 2
1 1
2 1 3 2
1 1
4 6 2
3 3 3 3
3 0
4 4 4 4
2 0
8 9 3
1 2 3 4 5 6 7 8
0 0
2 3 4 5 6 7 8 9
1 0
3 4 5 6 7 8 9 9
2 0
```

Sample Output

```
1 1 1 3
You are cheating!
9 9 9 9 9 9 9 9
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski
Date: 2004-05-09
Time limit [s]: 15
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Prague 2000

SPOJ Problem Set

13. Hotline

Problem code: **HOTLINE**

Every customer sometimes needs help with new and unusual products. Therefore, hotline service is very important for every company. We need a single phone number where the customer can always find a friendly voice ready to help with anything. On the other hand, many people are needed to serve as hotline operators, and human resources are always very expensive. Moreover, it is not easy to pretend "friendly voice" at 4am and explain to a drunken man that you are really unable to give him the number to House of Parliament. It was also found that some of the questions repeat very often and it is very annoying to answer them again and again.

ACM is a modern company, wanting to solve its hotline problem. They want to decrease the number of human operators by creating a complex software system that would be able to answer most common questions. The customer's voice is analysed by a special Voice Recognition Module (VRM) and converted to a plain text. The text is then taken by an Artificial Automatic Adaptive Answering Algorithm (AAAAA). The most common questions are recognised and answered automatically. The replies are then converted to a sound by Text-to-Speech Module (TTS).

You are to write the AAAAA module. Because your algorithm should be adaptive, it has no explicit knowledge base. But it must be able to listen to sentences in English and remember the mentioned facts. Whenever the question is asked about such a fact, the system has to answer it properly. The VRM and TTS modules are already implemented, so the input and output of AAAAA will be in the text form.

Input

There is a single positive integer T on the first line of input. It stands for the number of dialogues to follow. Each dialogue consists of zero or more lines, each of them containing one sentence: either statement or question. The statement ends with a dot character (.), the question ends with a question mark (?). No statement will appear more than once, however the questions can be repeated. There is one extra line after each dialogue. That line ends with an exclamation mark (!).

Sentences can contain words, spaces and punctuation characters (such as commas, colons, semicolons etc.). All words contain only letters of English alphabet and are case-sensitive. That means the same word is always written the same way, usually in lowercase. Acronyms, names and some other words can begin with capital letters. For simplicity, all sentences begin with a lowercase letter. Only if the first word should be written with a capital, the sentence begins with a capital letter. There are no unneeded spaces between words. No line will have more than 100 characters. There will be at most 100 statements per each test case.

Statements

Each statement has one of the following two forms (_ denotes a space):

subject *_predicate*[s] [*_object*] .

subject *_don't|doesn't* *_predicate* [*_object*] .

The square brackets mark an optional part, the vertical line two possible variants. Subject is a single word, noun or pronoun in singular. Predicate is a verb (single word) denoting some activity. Object can be any text. Object does not contain any dots. Any pair "verb + object" determines unique activity. The same verb with different objects makes different independent activities, i.e. the different and independent meaning of the sentence. Sentence without any object can be considered as sentence with an empty object. The verb without an object has different and independent meaning than the same verb with any non-empty object.

The first variant of sentence denotes a positive statement. The word "*predicate[s]*" means verb that matches the subject of the sentence. If the subject is "I" or "you", the verb has the same form as the infinitive. With any other subject, the letter "s" is appended on the end of the verb. Assume there are no irregular verbs.

The second variant is a negative statement. Verb "don't" or "doesn't" must also match the subject. The form "don't" is used with either "I" or "you", "doesn't" is used in any other case.

A special generic subject "everybody" can be used. It means the activity holds for any subject. Other special subject is "nobody". Such sentence also holds for any subject, but its meaning is negative. Both of these generic subjects can be used with the first variant only (without "doesn't"). The sentence "nobody likes something" is exactly equal to "everybody doesn't like something", except the latter form will never occur in the input.

Questions

Each question has one of the following three forms:

1. do|does *_subject _predicate* [*_object*] ?
2. who *_predicates* [*_object*] ?
3. what *_do|does _subject* do ?

The word "do|does" always matches the subject ("do I?", "do you?", "does any other subject?"). In the second type of question, predicate always matches the word "who", i.e. the "s" is always appended. Generic subjects cannot be used in questions.

Output

For each dialogue, your program must output the line Dialogue #*D*:, where *D* is the sequence number of dialogue, starting with 1. Then print exactly three lines for every question: the first line repeats the question, the second line contains the answer, and the third line is empty. Print nothing for statements. After each dialogue, print the same line with an exclamation mark that was in the input. Then print one extra empty line. Empty line contains a new-line character only.

The answer must be properly formatted to be accepted by a TTS module. Only the statements appearing in the input before the answer are used for the corresponding reply. If there is any contradiction among statements, the reply is always I am abroad.. If the question and statements consider the special subject "you", it must be replaced with "I" in the answer. If the question considers special subject "I", it must be replaced with "you" in the answer. The verb must always match the subject of the sentence. The exact form of the correct answer depends on the type of question.

1. does subject predicate [object] ?

If there is any positive statement about the mentioned subject (or generic subject "everybody"), predicate and object, the answer is:

yes , _subject _predicate[s] [_object] .

If there is any negative statement about the mentioned subject (or generic subject "nobody"), predicate and object, the answer is:

no , _subject _don't|doesn't _predicate [_object] .

Otherwise, the answer is: maybe .

Subject in the answer is always the same subject as the subject of the question.

2. who predicates [object] ?

If there is a positive statement considering any subject, the specified predicate and object, the answer is:

subject _predicate[s] [_object] .

If two or more subjects match the activity, replace the subject in the answer with enumeration of all such subjects, in the same order as the corresponding statements have appeared in the input. Subjects are separated with comma and space, last two subjects are separated with the word "and". If "everybody" belongs to the group of enumerated subjects, do not enumerate subjects, and print "everybody" only. If the enumeration contains at least two subjects, the predicate matches the plural subject (i.e. verb is without trailing "s"), otherwise it matches the only subject.

subject1 , _subject2 _and _subject3 predicate [_object] .

If there is a negative statement considering the generic subject "nobody", the specified predicate and object, the answer is:

nobody _predicates [_object] .

Otherwise, the answer is: I don't know .

3. what does subject do ?

If there are one or more sentences (both positive and negative) considering the specified subject (or a generic subject "everybody" or "nobody"), all verbs and objects from such sentences must be included in a reply in the same order as the corresponding sentences have appeared in the input. No verb-object pair can be included more than once (the eventual second appearance must be skipped). The verb-object pairs are separated by a comma followed by a space, the last verb is separated by a comma and the word "and". Please note the comma is printed here although there was no comma when separating the subjects in the previous type of answer (see above). The negative answers have the same form as the statements, that means the verb "don't" or "doesn't" is used:

subject [_don't|doesn't] _predicate1[s] [_object1] ,

[_don't|doesn't] _predicate2[s] [_object2] ,

_and [_don't|doesn't] _predicate3[s] [_object3] .

subject [_don't|doesn't] _predicate1[s] [_object1] ,

_and [_don't|doesn't] _predicate2[s] [_object2] .

subject [_don't|doesn't] _predicate[s] [_object] .

Otherwise, the answer is: I don't know.

Example

Sample Input:

```
1
I like hotdogs.
nobody likes to work.
everybody smiles.
what do I do?
who smiles?
what do you do?
does Joe smile?
do I like to work?
everybody hurts sometimes.
who walks there?
Michal walks there.
who walks there?
what does Michal do?
do you understand?
nobody walks there.
do you understand now?
bye!
```

Sample Output:

```
Dialogue #1:
what do I do?
you like hotdogs, don't like to work, and smile.

who smiles?
everybody smiles.

what do you do?
I don't like to work, and smile.

does Joe smile?
yes, Joe smiles.

do I like to work?
no, you don't like to work.

who walks there?
I don't know.

who walks there?
Michal walks there.

what does Michal do?
Michal doesn't like to work, smiles, hurts sometimes, and walks there.

do you understand?
maybe.

do you understand now?
I am abroad.

bye!
```

Added by: Adrian Kosowski
Date: 2004-05-09
Time limit [s]: 3
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Prague 2000

SPOJ Problem Set

14. I-Keyboard

Problem code: IKEYB

Most of you have probably tried to type an SMS message on the keypad of a cellular phone. It is sometimes very annoying to write longer messages, because one key must be usually pressed several times to produce a single letter. It is due to a low number of keys on the keypad. Typical phone has twelve keys only (and maybe some other control keys that are not used for typing). Moreover, only eight keys are used for typing 26 letters of an English alphabet. The standard assignment of letters on the keypad is shown in the left picture:

1	2 abc	3 def
4 ghi	5 jkl	6 mno
7 pqrs	8 tuv	9 wxyz
*	0 <i>space</i>	#

1	2 abcd	3 efg
4 hijk	5 lm	6 nopq
7 rs	8 tuv	9 wxyz
*	0 <i>space</i>	#

There are 3 or 4 letters assigned to each key. If you want the first letter of any group, you press that key once. If you want the second letter, you have to press the key twice. For other letters, the key must be pressed three or four times. The authors of the keyboard did not try to optimise the layout for minimal number of keystrokes. Instead, they preferred the even distribution of letters among the keys. Unfortunately, some letters are more frequent than others. Some of these frequent letters are placed on the third or even fourth place on the standard keyboard. For example, S is a very common letter in an English alphabet, and we need four keystrokes to type it. If the assignment of characters was like in the right picture, the keyboard would be much more comfortable for typing average English texts.

ACM have decided to put an optimised version of the keyboard on its new cellular phone. Now they need a computer program that will find an optimal layout for the given letter frequency. We need to preserve alphabetical order of letters, because the user would be confused if the letters were mixed. But we can assign any number of letters to a single key.

Input

There is a single positive integer T on the first line of input (equal to about 2000). It stands for the number of test cases to follow. Each test case begins with a line containing two integers K, L ($1 \leq K \leq L \leq 90$) separated by a single space. K is the number of keys, L is the number of letters to be mapped onto those keys. Then there are two lines. The first one contains exactly K characters each representing a name of one key. The second line contains exactly L characters representing names of letters of an alphabet. Keys and letters are represented by digits, letters (which are case-sensitive), or any punctuation characters (ASCII code between 33 and 126 inclusively). No two keys have the same character, no two letters are the same. However, the name of a letter can be used also as a name for a key.

After those two lines, there are exactly L lines each containing exactly one positive integer F_1, F_2, \dots, F_L . These numbers determine the frequency of every letter, starting with the first one and continuing with the others sequentially. The higher number means the more common letter. No frequency will be higher than 100000.

Output

Find an optimal keyboard for each test case. Optimal keyboard is such that has the lowest "price" for typing average text. The *price* is determined as the sum of the prices of each letter. The price of a letter is a product of the letter frequency (F_i) and its position on the key. The order of letters cannot be changed, they must be grouped in the given order.

If there are more solutions with the same price, we will try to maximise the number of letters assigned to the last key, then to the one before the last one etc.

More formally, you are to find a sequence P_1, P_2, \dots, P_L representing the position of every letter on a particular key. The sequence must meet following conditions:

- $P_1 = 1$
- for each $i > 1$, either $P_i = P_{i-1} + 1$ or $P_i = 1$
- there are at most K numbers P_i such that $P_i = 1$
- the sum of products $S_P = \text{Sum}[i=1..L] F_i \cdot P_i$ is minimal
- for any other sequence Q meeting these criteria and with the same sum $S_Q = S_P$, there exists such M , $1 \leq M \leq L$ that for any J , $M < J \leq L$, $P_J = Q_J$, and $P_M > Q_M$.

The output for every test case must start with a single line saying Keypad # I :, where I is a sequential order of the test case, starting with 1. Then there must be exactly K lines, each representing one letter, in the same order that was used in input. Each line must contain the character representing the key, a colon, one space and a list of letters assigned to that particular key. Letters are not separated from each other.

Print one blank line after each test case, including the last one.

Example

Sample Input:

```
1
8 26
23456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
3371
589
1575
1614
6212
971
773
1904
2989
123
209
```

1588
1513
2996
3269
1080
121
2726
3083
4368
1334
518
752
427
733
871

Sample Output:

Keypad #1:
2: ABCD
3: EFG
4: HIJK
5: LM
6: NOPQ
7: RS
8: TUV
9: WXYZ

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski
Date: 2004-05-09
Time limit [s]: 10
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Prague 2000

SPOJ Problem Set

15. The Shortest Path

Problem code: SHPATH

Given a list of cities. Each direct connection between two cities has its transportation cost (an integer bigger than 0). The goal is to find the paths of minimum cost between pairs of cities. Assume that the cost of each path (which is the sum of costs of all direct connections belonging to this path) is at most 200000. The name of a city is a string containing characters a,...,z and is at most 10 characters long.

Input

```
s [the number of tests <= 10]
n [the number of cities <= 10000]
NAME [city name]
p [the number of neighbours of city NAME]
nr cost [nr - index of a city connected to NAME (the index of the first city is 1)]
        [cost - the transportation cost]
r [the number of paths to find <= 100]
NAME1 NAME2 [NAME1 - source, NAME2 - destination]
[empty line separating the tests]
```

Output

```
cost [the minimum transportation cost from city NAME1 to city NAME2 (one per line)]
```

Example

```
Input:
1
4
gdansk
2
2 1
3 3
bydgoszcz
3
1 1
3 1
4 4
torun
3
1 3
2 1
4 1
warszawa
2
2 4
3 1
2
gdansk warszawa
bydgoszcz warszawa
```


Output:

3

2

Warning: large Input/Output data, be careful with certain languages

Added by: Darek Dereniowski

Date: 2004-05-10

Time limit [s]: 10

Source limit [B]:50000

Resource: DASM Programming League 2003 (problemset 11)

SPOJ Problem Set

16. Sphere in a tetrahedron

Problem code: TETRA

Of course a Sphere Online Judge System is bound to have some tasks about spheres. So here is one. Given the lengths of the edges of a tetrahedron calculate the radius of a sphere inscribed in that tetrahedron (i.e. a sphere tangent to all the faces).

Input

Number N of test cases in a single line. ($N \leq 30$) Each of the next N lines consists of 6 integer numbers -- the lengths of the edges of a tetrahedron separated by single spaces. The edges are not longer than 1000 and for the tetrahedron WXYZ, the order of the edges is: WX, WY, WZ, XY, XZ, YZ.

Output

N lines, each consisting of a real number given with four digits decimal precision equal to the radius of a sphere inscribed in the given tetrahedron.

Example

```
Input :
2
1 1 1 1 1 1
1000 999 998 5 5 6
```

```
Output:
0.2041
1.4189
```

Added by: Adam Dzedzej
Date: 2004-05-11
Time limit [s]: 1
Source limit [B]:50000

SPOJ Problem Set

17. The Bytelandian Cryptographer (Act I)

Problem code: CRYPTO1

The infamous Bytelandian Bit-eating Fanatic Organisation (BBFO for short) plans to launch an all-out denial-of-service attack on the Bytelandian McDecimal's fast food network by blocking the entrance to every restaurant with a camel (the purpose being to rid the Organisation of unhealthy competition, obviously). In a sly and perfidious move, the head cryptographer of BBFO decided to disclose the date and time of the planned attack to the management of McDecimal's, but in encrypted form (ha ha). He calculated the number of seconds from midnight 1970.01.01 GMT to the moment of attack, squared it, divided it by 4000000007 and sent the remainder by e-mail to McDecimal's. This made the original date impossible to decode.

Or did it?

* * *

You work as the head algorithmist at McDecimal's HQ and know nothing of what is happening in Byteland. Things are not going well. You are playing a quiet game of hearts against your computer and wondering why on earth Management are considering making you redundant. Suddenly, the CEO bursts into your office, saying:

- Look here, young man[lady]! I have this number and those guys claim it is supposed to be some date. I am giving you one second to tell me what it all means!

I am afraid you have no choice. You can't ask any further questions.
You just have to answer, now.

Input

The encrypted timestamp.

Output

The decrypted GMT time and date of attack, somewhere between 1970 and 2030, using standard 26 character formatting.

Example

Input:

1749870067

Output:

Sun Jun 13 16:20:39 2004

Added by: Adrian Kosowski
Date: 2004-05-13
Time limit [s]: 1
Source limit [B]:10000
Resource: ;)

SPOJ Problem Set

18. The Bytelandian Cryptographer (Act II)

Problem code: CRYPTO2

Encouraged by his last successful exploit, the Bytelandian fanatic cryptographer impudently encrypted a three-digit number by subtracting 1 from it.

This time he has **really** overstepped the mark! Soldier, go and beat him, for Burger King & Country! Oh, and remember your good manners, use Brainf**k (no other language is allowed).

Input

An encrypted 3-digit positive integer.

Output

The decrypted value.

Example

Input:

699

Output:

700

Added by: Adrian Kosowski

Date: 2004-05-28

Time limit [s]: 1

Source limit [B]:50000

Resource: Sometimes the simplest language is the most pleasing.

SPOJ Problem Set

19. The Bytelandian Cryptographer (Act III)

Problem code: CRYPTO3

The Bytelandian cryptographer acknowledged he was sorely beaten in Act 2. He renounced his own methods of encryption and decided to return to the classic techniques.

Not knowing what to do next, he went to the cinema to chew the problem over. To his surprise, he found that the cone containing pop-corn was in fact a rolled up page torn from the classic book, *RSA for newbies in 24 seconds*. The page in question contained the entire key-generating and encryption algorithm. Fascinated, he thought up two different prime numbers p and q , and calculated his own public key, and revealed the product $p*q$ to the wide world. Then, he began work on his wicked scheme of encryption.

History repeats. Once more, you receive an encrypted message from the cryptographer. This time you know that without additional information you are beaten, so you decide to use the psychological approach. You phone the Bytelandian cryptographer, and ask him whether he couldn't give you a little hint. What you really want to know is the number u of positive integers which are smaller than $p*q$ and have no common factors with $p*q$ other than 1. But the cryptographer, sensing that this would allow you to decode the message right away, refuses to tell you this number. Eventually, after a lot of asking, he gives you a piece of utterly useless information: he tells you how many positive integers x cannot be represented in the form $x=a*p+b*q$, regardless of what non-negative integer values a and b assume.

You begin to wonder whether the information you received from the cryptographer is not by any chance enough to find the value of u .

Even if the only languages at your disposal are Brainfk and Intercal...**

Input

The number provided by the cryptographer (a positive integer of at most 99 decimal digits). The input ends with a new line symbol.

Output

The value of u .

Example

Input :

1

Output :

2

(This example is possible for $p=2$, $q=3$)

Added by: Adrian Kosowski
Date: 2004-05-29
Time limit [s]: 5
Source limit
[B]: 50000
Resource: Sadly, the ability to make a simple problem difficult to understand is seldom considered a talent.

SPOJ Problem Set

20. The Bytelandian Cryptographer (Act IV)

Problem code: CRYPTO4

The Bytelandian Cryptographer has been requested by the BBFO to put forward an encryption scheme which would allow the BBFO to communicate with its foreign associates. After some intensive studies, he has decided upon the Vigenère cipher. Messages written using 26 upper case characters of the Latin alphabet: A, B, ..., Z which are interpreted as integers 0,1, ..., 25 respectively. The secret cypher for transmitting a message is known to both sides and consists of n integers k_1, k_2, \dots, k_n . Using this cypher, the i -th number x_i of the input message x is encrypted to the form of the i -th number of the output message y , as follows:

$$y_i = (x_i + k_{1 + ((i-1) \bmod n)}) \bmod 26.$$

You are trying to find out the content of a message transmitted by the BBFO. By a lucky stroke of fortune, your spys managed to intercept the message in both its plaintext and encrypted form (x and y respectively). Unfortunately, during their dramatic escape the files they were carrying were pierced by bullets and fragments of messages x and y were inadvertently lost. Or were they? It is your task to reconstruct as much of message x as you possibly can.

Input

The first line of input contains a single integer $t \leq 200$ denoting the number of test cases. t test case descriptions follow.

For each test case, the first line contains one integer m which is some upper bound on the length of the cypher ($1 \leq n \leq m \leq 100000$). The second line of input contains the original message x , while the third line contains the encrypted message y . The messages are expressed using characters 'A'-'Z' (interpreted as integers 0-25) and '*' (denoting a single character illegible due to damage). The total length of the input file is not more than 2MB.

Output

For each test case output a single line containing the original message x , with asterisks '*' in place of only those characters whose value cannot be determined.

Example

Input :

```
4
1
A*X*C
**CM*
4
*B***A
AAAAAA
6
*B***A
AAAAAA
4
```


*AA*****
AAAAAAAAAA

Output:

A*XHC
*BA*BA
*B***A
*AA**A****

**Warning: large Input/Output data, be careful with certain languages.
The time limit is strict for this problem.**

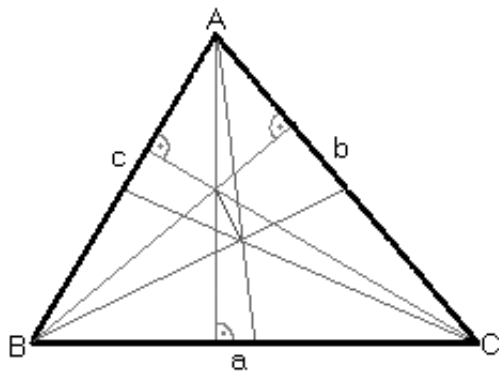
Added by: Konrad Piwakowski
Date: 2004-11-16
Time limit [s]: 42
Source limit [B]: 50000
Resource: DASM Programming League 2004 (problemset 3)

SPOJ Problem Set

22. Triangle From Centroid

Problem code: TRICENTR

Given the length of side a of a triangle and the distances from the centroid (the point of concurrence of the medians - red in the picture) to all sides: a , b and c , calculate this triangle's area and the distance (blue line) from the orthocenter (the point of concurrence of the heights - green in the picture) to the centroid.



Input

In the first line integer n - the number of test cases (equal to about 1000). The next n lines - 4 floating point values: the length of side a , and distances from the centroid to sides a , b and c .

Output

n lines consisting of 2 floating point values with 3 digits after the decimal point: the area of the triangle and the distance from the orthocenter to centroid.

Example

Input :

2

3.0 0.8660254038 0.8660254038 0.8660254038
657.8256599140 151.6154399062 213.5392629932 139.4878846649

Output :

3.897 0.000
149604.790 150.275

Added by: Patryk Pomykalski
Date: 2004-05-22
Time limit [s]: 1
Source limit [B]:50000

SPOJ Problem Set

23. Pyramids

Problem code: PIR

Recently in Farland, a country in Asia, the famous scientist Mr. Log Archeo discovered ancient pyramids. But unlike those in Egypt and Central America, they have a triangular (not rectangular) foundation. That is, they are tetrahedrons in the mathematical sense. In order to find out some important facts about the early society of the country (it is widely believed that the pyramid sizes are closely connected with Farland's ancient calendar), Mr. Archeo needs to know the volume of the pyramids. Unluckily, he has reliable data about their edge lengths only. Please, help him!

Input

t [number of tests to follow] In each of the next t lines six positive integer numbers not exceeding 1000 separated by spaces (each number is one of the edge lengths of the pyramid ABCD). The order of the edges is the following: AB, AC, AD, BC, BD, CD.

Output

For each test output a real number - the volume, printed accurate to four digits after decimal point.

Example

```
Input :
2
1 1 1 1 1 1
1000 1000 1000 3 4 5
```

```
Output :
0.1179
1999.9937
```

Added by: Adam Dzedzej
Date: 2004-05-14
Time limit [s]: 1
Source limit [B]: 10000
Resource: ACM ICPC 2002-2003 NEERC, Northern Subregion

SPOJ Problem Set

24. Small factorials

Problem code: FCTRL2

You are asked to calculate factorials of some small positive integers.

Input

An integer t , $1 \leq t \leq 100$, denoting the number of testcases, followed by t lines, each containing a single integer n , $1 \leq n \leq 100$.

Output

For each integer n given at input, display a line with the value of $n!$

Example

Sample input:

```
4
1
2
5
3
```

Sample output:

```
1
2
120
6
```

Added by: Adrian Kosowski

Date: 2004-05-28

Time limit [s]: 1

Source limit [B]: 2000

SPOJ Problem Set

25. Pouring water

Problem code: POUR1

Given two vessels, one of which can accommodate a litres of water and the other - b litres of water, determine the number of steps required to obtain exactly c litres of water in one of the vessels.

At the beginning both vessels are empty. The following operations are counted as 'steps':

- emptying a vessel,
- filling a vessel,
- pouring water from one vessel to the other, without spilling, until one of the vessels is either full or empty.

Input

An integer t , $1 \leq t \leq 100$, denoting the number of testcases, followed by t sets of input data, each consisting of three positive integers a , b , c , not larger than 40000, given in separate lines.

Output

For each set of input data, output the minimum number of steps required to obtain c litres, or -1 if this is impossible.

Example

Sample input:

```
2
5
2
3
2
3
4
```

Sample output:

```
2
-1
```

Added by: Adrian Kosowski

Date: 2004-05-31

Time limit [s]: 1

Source limit [B]: 50000

Resource: An ancient problem, formulated in these words by Mr Tadeusz Ratajczak

SPOJ Problem Set

26. Build the Fence

Problem code: BSHEEP

At the beginning of spring all the sheep move to the higher pastures in the mountains. If there are thousands of them, it is well worthwhile gathering them together in one place. But sheep don't like to leave their grass-lands. Help the shepherd and build him a fence which would surround all the sheep. The fence should have the smallest possible length! Assume that sheep are negligibly small and that they are not moving. Sometimes a few sheep are standing in the same place. If there is only one sheep, it is probably dying, so no fence is needed at all...

Input

t [the number of tests ≤ 100]

[empty line]

n [the number of sheep ≤ 100000]

x_1 y_1 [coordinates of the first sheep]

...

x_n y_n

[integer coordinates from -10000 to 10000]

[empty line]

[other lists of sheep]

Text grouped in [] does not appear in the input file. Assume that sheep are numbered in the input order.

Output

o [length of circumference, 2 digits precision]

p1 p2 ... pk

[the sheep that are standing in the corners of the fence; the first one should be positioned bottommost and as far to the left as possible, the others ought to be written in anticlockwise order; ignore all sheep standing in the same place but the first to appear in the input file; the number of sheep should be the smallest possible]

[empty line]

[next solutions]

Example

Input :

8

5

0 0

0 5

10 5

3 3

10 0

1
0 0

3
0 0
1 0
2 0

4
0 0
0 0
0 1
1 0

3
0 0
0 1
1 0

6
0 0
-1 -1
1 1
2 2
3 3
4 4

2
10 0
0 0

7
-3 -4
2 -3
4 3
-4 2
0 5
2 -3
-1 4

Output:
30.00
1 5 3 2

0.00
1

4.00
1 3

3.41
1 4 3

3.41
1 3 2

14.14
2 6

20.00

2 1

26.98

1 2 3 5 4

Warning: large Input/Output data, be careful with certain languages

Added by: Michal Malafiejski

Date: 2004-06-01

Time limit [s]: 17

Source limit [B]:50000

Resource: PAL

SPOJ Problem Set

27. Sorting Bank Accounts

Problem code: SBANK

In one of the internet banks thousands of operations are being performed every day. Since certain customers do business more actively than others, some of the bank accounts occur many times in the list of operations. Your task is to sort the bank account numbers in ascending order. If an account appears twice or more in the list, write the number of repetitions just after the account number. The format of accounts is as follows: **2** control digits, an **8**-digit code of the bank, **16** digits identifying the owner (written in groups of four digits), for example 30 10103538 2222 1233 6160 0142.

Banks are real-time institutions and they need FAST solutions. If you feel you can meet the challenge within a very stringent time limit, go ahead! A well designed sorting algorithm in a fast language is likely to succeed.

Input

t [the number of tests ≤ 5]
 n [the number of accounts $\leq 100\,000$]
[list of accounts]
[empty line]
[next test cases]

Output

[sorted list of accounts with the number of repeated accounts]
[empty line]
[other results]

Example

Input :

2

6

03 10103538 2222 1233 6160 0142
03 10103538 2222 1233 6160 0141
30 10103538 2222 1233 6160 0141
30 10103538 2222 1233 6160 0142
30 10103538 2222 1233 6160 0141
30 10103538 2222 1233 6160 0142

5

30 10103538 2222 1233 6160 0144
30 10103538 2222 1233 6160 0142
30 10103538 2222 1233 6160 0145
30 10103538 2222 1233 6160 0146
30 10103538 2222 1233 6160 0143

Output :

03 10103538 2222 1233 6160 0141 1

03 10103538 2222 1233 6160 0142 1
30 10103538 2222 1233 6160 0141 2
30 10103538 2222 1233 6160 0142 2

30 10103538 2222 1233 6160 0142 1
30 10103538 2222 1233 6160 0143 1
30 10103538 2222 1233 6160 0144 1
30 10103538 2222 1233 6160 0145 1
30 10103538 2222 1233 6160 0146 1

Added by: Michal Malafiejski
Date: 2004-06-01
Time limit [s]: 17
Source limit [B]:50000
Resource: PAL

SPOJ Problem Set

28. Help the Military Recruitment Office!

Problem code: HMRO

At the end of year 2004, the regional agencies of the Polish Military Recruitment Office (known as WKU in Polish) is sending a call to all boys born in 1984. Every recruit has his personal 11-digit identification number (PESEL, format: YYMMDDXXXXX, where YYMMDD is the date of birth, and XXXXX is a zero-padded integer smaller than 100000). Every agency of the Military Recruitment Office has its own code (MRO, format: a place code consisting of 3 upper case letters and a one-digit number). But this year the army underwent some reforms and not all boys at conscription age are going to be recruited. The list of closed down MRO points is as follows: the code of the closed down MRO is followed by the code of some other MRO, to which all the recruits are now going to be assigned. The list of recruits contains their PESEL codes. Your task is to prepare the complete list of recruits and determine the codes of their new MRO-s.

Input

```
s [the number of tests <= 10]
p [the number of boys at conscription age <= 100000]
PESEL and MRO code
z [the number of closed down MRO points <= 100000]
old_code new_code [old_code - the code of closed down MRO,
new_code - its new MRO code]
p [the number of recruits <= 100000]
PESEL [PESEL code of recruit]
[empty line]
[next tests]
```

Output

```
one PESEL and MRO code per line in the order of input
[empty line between tests]
[other results]
```

Example

```
Input:
1
4
84101011111 GDA1
84010122222 GDA2
84010233333 GDA2
84020255555 GDY1
1
GDA2 GDA1
3
84101011111
84010122222
84020255555
```

Output:
84101011111 GDA1
84010122222 GDA1
84020255555 GDY1

Warning: large Input/Output data, be careful with certain languages

Added by: Michal Malafiejski
Date: 2004-06-01
Time limit [s]: 11
Source limit [B]:50000
Resource: PAL (thanks to Adam Nadolski)

SPOJ Problem Set

29. Hash it!

Problem code: HASHIT

Your task is to calculate the result of the hashing process in a table of 101 elements, containing keys that are strings of length at most 15 letters (ASCII codes 'A',..., 'z'). Implement the following operations:

- find the index of the element defined by the key (ignore, if no such element),
- insert a new key into the table (ignore insertion of the key that already exists),
- delete a key from the table (without moving the others), by marking the position in table as *empty* (ignore non-existing keys in the table)

When performing find, insert and delete operations define the following function:

integer Hash(string key),

which for a string $key = a_1 \dots a_n$ returns the value:

$Hash(key) = h(key) \bmod 101$, where

$h(key) = 19 * (ASCII(a_1) * 1 + \dots + ASCII(a_n) * n)$.

Resolve collisions using the open addressing method, i.e. try to insert the key into the table at the first free position: $(Hash(key) + j^2 + 23*j) \bmod 101$, for $j = 1, \dots, 19$. After examining of at least 20 table entries, we assume that the insert operation cannot be performed.

Input

t [the number of test cases ≤ 100]

n_1 [the number of operations (one per line) ≤ 1000]

ADD:string

[or]

DEL:string [other test cases, without empty lines between series]

Output

For every test case you have to create a new table, insert or delete keys, and write to the output:

the number of keys in the table [first line]

index:key [sorted by indices]

Example

Input :

2

11

ADD:marsz

ADD:marsz

ADD:Dabrowski

ADD:z

ADD:ziemii

ADD:wloskiej

ADD:do

ADD:Polski
DEL:od
DEL:do
DEL:wloskiej
8
ADD:my
ADD:favourite
ADD:online
ADD:judge
ADD:sphere!
DEL:judge
DEL:my
DEL:my

Output:
5
34:Dabrowski
46:Polski
63:marsz
76:ziemii
96:z
3
15:sphere!
37:favourite
70:online

Added by: Michal Malafiejski
Date: 2004-06-01
Time limit [s]: 6
Source limit [B]:50000
Resource: PAL (thanks do Krzysztof Giaro)

SPOJ Problem Set

30. Bytelandian Blingors Network

Problem code: BLINNET

We have discovered the fastest communication medium Bytelandian scientists announced, and they called it *blingors*. The blingors are incomparably better than other media known before. Many companies in Byteland started to build blingors networks, so the information society in the kingdom of Bytes is fact! The priority is to build the core of the blingors network, joining main cities in the country. Assume there is some number of cities that will be connected at the beginning. The cost of building blingors connection between two cities depends on many elements, but it has been successfully estimated. Your task is to design the blingors network connections between some cities in this way that between any pair of cities is a communication route. The cost of this network should be as small as possible.

Remarks

- The name of the city is a string of at most 10 letters from *a,...,z*.
- The cost of the connection between two cities is a positive integer.
- The sum of all connections is not greater than $2^{32}-1$.
- The number of cities is not greater than 10 000.

Input

```
s [number of test cases <= 10]
n [number of cities <= 10 000]
NAME [city name]
p [number of neighbouring cities to the city NAME]
neigh cost
    [neigh - the unique number of city from the main list
     cost - the cost of building the blingors connection from NAME to neigh]

[empty line between test cases]
```

Output

[separate lines] *cost* [the minimum cost of building the blingors network]

Example

Input :

```
2

4
gdansk
2
2 1
3 3
bydgoszcz
3
1 1
3 1
```

```
4 4
torun
3
1 3
2 1
4 1
warszawa
2
2 4
3 1

3
ixowo
2
2 1
3 3
iyekowo
2
1 1
3 7
zetowo
2
1 3
2 7
```

Output:

```
3
4
```

Warning: large Input/Output data, be careful with certain languages

Added by: Lukasz Kuszner
Date: 2004-06-01
Time limit [s]: 8
Source limit [B]: 50000
Resource: PAL

SPOJ Problem Set

31. Fast Multiplication

Problem code: MUL

Multiply the given numbers.

Input

```
n [the number of multiplications <= 1000]
l1 l2 [numbers to multiply (at most 10000 decimal digits each)]
```

Text grouped in [] does not appear in the input file.

Output

The results of multiplications.

Example

```
Input:
5
4 2
123 43
324 342
0 12
9999 12345
```

```
Output:
8
5289
110808
0
123437655
```

Warning: large Input/Output data, be careful with certain languages

Added by: Darek Dereniowski
Date: 2004-06-01
Time limit [s]: 3
Source limit [B]: 50000
Resource: PAL

SPOJ Problem Set

32. A Needle in the Haystack

Problem code: NHAY

Write a program that finds all occurrences of a given pattern in a given input string. This is often referred to as finding a *needle* in a *haystack*.

The program has to detect **all** occurrences of the needle in the haystack. It should take the needle and the haystack as input, and output the positions of each occurrence, as shown below. The suggested implementation is the KMP algorithm, but this is not a requirement. However, a naive approach will probably exceed the time limit, whereas other algorithms are more complicated... The choice is yours.

Input

The input consists of a number of test cases. Each test case is composed of three lines, containing:

- the length of the needle,
- the needle itself,
- the haystack.

The length of the needle is only limited by the memory available to your program, so do not make any assumptions - instead, read the length and allocate memory as needed. The haystack is **not** limited in size, which implies that your program should not read the whole haystack at once. The KMP algorithm is stream-based, i.e. it processes the haystack character by character, so this is not a problem.

The test cases come one after another, each occupying three lines, with no additional space or line breaks in between.

Output

For each test case your program should output all positions of the needle's occurrences within the haystack. If a match is found, the output should contain the position of the first character of the match. Characters in the haystack are numbered starting with zero.

For a given test case, the positions output should be sorted in ascending order, and each of these should be printed in a separate line. For two different test cases, the positions should be separated by an empty line.

Example

```
Sample input:2
na
banananobano
6
foobar
foo
9
foobarfoo
barfoobarfoobarfoobarfoo
```

Sample output:

```
2
4

3
9
15
21
```

Note the double empty line in the output, which means that no match was found for the second test case.

Warning: large Input/Output data, be careful with certain languages

Added by: Michal Malafiejski

Date: 2004-06-03

Time limit [s]: 10

Source limit [B]:50000

Resource: the problem was phrased and test data was supplied by Mr. Maciej 'hawk' Jarzebski

SPOJ Problem Set

33. Trip

Problem code: TRIP

Alice and Bob want to go on holiday. Each of them has drawn up a list of cities to be visited in turn. A list may contain a city more than once. As they want to travel together, they have to agree upon a common route. Noone wants to change the order of the cities on his list or add other cities. Therefore they have no choice but to remove some cities from the list. Of course the common route is to involve as much sight-seeing in cities as possible. There are exactly 26 cities in the region. Therefore they are encoded on the lists as lower case letters from 'a' to 'z'.

Input

The first line of input contains a number $T \leq 10$ that indicates the number of test cases to follow. Each test case consists of two lines; the first line is the list of Alice, the second line is the list of Bob. Each list consists of 1 to 80 lower case letters.

Output

The output for each test case should contain all different trips exactly once that meet the conditions described above. There is at least one such trip, but never more than 1000 different ones. You should order the trips in lexicographic order. Print one blank line between the output of different test cases.

Example

Input

```
1
abcabcaa
acbacba
```

Output

```
ababa
abaca
abcba
acaba
acaca
acbaa
acbca
```

Added by: Adrian Kuegel
Date: 2004-06-05
Time limit [s]: 5
Source limit [B]: 50000
Resource: CEOI 2003

SPOJ Problem Set

34. Run Away

Problem code: RUNAWAY

One of the traps we will encounter in the Pyramid is located in the Large Room. A lot of small holes are drilled into the floor. They look completely harmless at the first sight. But when activated, they start to throw out very hot java, uh ... pardon, lava. Unfortunately, all known paths to the Center Room (where the Sarcophagus is) contain a trigger that activates the trap. The ACM were not able to avoid that. But they have carefully monitored the positions of all the holes. So it is important to find the place in the Large Room that has the maximal distance from all the holes. This place is the safest in the entire room and the archaeologist has to hide there.

Input

The input consists of T test cases. The number of them (T) is given on the first line of the input file. Each test case begins with a line containing three integers X, Y, M separated by space. The numbers satisfy conditions: $1 \leq X, Y \leq 10000, 1 \leq M \leq 1000$. The numbers X and Y indicate the dimensions of the Large Room which has a rectangular shape. The number M stands for the number of holes. Then exactly M lines follow, each containing two integer numbers U_i and V_i ($0 \leq U_i \leq X, 0 \leq V_i \leq Y$) indicating the coordinates of one hole. There may be several holes at the same position.

Output

Print exactly one line for each test case. The line should contain the sentence "The safest point is (P, Q).", where P and Q are the coordinates of the point in the room that has the maximum distance from the nearest hole, rounded to the nearest number with exactly one digit after the decimal point (0.05 rounds up to 0.1).

Example

Sample Input:

```
3
1000 50 1
10 10
100 100 4
10 10
10 90
90 10
90 90
3000 3000 4
1200 85
63 2500
2700 2650
2990 100
```

Sample output:

```
The safest point is (1000.0, 50.0).
The safest point is (50.0, 50.0).
The safest point is (1433.0, 1669.8).
```

Added by: Adrian Kosowski
Date: 2004-06-06
Time limit [s]: 30
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Prague 1999

SPOJ Problem Set

35. Equipment Box

Problem code: EQBOX

There is a large room in the Pyramid called *Room-of-No-Return*. Its floor is covered by rectangular tiles of equal size. The name of the room was chosen because of the very high number of traps and mechanisms in it. The ACM group has spent several years studying the secret plan of this room. It has made a clever plan to avoid all the traps. A specially trained mechanic was sent to deactivate the most feared trap called Shattered Bones. After deactivating the trap the mechanic had to escape from the room. It is very important to step on the center of the tiles only; he must not touch the edges. One wrong step and a large rock falls from the ceiling squashing the mechanic like a pancake. After deactivating the trap, he realized a horrible thing: the ACM plan did not take his equipment box into consideration. The box must be laid onto the ground because the mechanic must have both hands free to prevent contact with other traps. But when the box is laid on the ground, it could touch the line separating the tiles. And this is the main problem you are to solve.

Input

The input consists of T test cases (T is equal to about 10000). The number of them (T) is given on the first line of the input file. Each test case consists of a single line. The line contains exactly four integer numbers separated by spaces: A , B , X and Y . A and B indicate the dimensions of the tiles, X and Y are the dimensions of the equipment box ($1 \leq A, B, X, Y \leq 50000$).

Output

Your task is to determine whether it is possible to put the box on a single tile -- that is, if the whole box fits on a single tile without touching its border. If so, you are to print one line with the sentence "Escape is possible.". Otherwise print the sentence "Box cannot be dropped.".

Example

Sample Input:

```
2
10 10 8 8
8 8 10 10
```

Sample output:

```
Escape is possible.
Box cannot be dropped.
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski
Date: 2004-06-06
Time limit [s]: 10
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Prague 1999

SPOJ Problem Set

36. Secret Code

Problem code: CODE1

The Sarcophagus itself is locked by a secret numerical code. When somebody wants to open it, he must know the code and set it exactly on the top of the Sarcophagus. A very intricate mechanism then opens the cover. If an incorrect code is entered, the tickets inside would catch fire immediately and they would have been lost forever. The code (consisting of up to 100 integers) was hidden in the Alexandrian Library but unfortunately, as you probably know, the library burned down completely.

But an almost unknown archaeologist has obtained a copy of the code something during the 18th century. He was afraid that the code could get to the “wrong people” so he has encoded the numbers in a very special way. He took a random complex number B that was greater (in absolute value) than any of the encoded numbers. Then he counted the numbers as the digits of the system with basis B . That means the sequence of numbers $a_n, a_{n-1}, \dots, a_1, a_0$ was encoded as the number $X = a_0 + a_1B + a_2B^2 + \dots + a_nB^n$.

Your goal is to decrypt the secret code, i.e. to express a given number X in the number system to the base B . In other words, given the numbers X and B you are to determine the “digit” a_0 through a_n .

Input

The input consists of T test cases (equal to about 100000). The number of them (T) is given on the first line of the input file. Each test case consists of one single line containing four integer numbers X_r, X_i, B_r, B_i ($|X_r|, |X_i| \leq 1000000, |B_r|, |B_i| \leq 16$). These numbers indicate the real and complex components of numbers X and B , i.e. $X = X_r + iX_i, B = B_r + iB_i$. B is the basis of the system ($|B| > 1$), X is the number you have to express.

Output

Your program must output a single line for each test case. The line should contain the “digits” $a_n, a_{n-1}, \dots, a_1, a_0$, separated by commas. The following conditions must be satisfied:

- for all i in $\{0, 1, 2, \dots, n\}$: $0 \leq a_i < |B|$
- $X = a_0 + a_1B + a_2B^2 + \dots + a_nB^n$
- if $n > 0$ then $a_n \neq 0$
- $n \leq 100$

If there are no numbers meeting these criteria, output the sentence "The code cannot be decrypted.". If there are more possibilities, print any of them.

Example

Sample Input

```
4
-935 2475 -11 -15
1 0 -3 -2
93 16 3 2
191 -192 11 -12
```

Sample output:

```
8,11,18
1
The code cannot be decrypted.
16,15
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 10

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 1999

SPOJ Problem Set

37. The Proper Key

Problem code: PROPKEY

Many people think that Tetris was invented by two Russian programmers. But that is not the whole truth. The idea of the game is very old -- even the Egyptians had something similar. But they did not use it as a game. Instead, it was used as a very complicated lock. The lock was made of wood and consisted of a large number of square fields, laid out in regular rows and columns. Each field was either completely filled with wood, or empty. The key for this lock was two-dimensional and it was made by joining square parts of the same size as the fields of the lock. So they had a 2D lock and 2D key that could be inserted into the lock from the top. The key was designed so that it was not possible to move it upwards. It could only fall down and it could slide sideways -- exactly like in a Tetris game. The only difference is that the key could not be rotated. Rotation in Tetris is really a Russian invention.

The entry gate into the Pyramid has such a lock. The ACM archaeologists have found several keys and one of them belongs to the lock with a very high probability. Now they need to try them out and find which one to use. Because it is too time-consuming to try all of them, it is better to begin with those keys that may be inserted deeper into the lock. Your program should be able to determine how deep a given key can be inserted into a given lock.

Input

The input consists of T test cases. The number of them (T) is given on the first line of the input file. Each test case begins with a line containing two integers R and C ($1 \leq R, C \leq 100$) indicating the key size. Then exactly R rows follow, each containing C characters. Each character is either a hash mark (#) or a period (.). A hash mark represents one square field made of wood; a period is an empty field. The wooden fields are always connected, i.e. the whole key is made of one piece. Moreover, the key remains connected even if we cut off arbitrary number of rows from its top. There is always at least one non-empty field in the top-most and bottom-most rows and the left-most and right-most columns.

After the key description, there is a line containing two integers D and W ($1 \leq D \leq 10000$, $1 \leq W \leq 1000$). The number W is the lock width, and D is its depth. The next D lines contain W characters each. The character may be either a hash mark (representing the wood) or a period (the free space).

Output

Your program should print one line of output for each test case. The line should contain the statement "The key falls to depth X ". Replace X with the maximum depth to which the key can be inserted by moving it down and sliding it to the left or right only. The depth is measured as the distance between the bottom side of the key and the top side of the lock. If it is possible to move the key through the whole lock and take it away at the bottom side, output the sentence "The key can fall through".

Example

Sample Input:

```
4
2 4
#.##
###.
3 6
#...#
#...#
#..###
2 3
##.
.##
2 7
#.#.#.#
.#.#.#.
1 1
#
1 10
###...###
3 2
##
.#
.#
1 5
#.#.#
```

Sample output:

```
The key falls to depth 2.
The key falls to depth 0.
The key can fall through.
The key falls to depth 2.
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 10

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 1999

SPOJ Problem Set

38. Labyrinth

Problem code: LABYR1

The northern part of the Pyramid contains a very large and complicated labyrinth. The labyrinth is divided into square blocks, each of them either filled by rock, or free. There is also a little hook on the floor in the center of every free block. The ACM have found that two of the hooks must be connected by a rope that runs through the hooks in every block on the path between the connected ones. When the rope is fastened, a secret door opens. The problem is that we do not know which hooks to connect. That means also that the necessary length of the rope is unknown. Your task is to determine the maximum length of the rope we could need for a given labyrinth.

Input

The input consists of T test cases. The number of them (T) is given on the first line of the input file. Each test case begins with a line containing two integers C and R ($3 \leq C, R \leq 1000$) indicating the number of columns and rows. Then exactly R lines follow, each containing C characters. These characters specify the labyrinth. Each of them is either a hash mark (#) or a period (.). Hash marks represent rocks, periods are free blocks. It is possible to walk between neighbouring blocks only, where neighbouring blocks are blocks sharing a common side. We cannot walk diagonally and we cannot step out of the labyrinth.

The labyrinth is designed in such a way that there is exactly one path between any two free blocks. Consequently, if we find the proper hooks to connect, it is easy to find the right path connecting them.

Output

Your program must print exactly one line of output for each test case. The line must contain the sentence "Maximum rope length is X ." where X is the length of the longest path between any two free blocks, measured in blocks.

Example

Sample Input:

```
2
3 3
###
#.#
###
7 6
#####
#.#.###
#.#.###
#.#.#.#
#.....#
#####
```

Sample output:

Maximum rope length is 0.
Maximum rope length is 8.

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 10

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 1999

SPOJ Problem Set

39. Piggy-Bank

Problem code: PIGBANK

Before ACM can do anything, a budget must be prepared and the necessary financial support obtained. The main income for this action comes from Irreversibly Bound Money (IBM). The idea behind is simple. Whenever some ACM member has any small money, he takes all the coins and throws them into a piggy-bank. You know that this process is irreversible, the coins cannot be removed without breaking the pig. After a sufficiently long time, there should be enough cash in the piggy-bank to pay everything that needs to be paid.

But there is a big problem with piggy-banks. It is not possible to determine how much money is inside. So we might break the pig into pieces only to find out that there is not enough money. Clearly, we want to avoid this unpleasant situation. The only possibility is to weigh the piggy-bank and try to guess how many coins are inside. Assume that we are able to determine the weight of the pig exactly and that we know the weights of all coins of a given currency. Then there is some minimum amount of money in the piggy-bank that we can guarantee. Your task is to find out this worst case and determine the minimum amount of cash inside the piggy-bank. We need your help. No more prematurely broken pigs!

Input

The input consists of T test cases. The number of them (T) is given on the first line of the input file. Each test case begins with a line containing two integers E and F . They indicate the weight of an empty pig and of the pig filled with coins. Both weights are given in grams. No pig will weigh more than 10 kg, that means $1 \leq E \leq F \leq 10000$. On the second line of each test case, there is an integer number N ($1 \leq N \leq 500$) that gives the number of various coins used in the given currency. Following this are exactly N lines, each specifying one coin type. These lines contain two integers each, P and W ($1 \leq P \leq 50000$, $1 \leq W \leq 10000$). P is the value of the coin in monetary units, W is its weight in grams.

Output

Print exactly one line of output for each test case. The line must contain the sentence "The minimum amount of money in the piggy-bank is X ." where X is the minimum amount of money that can be achieved using coins with the given total weight. If the weight cannot be reached exactly, print a line "This is impossible."

Example

Sample Input:

```
3
10 110
2
1 1
30 50
10 110
```

```
2
1 1
50 30
1 6
2
10 3
20 4
```

Sample output:

```
The minimum amount of money in the piggy-bank is 60.
The minimum amount of money in the piggy-bank is 100.
This is impossible.
```

Added by: Adrian Kosowski
Date: 2004-06-06
Time limit [s]: 10
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Prague 1999

SPOJ Problem Set

40. Lifting the Stone

Problem code: STONE

There are many secret openings in the floor which are covered by a big heavy stone. When the stone is lifted up, a special mechanism detects this and activates poisoned arrows that are shot near the opening. The only possibility is to lift the stone very slowly and carefully. The ACM team must connect a rope to the stone and then lift it using a pulley. Moreover, the stone must be lifted all at once; no side can rise before another. So it is very important to find the centre of gravity and connect the rope exactly to that point. The stone has a polygonal shape and its height is the same throughout the whole polygonal area. Your task is to find the centre of gravity for the given polygon.

Input

The input consists of T test cases (equal to about 500). The number of them (T) is given on the first line of the input file. Each test case begins with a line containing a single integer N ($3 \leq N \leq 1000000$) indicating the number of points that form the polygon. This is followed by N lines, each containing two integers X_i and Y_i ($|X_i|, |Y_i| \leq 20000$). These numbers are the coordinates of the i -th point. When we connect the points in the given order, we get a polygon. You may assume that the edges never touch each other (except the neighbouring ones) and that they never cross. The area of the polygon is never zero, i.e. it cannot collapse into a single line.

Output

Print exactly one line for each test case. The line should contain exactly two numbers separated by one space. These numbers are the coordinates of the centre of gravity. Round the coordinates to the nearest number with exactly two digits after the decimal point (0.005 rounds up to 0.01). Note that the centre of gravity may be outside the polygon, if its shape is not convex. If there is such a case in the input data, print the centre anyway.

Example

Sample Input:

```
2
4
5 0
0 5
-5 0
0 -5
4
1 1
11 1
11 11
1 11
```

Sample output:

```
0.00 0.00
6.00 6.00
```

Added by: Adrian Kosowski
Date: 2004-06-06
Time limit [s]: 10
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Prague 1999

SPOJ Problem Set

41. Play on Words

Problem code: WORDS1

Some of the secret doors contain a very interesting word puzzle. The team of archaeologists has to solve it to open that doors. Because there is no other way to open the doors, the puzzle is very important for us.

There is a large number of magnetic plates on every door. Every plate has one word written on it. The plates must be arranged into a sequence in such a way that every word begins with the same letter as the previous word ends. For example, the word ‘acm’ can be followed by the word ‘motorola’. Your task is to write a computer program that will read the list of words and determine whether it is possible to arrange all of the plates in a sequence (according to the given rule) and consequently to open the door.

Input

The input consists of T test cases. The number of them (T , equal to about 500) is given on the first line of the input file. Each test case begins with a line containing a single integer number N that indicates the number of plates ($1 \leq N \leq 100000$). Then exactly N lines follow, each containing a single word. Each word contains at least two and at most 1000 lowercase characters, that means only letters ‘a’ through ‘z’ will appear in the word. The same word may appear several times in the list.

Output

Your program has to determine whether it is possible to arrange all the plates in a sequence such that the first letter of each word is equal to the last letter of the previous word. All the plates from the list must be used, each exactly once. The words mentioned several times must be used that number of times.

If there exists such an ordering of plates, your program should print the sentence "Ordering is possible.". Otherwise, output the sentence "The door cannot be opened.".

Example

Sample input:

```
3
2
acm
ibm
3
acm
malform
mouse
2
ok
ok
```

Sample output:

```
The door cannot be opened.  
Ordering is possible.  
The door cannot be opened.
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 15

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 1999

SPOJ Problem Set

42. Adding Reversed Numbers

Problem code: ADDREV

The Antique Comedians of Malidinesia prefer comedies to tragedies. Unfortunately, most of the ancient plays are tragedies. Therefore the dramatic advisor of ACM has decided to transfigure some tragedies into comedies. Obviously, this work is very hard because the basic sense of the play must be kept intact, although all the things change to their opposites. For example the numbers: if any number appears in the tragedy, it must be converted to its reversed form before being accepted into the comedy play.

Reversed number is a number written in arabic numerals but the order of digits is reversed. The first digit becomes last and vice versa. For example, if the main hero had 1245 strawberries in the tragedy, he has 5421 of them now. Note that all the leading zeros are omitted. That means if the number ends with a zero, the zero is lost by reversing (e.g. 1200 gives 21). Also note that the reversed number never has any trailing zeros.

ACM needs to calculate with reversed numbers. Your task is to add two reversed numbers and output their reversed sum. Of course, the result is not unique because any particular number is a reversed form of several numbers (e.g. 21 could be 12, 120 or 1200 before reversing). Thus we must assume that no zeros were lost by reversing (e.g. assume that the original number was 12).

Input

The input consists of N cases (equal to about 10000). The first line of the input contains only positive integer N . Then follow the cases. Each case consists of exactly one line with two positive integers separated by space. These are the reversed numbers you are to add.

Output

For each case, print exactly one line containing only one integer - the reversed sum of two reversed numbers. Omit any leading zeros in the output.

Example

Sample input:

```
3
24 1
4358 754
305 794
```

Sample output:

```
34
1998
1
```

Added by: Adrian Kosowski
Date: 2004-06-06
Time limit [s]: 10
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Prague 1998

SPOJ Problem Set

43. Copying Books

Problem code: BOOKS1

Before the invention of book-printing, it was very hard to make a copy of a book. All the contents had to be re-written by hand by so called *scribers*. The scribe had been given a book and after several months he finished its copy. One of the most famous scribes lived in the 15th century and his name was Xaverius Endricus Remius Ontius Xendrianus (*Xerox*). Anyway, the work was very annoying and boring. And the only way to speed it up was to hire more scribes.

Once upon a time, there was a theater ensemble that wanted to play famous Antique Tragedies. The scripts of these plays were divided into many books and actors needed more copies of them, of course. So they hired many scribes to make copies of these books. Imagine you have m books (numbered $1, 2 \dots m$) that may have different number of pages ($p_1, p_2 \dots p_m$) and you want to make one copy of each of them. Your task is to divide these books among k scribes, $k \leq m$. Each book can be assigned to a single scribe only, and every scribe must get a continuous sequence of books. That means, there exists an increasing succession of numbers $0 = b_0 < b_1 < b_2, \dots < b_{k-1} \leq b_k = m$ such that i -th scribe gets a sequence of books with numbers between $b_{i-1} + 1$ and b_i . The time needed to make a copy of all the books is determined by the scribe who was assigned the most work. Therefore, our goal is to minimize the maximum number of pages assigned to a single scribe. Your task is to find the optimal assignment.

Input

The input consists of N cases (equal to about 200). The first line of the input contains only positive integer N . Then follow the cases. Each case consists of exactly two lines. At the first line, there are two integers m and k , $1 \leq k \leq m \leq 500$. At the second line, there are integers p_1, p_2, \dots, p_m separated by spaces. All these values are positive and less than 10000000.

Output

For each case, print exactly one line. The line must contain the input succession p_1, p_2, \dots, p_m divided into exactly k parts such that the maximum sum of a single part should be as small as possible. Use the slash character ('/') to separate the parts. There must be exactly one space character between any two successive numbers and between the number and the slash.

If there is more than one solution, print the one that minimizes the work assigned to the first scribe, then to the second scribe etc. But each scribe must be assigned at least one book.

Example

Sample input:

```
2
9 3
100 200 300 400 500 600 700 800 900
5 4
```

100 100 100 100 100

Sample output:

100 200 300 400 500 / 600 700 / 800 900
100 / 100 / 100 / 100 100

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 10

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 1998

SPOJ Problem Set

44. Substitution Cipher

Problem code: SCYPHER

Antique Comedians of Malidinesia would like to play a new discovered comedy of Aristofanes. Putting it on a stage should be a big surprise for the audience so all the preparations must be kept absolutely secret. The ACM director suspects one of his competitors of reading his correspondence. To prevent other companies from revealing his secret, he decided to use a substitution cipher in all the letters mentioning the new play.

Substitution cipher is defined by a substitution table assigning each character of the substitution alphabet another character of the same alphabet. The assignment is a bijection (to each character exactly one character is assigned -- not necessary different). The director is afraid of disclosing the substitution table and therefore he changes it frequently. After each change he chooses a few words from a dictionary by random, encrypts them and sends them together with an encrypted message. The plain (i.e. non-encrypted) words are sent by a secure channel, not by mail. The recipient of the message can then compare plain and encrypted words and create a new substitution table.

Unfortunately, one of the ACM cipher specialists have found that this system is sometimes insecure. Some messages can be decrypted by the rival company even without knowing the plain words. The reason is that when the director chooses the words from the dictionary and encrypts them, he never changes their order (the words in the dictionary are lexicographically sorted). String $a_1a_2 \dots a_p$ is lexicographically smaller than $b_1b_2 \dots b_q$ if there exists an integer i , $i \leq p$, $i \leq q$, such that $a_j = b_j$ for each j , $1 \leq j < i$ and $a_i < b_i$.

The director is interested in which of his messages could be read by the rival company. You are to write a program to determine that.

Input

The input consists of N cases (equal to about 1000). The first line of the input contains only positive integer N . Then follow the cases. The first line of each case contains only two positive integers A , $1 \leq A \leq 26$, and K , separated by space. A determines the size of the substitution alphabet (the substitution alphabet consists of the first A lowercase letters of the english alphabet (a-z) and K is the number of encrypted words. The plain words contain only the letters of the substitution alphabet. The plain message can contain any symbol, but only the letters of the substitution alphabet are encrypted. Then follow K lines, each containing exactly one encrypted word. At the next line is encrypted message.

Output

For each case, print exactly one line. If it is possible to decrypt the message uniquely, print the decrypted message. Otherwise, print the sentence 'Message cannot be decrypted.'.

Example

Sample input:

```
2
5 6
cebdbac
cac
ecd
dca
aba
bac
cedab
4 4
cca
cad
aac
bca
bdac
```

Sample output:

```
abcde
Message cannot be decrypted.
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 10

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Prague 1998

SPOJ Problem Set

45. Commedia dell Arte

Problem code: COMMEDIA

So called *commedia dell' arte* is a theater genre first played in Italy at the beginning of the sixteenth century. It was inspired with the Roman Theater. The play had no fixed script and the actors (also called *performers*) had to improvise a lot. There were only a simple directions by the author like "enter the stage and make something funny" or "everyone comes on stage and everything is resolved happily". You can see it might be very interesting to play the *commedia dell' arte*. Therefore the ACM want to put a new play on a stage, which was completely unknown before. The main hero has a puzzle that takes a very important role in the play and gives an opportunity of many improvisations. The puzzle is the worldwide known *Lloyd's Fifteen Puzzle*. ACM wants to make the play more interesting so they want to replace the "standard" puzzle with a three-dimensional one. The puzzle consists of a cube containing M^3 slots. Each slot except one contains a cubic tile (one position is free). The tiles are numbered from 1 to $M^3 - 1$. The goal of the puzzle is to get the original ordering of the tiles after they have been randomly reshuffled. The only allowed moves are sliding a neighbouring tile into the free position along one of the three principal directions. Original configuration is when slot with coordinates (x, y, z) from $\{0, \dots, M-1\}^3$ contains tile number $z \cdot M^2 + y \cdot M + x + 1$ and slot $(M-1, M-1, M-1)$ is free.

You are to write a program to determine whether it is possible to solve the puzzle or not.

Input

The input consists of N cases. The first line of the input contains only positive integer N . Then follow the cases. The first line of each case contains only one integer M , $1 \leq M \leq 100$. It is the size of 3D puzzle cube. Then follow M lines, each contains exactly M^2 numbers on the tiles for one layer. First is the layer on the top of the cube and the last one on the bottom. In each layer numbers are arranged from the left top corner linewise to the right bottom corner of the layer. In other words, slot with coordinates (x, y, z) is described by the $(x + M \cdot y + 1)$ -th number on the $(z + 1)$ -th line. Numbers are separated by space. Number 0 means free position.

Output

For each case, print exactly one line. If the original configuration can be reached by sliding the tiles, print the sentence 'Puzzle can be solved.'. Otherwise, print the sentence 'Puzzle is unsolvable.'.

Example

Sample input:

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

4 6 0 7

Sample output:

Puzzle is unsolvable.
Puzzle can be solved.

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 10

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 1998

SPOJ Problem Set

47. Skyscraper Floors

Problem code: SCRAPER

What a great idea it is to build skyscrapers! Using not too large area of land, which is very expensive in many cities today, the skyscrapers offer an extremely large utility area for flats or offices. The only disadvantage is that it takes too long to get to the upper floors. Of course these skyscrapers have to be equipped not only with a stairway but also with several elevators. But even using ordinary elevators is very slow. Just imagine you want to get from the very top floor to the base floor and many other people on other floors want the same. As a result the elevator stops on almost every floor and since its capacity is limited and the elevator is already full from the upper floors, most stops are useless and just cause a delay. If there are more elevators in the skyscrapers, this problem is a little bit eliminated but still not completely. Most people just press all the buttons of all the elevators and then take the first one so that all elevators will stop on the floor anyway.

However, the solution exists as we shall see. The Antique Comedians of Midilesia headquarters reside in a skyscraper with a very special elevator system. The elevators do not stop on every floor but only on every X -th floor. Moreover each elevator can go just to a certain floor Y (called starting floor) and cannot go any lower. There is one high-capacity elevator which can stop on every elevator's starting floor.

The ACM has a big problem. The headquarters should be moved to another office this week, possibly on a different floor. Unfortunately, the high-capacity elevator is out of order right now so it is not always possible to go to the base floor. One piece of furniture cannot be moved using the stairway because it is too large to pass through the stairway door. You are to write a program that decides whether it is possible to move a piece of furniture from the original office to the other.

Input

The input consists of N cases (equal to about 2000). The first line contains only one positive integer N . Then follow the cases. Each case starts with a line containing four integers F, E, A, B , where $F, 1 \leq F < 50000000$ determines the number of floors in the skyscraper (this means that there are floors 0 to $F-1$), $E, 0 < E < 100$ is the number of elevators and $A, B, 0 \leq A, B < F$ are numbers of the two floors between which the piece of furniture should be moved. Then follow E lines. Each of them contains description of one elevator. There are exactly two integers X and $Y, X > 0, Y \geq 0$ at each line. Y determines, that the elevator starts on the Y -th floor and X determines, that it stops on every X -th floor, eg. for $X = 3, Y = 7$ the elevator stops on floors 7, 10, 13, 16, etc.).

Output

For each case, print exactly one line. If floor B is reachable from floor A not using the stairway, print the sentence 'It is possible to move the furniture.', otherwise print 'The furniture cannot be moved.'.

Example

Sample input:

```
2
22 4 0 6
3 2
4 7
13 6
10 0
1000 2 500 777
2 0
2 1
```

Sample output:

```
It is possible to move the furniture.
The furniture cannot be moved.
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 30

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 1998

SPOJ Problem Set

48. Glass Beads

Problem code: BEADS

Once upon a time there was a famous actress. As you may expect, she played mostly Antique Comedies most of all. All the people loved her. But she was not interested in the crowds. Her big hobby were beads of any kind. Many bead makers were working for her and they manufactured new necklaces and bracelets every day. One day she called her main *Inspector of Bead Makers (IBM)* and told him she wanted a very long and special necklace.

The necklace should be made of glass beads of different sizes connected to each other but without any thread running through the beads, so that means the beads can be disconnected at any point. The actress chose the succession of beads she wants to have and the IBM promised to make the necklace. But then he realized a problem. The joint between two neighbouring beads is not very robust so it is possible that the necklace will get torn by its own weight. The situation becomes even worse when the necklace is disjoined. Moreover, the point of disconnection is very important. If there are small beads at the beginning, the possibility of tearing is much higher than if there were large beads. IBM wants to test the robustness of a necklace so he needs a program that will be able to determine the worst possible point of disjoining the beads.

The description of the necklace is a string $A = a_1 a_2 \dots a_m$ specifying sizes of the particular beads, where the last character a_m is considered to precede character a_1 in circular fashion.

The disjoint point i is said to be worse than the disjoint point j if and only if the string $a_i a_{i+1} \dots a_n a_1 \dots a_{i-1}$ is lexicographically smaller than the string $a_j a_{j+1} \dots a_n a_1 \dots a_{j-1}$. String $a_1 a_2 \dots a_n$ is lexicographically smaller than the string $b_1 b_2 \dots b_n$ if and only if there exists an integer i , $i \leq n$, so that $a_j = b_j$, for each j , $1 \leq j < i$ and $a_i < b_i$.

Input

The input consists of N cases. The first line of the input contains only positive integer N . Then follow the cases. Each case consists of exactly one line containing necklace description. Maximal length of each description is 10000 characters. Each bead is represented by a lower-case character of the english alphabet (a-z), where $a < b \dots z$.

Output

For each case, print exactly one line containing only one integer -- number of the bead which is the first at the worst possible disjoining, i.e. such i , that the string $A[i]$ is lexicographically smallest among all the n possible disjoinings of a necklace. If there are more than one solution, print the one with the lowest i .

Example

Sample input:

```
4
helloworld
amandamanda
dontcallmebfu
aaabaaa
```

Sample output:

```
10
11
6
5
```

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 10

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Prague 1998

SPOJ Problem Set

49. Hares and Foxes

Problem code: HAREFOX

The Antique Comedians of Malidinesia play an interesting comedy where many animals occur. Because they want their plays to be as true as possible, a specialist studies the behaviour of various animals. Recently, he is interested in a binary dynamic ecological system hares-foxes (SHF). As a part of this project, you are asked to design and implement intelligent automatic target evaluation simulator (IATES) for this system. The behaviour of the SHF follows so called *standard model*, described by the following set of difference equations.

$$\begin{aligned}h_{y+1} &= a.h_y - b.f_y \\ f_{y+1} &= c.f_y + d.h_y\end{aligned}$$

where h_y resp. f_y represent the difference of the number of hares resp. foxes in year y and the reference count determined at the beginning of the experiment. The units of h_y and f_y are unknown. Therefore, h_y and f_y are to be treated as real numbers. Your task is to write a program to determine the long term evolution of SHF.

Input

The input consists of N cases (equal to about 5000). The first line of the input contains only positive integer N . Then follow the cases. Each case consists of six real numbers a, b, c, d, h_{1998} and f_{1998} , written in this order on three lines, two numbers per line, separated by one or more spaces. The numbers are given in the classical format, i.e. optional sign, sequence of digits, optional dot and optional sequence of digits. The text form of a number does not exceed 10 characters. Each case is followed by one empty line.

Output

For each case, print one of the following sentences:

- 'Ecological balance will develop.' - if after sufficiently long time the population of both hares and foxes approaches the reference count with an arbitrary a priori given precision, i.e. $\lim h_y = 0$ and $\lim f_y = 0$.
- 'Hares will die out while foxes will overgrow.' - if after sufficiently long time the population of hares resp. foxes falls under resp. exceeds any a priori given threshold, i.e. $\lim h_y = -infinity$ and $\lim f_y = +infinity$.
- 'Hares will overgrow while foxes will die out.' - if after sufficiently long time the population of foxes resp. hares falls under resp. exceeds any a priori given threshold, i.e. $\lim h_y = +infinity$ and $\lim f_y = -infinity$.
- 'Both hares and foxes will die out.' - if after sufficiently long time the population of both hares and foxes falls under any a priori given threshold, i.e. $\lim h_y = -infinity$ and $\lim f_y = -infinity$.
- 'Both hares and foxes will overgrow.' - if after sufficiently long time the

population of both hares and foxes exceeds any a priori given threshold, i.e. $\lim h_y = +infinity$ and $\lim f_y = +infinity$.

- 'Chaos will develop.' - if none of the above mentioned description fits.

Example

Sample input:

```
2
2 0.5
0.5 0.6
2 3
```

```
0.1 1
2 0.1
1 1
```

Sample output:

```
Both hares and foxes will overgrow.
Hares will die out while foxes will overgrow.
```

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 10

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Prague 1998

SPOJ Problem Set

50. Invitation Cards

Problem code: INCARDS

In the age of television, not many people attend theater performances. Antique Comedians of Malidinesia are aware of this fact. They want to propagate theater and, most of all, Antique Comedies. They have printed invitation cards with all the necessary information and with the programme. A lot of students were hired to distribute these invitations among the people. Each student volunteer has assigned exactly one bus stop and he or she stays there the whole day and gives invitation to people travelling by bus. A special course was taken where students learned how to influence people and what is the difference between influencing and robbery.

The transport system is very special: all lines are unidirectional and connect exactly two stops. Buses leave the originating stop with passengers each half an hour. After reaching the destination stop they return empty to the originating stop, where they wait until the next full half an hour, e.g. X:00 or X:30, where 'X' denotes the hour. The fee for transport between two stops is given by special tables and is payable on the spot. The lines are planned in such a way, that each round trip (i.e. a journey starting and finishing at the same stop) passes through a *Central Checkpoint Stop* (CCS) where each passenger has to pass a thorough check including body scan.

All the ACM student members leave the CCS each morning. Each volunteer is to move to one predetermined stop to invite passengers. There are as many volunteers as stops. At the end of the day, all students travel back to CCS. You are to write a computer program that helps ACM to minimize the amount of money to pay every day for the transport of their employees.

Input

The input consists of N cases. The first line of the input contains only positive integer N . Then follow the cases. Each case begins with a line containing exactly two integers P and Q , $1 \leq P, Q \leq 1000000$. P is the number of stops including CCS and Q the number of bus lines. Then there are Q lines, each describing one bus line. Each of the lines contains exactly three numbers - the originating stop, the destination stop and the price. The CCS is designated by number 1 . Prices are positive integers the sum of which is smaller than 1000000000 . You can also assume it is always possible to get from any stop to any other stop.

Output

For each case, print one line containing the minimum amount of money to be paid each day by ACM for the travel costs of its volunteers.

Example

Sample input:

```
2
2 2
1 2 13
2 1 33
```

```
4 6
1 2 10
2 1 60
1 3 20
3 4 10
2 4 5
4 1 50
```

Sample output:

```
46
210
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 10

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Prague 1998

SPOJ Problem Set

51. Fake tournament

Problem code: TOUR

We consider only special type of tournaments. Each tournament consists of a series of matches. We have n competitors at the beginning of a competition and after each match the loser is moved out of the competition and the winner stays in (there are no draws). The tournament ends when there is only one participant left - the winner. It is a task of National Sports Federation to schedule the matches. Members of this comitee can pick the contestants for the first match. Then, after they know the result, they say which of the remaining contestants meet in the second match, and so on until there is only one participant left.

It is easy to see that not only skill and training decides about the win, but also "luck" - i.e. the schedule. The members of NSF know it as well.

The comitee used the training time to look carefully on the performance of each probable contestant. It is clear now, at the start of the season, that some of the results between the competitors are 100% predictable. Having this information NFS considers if it is possible to schedule the matches in such a way that the given contestant x wins. That is to plan the matches for x only with those who will lose with him (then he wins the whole tournament of course). If it is possible then we say that **the tournament can be set for x**

Task

Your task is to write a program which determines the number of contestants of a given tournament for which it is possible to set it.

Input

t [number of tests to solve].

In the first line of each test: n ($1 \leq n \leq 1000$) - the number of participants of the tournament. We number the participants with numbers $1, 2, \dots, n$. The following line contains a list of participants who will inevitably win with participant 1. This list begins with a number m (the number of contestants "better" than 1) and numbers n_1, n_2, \dots, n_m delimited by single spaces.

Next $n-1$ lines contain analogous lists for participants $2, 3, \dots, n$.

Remark 1. The fact that participant **a** would lose with **b** and **b** would lose with **c** doesn't necessarily mean that **a** would lose with **c** in a direct match.

Remark 2. It is not possible that **a** is on the list of contestants better than **b** and **b** is on the list of **a** at the same time.

Output

For each test your program should output a single integer - the number of participants, for which it is possible to set the tournament.

Example

Input :

1

3

2 3 2

1 3

0

Output :

1

Added by: Adam Dzedzej

Date: 2004-06-08

Time limit [s]: 2

Source limit [B]:50000

Resource: Internet Contest **Pogromcy Algorytmów** (Algorithm Tamers)
Round IV, 2001

SPOJ Problem Set

53. Kamil

Problem code: KAMIL

Some kids cannot pronounce all letters, some of them they sometimes pronounce correctly and sometimes incorrectly. Kamil sometimes says T instead of K, but he never says K instead of T. Similarly he sometimes says D instead of G. Instead of R he sometimes says L and sometimes F. Of course it happens that he pronounces the letter correctly. Kamil's father always thinks how many words can mean the word spoken by his son (it doesn't matter if they are real English words).

Task

Write a program which

- reads from standard input the words spoken by Kamil
- counts how many different words can that mean
- writes the outcome on standard output

Input

Ten test cases (given one under another, you have to process all!). Every test case is a single line - a word spoken by Kamil. Only 26 capital letters are used. The length of the word is at most 20.

Output

For every testcase write an integer in a single line with a single integer, denoting the number of words which Kamil's word can mean.

Score

The score awarded to your program is the number of bytes the source code you submit. The fewer points you score, the better. Submissions are not allowed to exceed 256 bytes.

Remark. It may turn out impossible to solve this problem in some languages.

Example

```
Input:
FILIPEK
[and 9 test cases more]
Output:
4
[and 9 test cases more]
```

Added by: Adam Dzedzej

Date: 2004-06-08

Time limit [s]: 3

Source limit [B]: 256

Resource: Internet Contest Pogromcy Algorytmow (Algorithm Tamers) Round I, 2003

SPOJ Problem Set

54. Julka

Problem code: JULKA

Julka surprised her teacher at preschool by solving the following riddle:

Klaudia and Natalia have 10 apples together, but Klaudia has two apples more than Natalia. How many apples does each of the girls have?

Julka said without thinking: Klaudia has 6 apples and Natalia 4 apples. The teacher tried to check if Julka's answer wasn't accidental and repeated the riddle every time increasing the numbers. Every time Julka answered correctly. The surprised teacher wanted to continue questioning Julka, but with big numbers she couldn't solve the riddle fast enough herself. Help the teacher and write a program which will give her the right answers.

Task

Write a program which

- reads from standard input the number of apples the girls have together and how many more apples Klaudia has,
- counts the number of apples belonging to Klaudia and the number of apples belonging to Natalia,
- writes the outcome to standard output

Input

Ten test cases (given one under another, you have to process all!). Every test case consists of two lines. The first line says how many apples both girls have together. The second line says how many more apples Klaudia has. Both numbers are positive integers. It is known that both girls have no more than 10^{100} (1 and 100 zeros) apples together. As you can see apples can be very small.

Output

For every test case your program should output two lines. The first line should contain the number of apples belonging to Klaudia. The second line should contain the number of apples belonging to Natalia.

Example

Input:

```
10
2
[and 9 test cases more]
```

Output:

```
6
4
[and 9 test cases more]
```

Added by: Adam Dzedzej
Date: 2004-06-08
Time limit [s]: 3
Source limit [B]:50000
Resource: Internet Contest **Pogromcy Algorytmow** (Algorithm Tamers)
Round II, 2003

SPOJ Problem Set

55. Jasiiek

Problem code: JASIEK

Jasiek is only 6 years old, but he already has many skills. He likes drawing and asking riddles very much. This morning he got a sheet of grid paper and a pencil from his mother and he started drawing. All his drawings have some common properties:

- Jasiek colors full grid squares;
- if some coloured grid squares touch each other, it means they have a common edge or a corner;
- all grid squares are connected, which means between every two coloured grid squares there is a sequence of coloured grid squares, which have a common edge;
- there are no white holes, that is from every white grid box it is possible to draw a line to the boundary of the sheet which never touches any coloured grid square.

At noon mom phoned and asked what Jasiek's today's picture was. The boy didn't answer directly, but described the picture by a sequence of moves needed to walk around the centres of the coloured squares on its boundary, ie. those squares which have at least one common corner with a white square. Jasiek set the starting square and then gave the sequence of moves necessary to walk along the boundary squares anti-clockwise. Mom was very surprised by the complexity of the picture and especially by the number of coloured squares. Given Jasiek's description, can you quickly count how many coloured squares there are in the picture?

Task

Write a program which

- reads (from standard input) Jasiek's description of the picture,
- counts the number of coloured squares,
- writes out the outcome (to standard output).

Input

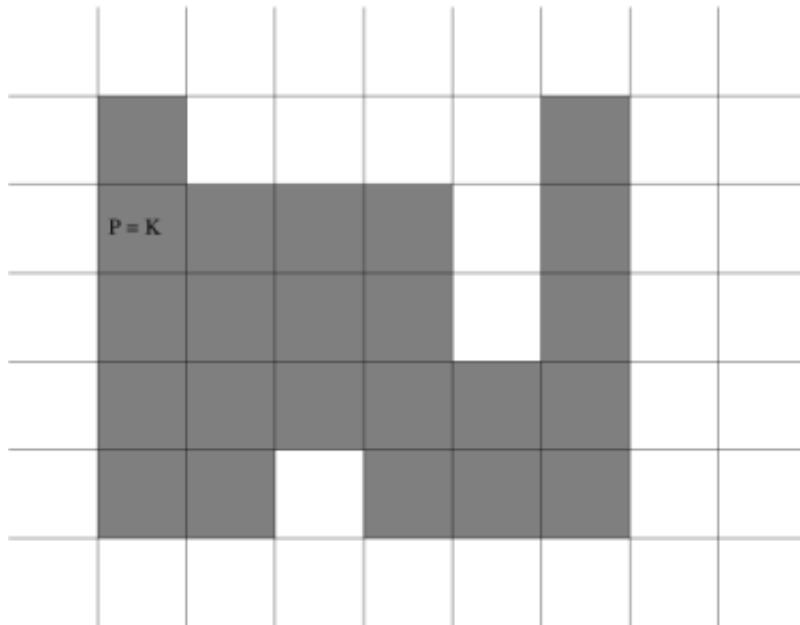
Ten test cases (given one under another, you have to process all!). Each of the test cases is a series of lines. Each line consists of only one character. The letter *P* means the beginning of the description. The letter *K* means the end of the description (and the test case). All other lines (if any) contain one of the letters N, W, S or E (N meaning North, W - West, S - South and E - East). Every line of the description corresponds to the relative position of the centre of some square on the boundary of the picture. The first and the last line correspond to the same square. A letter in a line other than the first or the last tells you which way you have to go in order to get to the next boundary square when going around the picture anti-clockwise. Jasiek's description finishes after going around the picture once. The length of the description doesn't exceed 20000 letters.

Output

For every testcase your program should write (to the standard output) only one line with one integer, equal to the number of coloured squares in Jasiek's picture.

Example

Only one test case.



Input :

P
S
S
S
E
N
E
E
S
E
E
N
N
N
N
S
S
S
W
W
N
N
W
W
W
N
S

K

Output:

23

Added by: Adam Dzedzej

Date: 2004-06-09

Time limit [s]: 5

Source limit [B]: 50000

Resource: Internet Contest **Pogromcy Algorytmow** (Algorithm Tamers)
Round III, 2003

SPOJ Problem Set

56. Dyzio

Problem code: DYZIO

Dyzio is Jasiek's friend and he also likes riddles. Here is a riddle he came up with:

Jasiek, here is a piece of string, which has to be cut into smaller pieces. I will not tell you directly how to do it, but look at this sequence of zeros (0) and ones (1). A one at the beginning means that the string has to be cut in half. If the first digit was zero, it would be the only digit in the sequence and mean you don't have to cut anything - I want the whole string. If you have to cut the string anyway then after the first 1 I wrote what to do with the left piece (according to the same rules as with the whole string) and then I wrote what to do with the right piece of string (all the time with the same rules of notation). Every time you have to cut the left piece first, only then can you cut the right one. Now start cutting and tell me, how many cuts you have to do until you have cut off the shortest piece.

Unfortunately mom hid the scissors from Jasiek, but luckily a computer was at hand and Jasiek quickly wrote a program simulating the string cutting. Can you write such a program?

Task

Write a program which

- reads (from standard input) description of the way the string is cut,
- counts how many cuts have to be made in order to get the first shortest piece.
- writes out the outcome (to standard output)

Input

Ten test cases (given one under another, you have to process all!). Each test case consists of two lines. In the first line there is a number n ($1 \leq n \leq 20000$). In the second line one zero-one word (a sequence of zeros and ones without spaces between them) of length n - the description of the cutting procedure given by Dyzio.

Output

For every testcase your program should write (to the standard output) only one line with one integer equal to the number of cuts which have to be made in order to get the shortest piece.

Example

Only one test case.

Input :
9
110011000
Output :
4

Added by: Adam Dzedzej
Date: 2004-06-10
Time limit [s]: 5
Source limit [B]:50000
Resource: Internet Contest **Pogromcy Algorytmow** (Algorithm Tamers)
Round III, 2003

SPOJ Problem Set

57. Supernumbers in a permutation

Problem code: SUPPER

An n -element permutation is an n -element sequence of distinct numbers from the set $\{1, 2, \dots, n\}$. For example the sequence 2,1,4,5,3 is a 5-element permutation. We are interested in the longest increasing subsequences in a permutation. In this exemplary permutation they are of length 3 and there are exactly 2 such subsequences: 2,4,5 and 1,4,5. We will call a number belonging to any of the longest increasing subsequences a *supernumber*. In the permutation 2,1,4,5,3 the supernumbers are 1,2,4,5 and 3 is not a supernumber. Your task is to find all supernumbers for a given permutation.

Task

Write a program which

- reads a permutation from standard input,
- finds all its supernumbers,
- writes all found numbers to standard output.

Input

Ten test cases (given one under another, you have to process all!). Each test case consists of two lines. In the first line there is a number n ($1 \leq n \leq 100000$). In the second line: an n -element permutation - n numbers separated by single spaces.

Output

For every test case your program should write two lines. In the first line - the number of supernumbers in the input permutation. In the second line the supernumbers separated by single spaces in increasing order.

Example

Only one test case.

Input :

5

2 1 4 5 3

Output :

4

1 2 4 5

Warning: large Input/Output data, be careful with certain languages

Added by: Adam Dzedzej
Date: 2004-06-10
Time limit [s]: 20
Source limit [B]: 50000
Resource: Internet Contest **Pogromcy Algorytmow** (Algorithm Tamers)
Round IV, 2003

SPOJ Problem Set

58. Crime at Piccadily Circus

Problem code: PICAD

Sherlock Holmes is carrying out an investigation into the crime at Piccadily Circus. Holmes is trying to determine the maximal and minimal number of people staying simultaneously at the crime scene at a moment when the crime could have been committed. Scotland Yard has already carried out a thorough investigation already, interrogated everyone seen at the crime scene and determined what time they appeared at the crime scene and what time they left. Doctor Watson offered his help to process the data gathered by Scotland Yard and find the numbers interesting Sherlock Holmes, but he has some difficulties. Help him!

Task

Write a program which

- reads from standard input the time interval during which the crime was committed and the data gathered by Scotland Yard,
- finds the minimal and the maximal number of people present simultaneously in the time interval when the crime could have been committed, (these numbers can be zero, though it would seem strange that noone was present at the crime scene when the crime was committed, but that's the type of crime Holmes and Watson have to deal with)
- writes the outcome to standard output.

Input

Ten test cases (given one under another, you have to process all!). The first line of each test case consists of two integer numbers p and k , $0 \leq p \leq k \leq 100000000$. These denote the first and the last moment when the crime could have been committed. The second line of each test case contains one integer n , $3 \leq n \leq 5000$. This is the number of people interrogated by Scotland Yard. The next n lines consist of two integers - line $i+2$ contains numbers a_i and b_i separated by a single space, $0 \leq a_i \leq b_i \leq 100000000$. These are the moments at which the i -th person appeared at and left the crime scene respectively. It means that the i -th person was at the crime scene for the whole time from moment a_i until moment b_i (inclusive).

Output

For every test case your program should write to the standard output only one line with two integers separated by a single space: the minimal and maximal number of people staying simultaneously at the crime scene, in the interval between moment p and k , (inclusive).

Example

Only one test case.

Input :

```
5 10
4
1 8
5 8
7 10
8 9
```

Output :

```
1 4
```

Added by: Adam Dzedzej

Date: 2004-06-10

Time limit [s]: 30

Source limit [B]:50000

Resource: Internet Contest **Pogromcy Algorytmow** (Algorithm Tamers)
Round IV, 2003

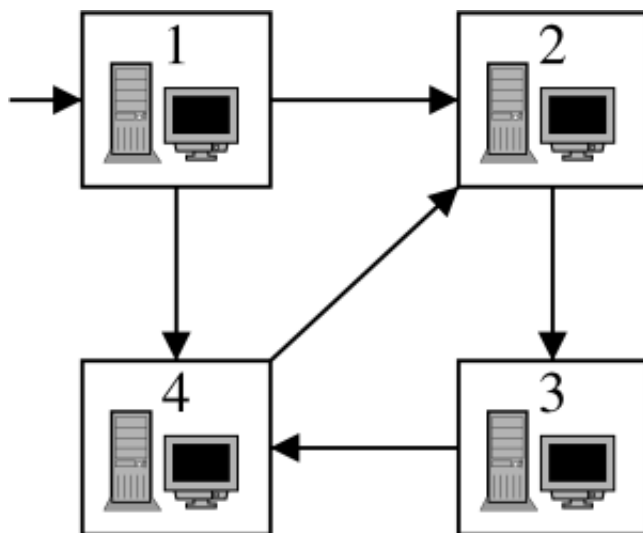
SPOJ Problem Set

59. Bytelandian Information Agency

Problem code: BIA

Bytelandian Information Agency (BIA) uses a net of n computers. The computers are numbered from 1 to n , and the computer number 1 is a server. The computers are connected by one-way information channels. Every channel connects a pair of computers. The whole network is organised in such a way that one can send information from the server to any other computer either directly or indirectly.

When BIA acquires new information, the information is put on the server and propagated in the net. The chief of BIA considers what would happen if one computer stopped working (was blown away by terrorists for example). It could happen that some other computers would stop receiving information from the server, because the broken computer was a necessary transmitter. We will call such computers *critical*. For example in the situation in the picture below the critical computers are 1 and 2. 1 is the server and all information sent from the server to 3 has to go through 2.



Task

Write a program which

- reads a description of the net from standard input,
- finds all critical computers.
- writes the numbers of critical computers to standard output.

Input

Ten test cases (given one under another, you have to process all!). Each test case consists of several lines. In the first line there are numbers n and m . n denotes the number of computers in the net, ($2 \leq n \leq 5000$). m denotes the number of information channels, $n-1 \leq m \leq 200000$. The following m lines describes a single information channel and consist of two integer numbers a and b separated by a space. It means the computer a sends information to computer b by that channel. You

may assume there are no two channels which start and end at the same points a, b .

Output

For every testcase your program should write two lines. In the first line k - the number of critical computers in the net. In the second line k numbers separated by single spaces - the numbers of critical computers in increasing order.

Example

Input:

```
4 5
1 2
1 4
2 3
3 4
4 2
[and 9 test cases more]
```

Output:

```
2
1 2
[and 9 test cases more]
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adam Dzedzej

Date: 2004-06-14

Time limit [s]: 15

Source limit [B]:50000

Resource: Internet Contest **Pogromcy Algorytmow** (Algorithm Tamers)
Round IV, 2003

SPOJ Problem Set

60. The Gordian Dance

Problem code: DANCE

The Gordian Dance is a traditional Bytelandian dance performed by two pairs of dancers. At the beginning the dancers are standing in the corners of the square $ABCD$, forming two pairs: $A-B$ and $C-D$. Every pair is holding an outstretched string. So in the starting position both strings are stretched horizontally and parallel.

$A \text{ ————— } B$

$D \text{ ————— } C$

The starting position of dancers.

The dance consists of a series of moves. There are two kinds of moves:

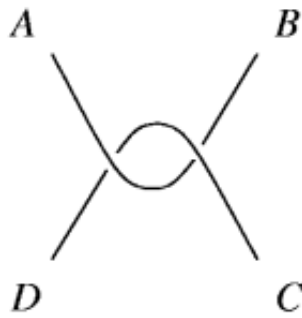
- (S) The dancers standing at points B and C swap positions (without releasing their strings) in such a way that the dancer standing at B raises the hand in which he is holding the string and, when going to point C , lets the dancer going from C to B pass in front of him, under his arm.
- (R) All dancers make a turn by 90 degrees clockwise without releasing their strings. This means that the dancer from A goes to B , the dancer from B goes to C , the dancer from C goes to D , and the dancer from D goes to A .

During the dance the strings tangle with each other, but in the end they should be untangled and stretched horizontally and parallel. The dancers do not have to occupy the same spots as in the beginning. The dance requires a lot of experience, because the strings can be extremely tangled during the dance. The sequence of moves after which they are no longer tangled and are stretched horizontally and parallel can be difficult to guess.

Your program should help beginner dancers end a dance. You are to determine the minimal number of moves required to end the dance given a sequence of moves already performed.

Illustration

For example after the sequence SS we get the following configuration.



The shortest sequence of moves required to end the dance is of length 5: *RSRSS*.

Task

Write a program which

- reads from standard input the moves made in a dance,
- finds the minimal number of moves required to untangle the strings and stretch them horizontally and parallel (the dancers don't have to be in their starting spots).
- writes the outcome to standard output.

Input

Ten test cases (given one under another, you have to process all!). The first line of each test case consists of one integer n equal to the number of moves already made, $0 \leq n \leq 1000000$. The second line of each test case consists of one word of length n , made up of letters *S* and/or *R*.

Output

For every testcase your program should write to standard output only one line with one integer number: the minimal number of moves required to untangle the strings and stretch them horizontally and parallel.

Example

Only one test case.

Input :

2

SS

Output :

5

Warning: large Input/Output data, be careful with certain languages

Added by: Adam Dzedzej

Date: 2004-06-15

Time limit [s]: 5

Source limit [B]: 50000

Resource: Internet Contest **Pogromcy Algorytmow**(Algorithm Tamers) 2003 Round V

SPOJ Problem Set

61. Brackets

Problem code: BRCKTS

We will call a **bracket word** any word constructed out of two sorts of characters: the opening bracket "(" and the closing bracket ")". Among these words we will distinguish **correct bracket expressions**. These are such bracket words in which the brackets can be matched into pairs such that

- every pair consists of an opening bracket and a closing bracket appearing further in the bracket word
- for every pair the part of the word between the brackets of this pair has equal number of opening and closing brackets

On a bracket word one can do the following operations:

- **replacement** -- changes the i -th bracket into the opposite one
- **check** -- if the word is a correct bracket expression

Task

Write a program which

- reads (from standard input) the bracket word and the sequence of operations performed,
- for every check operation determines if the current bracket word is a correct bracket expression,
- writes out the outcome (to standard output).

Input

Ten test cases (given one under another, you have to process all!). Each of the test cases is a series of lines. The first line of a test consists of a single number n ($1 \leq n \leq 30000$) denoting the length of the bracket word. The second line consists of n brackets, not separated by any spaces. The third line consists of a single number m -- the number of operations. Each of the following m lines carries a number k denoting the operation performed. $k=0$ denotes the check operation, $k>0$ denotes replacement of k -th bracket by the opposite.

Output

For every test case your program should print a line:

Test i :

where i is replaced by the number of the test and in the following lines, for every check operation in the i -th test your program should print a line with the word YES, if the current bracket word is a correct bracket expression, and a line with a word NO otherwise. (There should be as many lines as check operations in the test.)

Example

Input:

```
4
()((
4
4
0
2
0
[and 9 test cases more]
```

Output:

```
Test 1:
YES
NO
[and 9 test cases more]
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adam Dzedzej

Date: 2004-06-15

Time limit [s]: 25

Source limit [B]: 50000

Resource: Internet Contest **Pogromcy Algorytmow**(Algorithm Tamers) 2003 Round IV

SPOJ Problem Set

62. The Imp

Problem code: IMP

An Imp jumps on an infinite chessboard. Moves possible for the Imp are described by two pairs of integers: (a,b) and (c,d) - from square (x,y) the Imp can move to one of the squares: (x+a,y+b), (x-a,y-b), (x+c,y+d), (x-c,y-d). We want to know for which square different from (0,0) to which the Imp can jump from (0,0) (possibly in many moves) the value $|x|+|y|$ is the lowest.

Task

Write a program which

- reads from standard input two pairs (a,b) and (c,d) of integers, different from (0,0), describing moves of the Imp,
- determines a pair of integers (x,y) different from (0,0), for which the Imp can jump (possibly in many moves) from square (0,0) to square (x,y) and for which the value $|x|+|y|$ is the lowest.
- writes out to standard output the value $|x|+|y|$.

Input

Ten test cases. Each test consists of four numbers a,b,c,d in one line, separated by spaces.
-100000 \leq a, b, c, d \leq 100000

Output

For every test case your program should write a single line with a number equal the lowest possible value $|x|+|y|$.

Example

```
Input:
13 4 17 5
[and 9 test cases more]
Output:
2
[and 9 answers more]
```

Added by: Adam Dzedzej

Date: 2004-06-15

Time limit [s]: 5

Source limit [B]: 50000

Resource: Internet Contest **Pogromcy Algorytmow**(Algorithm Tamers) 2003 Round V

SPOJ Problem Set

63. Square Brackets

Problem code: SQRBR

You are given:

- a positive integer n ,
- an integer k , $1 \leq k \leq n$,
- an increasing sequence of k integers $0 < s_1 < s_2 < \dots < s_k \leq 2n$.

What is the number of proper bracket expressions of length $2n$ with opening brackets appearing in positions s_1, s_2, \dots, s_k ?

Illustration

Several proper bracket expressions:

```
[[]][[]][[]]  
[[[]]][[]][[]]
```

An improper bracket expression:

```
[[[[]]])][[]][[]]
```

There is exactly one proper expression of length 8 with opening brackets in positions 2, 5 and 7.

Task

Write a program which for each data set from a sequence of several data sets:

- reads integers n , k and an increasing sequence of k integers from input,
- computes the number of proper bracket expressions of length $2n$ with opening brackets appearing at positions s_1, s_2, \dots, s_k ,
- writes the result to output.

Input

The first line of the input file contains one integer d , $1 \leq d \leq 10$, which is the number of data sets. The data sets follow. Each data set occupies two lines of the input file. The first line contains two integers n and k separated by single space, $1 \leq n \leq 19$, $1 \leq k \leq n$. The second line contains an increasing sequence of k integers from the interval $[1; 2n]$ separated by single spaces.

Output

The i -th line of output should contain one integer - the number of proper bracket expressions of length $2n$ with opening brackets appearing at positions s_1, s_2, \dots, s_k .

Example

Sample input:

```
5
1 1
1
1 1
2
2 1
1
3 1
2
4 2
5 7
```

Sample output:

```
1
0
2
3
2
```

Added by: Adrian Kosowski

Date: 2004-06-22

Time limit [s]: 5

Source limit [B]:50000

Resource: III Polish Collegiate Team Programming Contest (AMPPZ), 1998

SPOJ Problem Set

64. Permutations

Problem code: PERMUT1

Let $A = [a_1, a_2, \dots, a_n]$ be a permutation of integers $1, 2, \dots, n$. A pair of indices (i, j) , $1 \leq i < j \leq n$, is an *inversion* of the permutation A if $a_i > a_j$. We are given integers $n > 0$ and $k \geq 0$. What is the number of n -element permutations containing exactly k inversions?

For instance, the number of 4-element permutations with exactly 1 inversion equals 3.

Task

Write a program which for each data set from a sequence of several data sets:

- reads integers n and k from input,
- computes the number of n -element permutations with exactly k inversions,
- writes the result to output.

Input

The first line of the input file contains one integer d , $1 \leq d \leq 10$, which is the number of data sets. The data sets follow. Each data set occupies one line of the input file and contains two integers n ($1 \leq n \leq 12$) and k ($0 \leq k \leq 98$) separated by a single space.

Output

The i -th line of the output file should contain one integer - the number of n -element permutations with exactly k inversions.

Example

Sample input:

```
1
4 1
```

Sample output:

```
3
```

Added by: Adrian Kosowski

Date: 2004-06-22

Time limit [s]: 5

Source limit [B]: 50000

Resource: III Polish Collegiate Team Programming Contest (AMPPZ), 1998

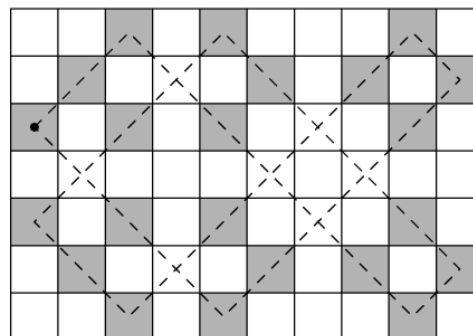
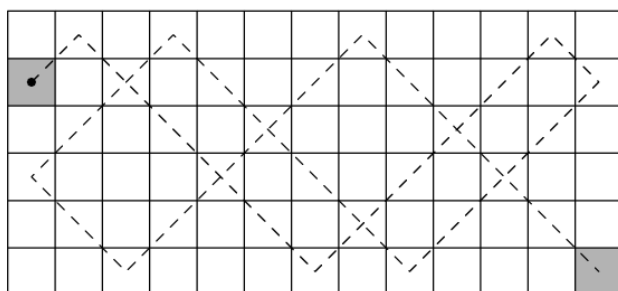
SPOJ Problem Set

65. Ball

Problem code: BALL1

On the rectangular chessboard of $n \times m$ square fields we choose one field adjacent to the edge of the chessboard, called the starting field. Then we put a ball in the center of this field and push it to roll through the chessboard. The diameter of the ball equals the width (and height) of chessboard field. The angle between the direction of ball movement and the edge of the chessboard equals 45 degrees. The ball bounces off the edges of the chessboard: if the ball touches the edge of the chessboard then each component of its velocity perpendicular to the edge touched is reversed. At the start the ball is pushed toward increasing coordinates (when the starting field is a field of the highest coordinate, the ball bounces momentarily).

We assign a point to a field of the chessboard each time the point of adjacency between the ball and the chessboard enters the interior of the field. The game is over when a point is assigned to the starting field. What is the number of fields to which an odd number of points is assigned? The following figures illustrate the problem. The route of the ball is marked with a dashed line. Fields with the odd number of points are shadowed.



Task

Write a program which for each data set from a sequence of several data sets:

- reads the dimensions of the chessboard and the coordinates of starting field from input,
- computes the number of fields with the odd number of points,
- writes the result to output.

Input

The first line of the input file contains one integer d , $1 \leq d \leq 10$, which is the number of data sets. The data sets follow. Each data set occupies one line of the input file. Such a line consists of four integers x , y , a , b separated with single spaces. These integers are the x - and y -dimensions of the chessboard and x - and y -coordinates of the starting field, respectively. Integers x and y are greater than two, the number of fields of the chessboard does not exceed 10^9 , the starting field is adjacent to the

edge of the chessboard.

Output

The i -th line of output should contain one integer which is equal to the number of fields of the chessboard with the odd number of points.

Example

Sample input:

```
2
13 6 1 5
10 7 1 5
```

Sample output:

```
2
22
```

Added by: Adrian Kosowski

Date: 2004-06-06

Time limit [s]: 10

Source limit [B]: 50000

Resource: III Polish Collegiate Team Programming Contest (AMPPZ), 1998

SPOJ Problem Set

66. Cross-country

Problem code: CRSCNTRY

Agness, a student of computer science, is very keen on crosscountry running, and she participates in races organised every Saturday in a big park. Each of the participants obtains a route card, which specifies a sequence of checkpoints, which they need to visit in the given order. Agness is a very attractive girl, and a number of male runners have asked her for a date. She would like to choose one of them during the race. Thus she invited all her admirers to the park on Saturday and let the race decide. The winner would be the one, who scores the maximum number of points. Agnes came up with the following rules:

- a runner scores one point if he meets Agnes at the checkpoint,
- if a runner scored a point at the checkpoint, then he cannot get another point unless he and Agnes move to the next checkpoints specified in their cards.
- route specified by the card may cross the same checkpoint more than once,
- each competitor must strictly follow race instructions written on his card.

Between two consecutive meetings, the girl and the competitors may visit any number of checkpoints. The boys will be really doing their best, so you may assume, that each of them will be able to visit any number of checkpoints whilst Agnes runs between two consecutive ones on her route.

Task

Write a program which for each data set from a sequence of several data sets:

- reads in the contents of Agnes' race card and contents of race cards presented to Tom,
- computes the greatest number of times Tom is able to meet Agnes during the race,
- writes it to output.

Input

There is one integer d in the first line of the input file, $1 \leq d \leq 10$. This is the number of data sets. The data sets follow. Each data set consists of a number of lines, with the first one specifying the route in Agnes' race card. Consecutive lines contain routes on cards presented to Tom. At least one route is presented to Tom. The route is given as a sequence of integers from interval $[1, 1000]$ separated by single spaces. Number 0 stands for the end of the route, though when it is placed at the beginning of the line it means the end of data set. There are at least two and at most 2000 checkpoints in a race card.

Output

The i -th line of the output file should contain one integer. That integer should equal the greatest number of times Tom is able to meet with Agnes for race cards given in the i -th data set.

Example

Sample input:

```
3
1 2 3 4 5 6 7 8 9 0
1 3 8 2 0
2 5 7 8 9 0
1 1 1 1 1 1 2 3 0
1 3 1 3 5 7 8 9 3 4 0
1 2 35 0
0
1 3 5 7 0
3 7 5 1 0
0
1 2 1 1 0
1 1 1 0
0
```

Sample output:

```
6
2
3
```

Added by: Adrian Kosowski

Date: 2004-06-08

Time limit [s]: 10

Source limit [B]:50000

Resource: III Polish Collegiate Team Programming Contest (AMPPZ), 1998

SPOJ Problem Set

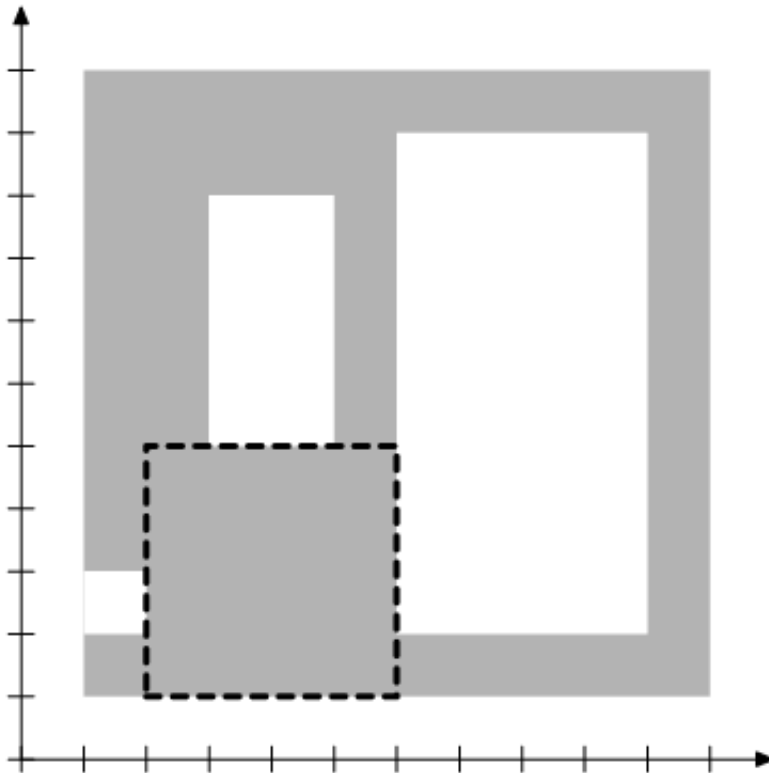
67. Cutting out

Problem code: CUTOOT

One has to cut out a number of rectangles from a paper square. The sides of each rectangle are to be parallel to the sides of the square. Some rectangles can be already cut out. What is the largest area of a rectangle which can be cut out from the remaining paper?

Illustration

Three rectangles have been cut out from the square 10x10 in the figure shown below. The area of the largest rectangle that can be cut out from the remaining paper is 16. One of such rectangles is shown with a dashed line.



Task

Write a program that for each data set from a sequence of several data sets:

- reads descriptions of a square and rectangles from the input,
- computes the area of the largest rectangle which can be cut out from the remaining paper,
- writes the result to output.

Input

The first line of the input file contains one positive integer d not larger than 10. This is the number of data sets. The data sets follow. Each set of data occupies two consecutive lines of the input file. The first line of each data set contains two integers n and r , $1 \leq n \leq 40000$, $0 \leq r \leq 100$. The integer n is the length of the sides of an input square. The integer r is the number of rectangles which have been cut out from the square. The second line of the data set contains a sequence of $4r$ integers x_1, x_2, \dots, x_{4r} from the interval $[0, n]$ separated by single spaces. For each $i = 1, \dots, r$, integers $x_{4i-3}, x_{4i-2}, x_{4i-1}, x_{4i}$ describe the i -th rectangle: x_{4i-3} is the distance of its left side from the left side of the square, x_{4i-2} is the distance of its right side from the left side of the square, x_{4i-1} is the distance of the bottom side of the rectangle from the bottom side of the square and x_{4i} is the distance of its top side from the bottom side of the square.

Output

For each $i = 1, \dots, d$, your program should write only one integer to the i -th line of the output file -- the largest area of a rectangle which can be cut out from the rest of the i -th square.

Example

Sample input:

```
2
6 2
0 3 0 3 3 6 3 6
10 3
0 5 0 5 0 10 5 10 9 10 0 5
```

Sample output:

```
9
20
```

Added by: Adrian Kosowski

Date: 2004-06-08

Time limit [s]: 10

Source limit [B]: 50000

Resource: III Polish Collegiate Team Programming Contest (AMPPZ), 1998

SPOJ Problem Set

68. Expression

Problem code: EXPR1

We are given an integer k and an arithmetic expression E with the operations '+', '-', and arguments from the set $\{0,1,\dots,9\}$. Is it possible to put some parentheses in E to get a new expression E' whose value equals k ? If the answer is positive what is the minimum number of pairs of parentheses '(', ')' that are necessary?

Illustration

It is sufficient to put one pair of parentheses in the expression $5 - 4 + 5$ to get an expression with value -4, namely $5 - (4 + 5) = -4$.

Task

Write a program that for each data set from a sequence of several data sets:

- reads an expression E and an integer k from input,
- verifies whether it is possible to put some parentheses in E to get a new expression E' whose value equals k and computes the minimal number of pairs of parentheses '(', ')' necessary, if the answer is positive,
- writes the result to output.

Input

The first line of the input file contains one positive integer d not larger than 10. This is the number of data sets. The data sets follow. Each set of data occupies two consecutive lines of the input file. The first line contains two integers n and k , $2 \leq n \leq 40$, $-180 \leq k \leq 180$. The even integer n is the length of E . The second line contains the expression itself written as a string of length n . The string contains operators '+' or '-' in odd positions and numbers from the set $\{0,1,\dots,9\}$ in even positions.

Output

For each $i = 1, \dots, d$, your program should write to the i -th line of the output file one word 'NO' if the i -th input expression cannot be transformed into any expression of value k , and the smallest number of pairs of parentheses necessary otherwise.

Example

Sample input:

```
5
6 -4
+5-4+5
2 1
+1
4 1
```

```
-1+1
4 0
-1+1
4 -2
-1+1
```

Sample output:

```
1
0
NO
0
1
```

Added by: Adrian Kosowski
Date: 2004-06-08
Time limit [s]: 10
Source limit [B]: 50000
Resource: III Polish Collegiate Team Programming Contest (AMPPZ), 1998

SPOJ Problem Set

69. Moulds

Problem code: MOULDS

In a factory, moulds for casting metal objects are produced by a special cutting device. The device is equipped with cuboid-shaped blade of size 1 mm x 1 mm x 30 mm (its height) which operates with each of its sides thus producing the mould from cuboid of size 250 mm x 250 mm x 30 mm (its height). The end of the blade never lowers below the bottom surface of the cuboid. In any moment the distance between initial and current position doesn't exceed 1000.

The machine understands special command language which has the following grammar:

```
<command block> ::= [ <command> ; {<command> ; } ]
<command>       ::= <lift> | <shift> | <command block>
<lift>          ::= ^ <distance>
<shift>         ::= @ <direction> <distance>
<direction>     ::= N | S | W | E
<distance>      ::= <sign> <number> | <number>
<number>        ::= <digit> {<digit>}
<sign>          ::= - | +
<digit>         ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

where {exp} means zero or more exps.

The command <lift> causes moving the blade downwards when the distance is a positive number and upwards otherwise. The command <shift> moves the blade in the appropriate direction (N--north, S--south, W--west, E--east).

Task

Write a program which for each data set from a sequence of several data sets:

- reads a command block from input,
- computes the volume of hollows made by the machine commanded by a given command block (assuming that before the execution the blade is located 1 mm above the north-west corner of the virgin cuboid),
- writes the result to output.

Input

The first line of the input file contains one integer d , $1 \leq d \leq 10$, which is the number of data sets. The data sets follow. Each data set occupies one line of the input file and is a word derived from <command block> of the above grammar of length not exceeding 10000 characters.

Output

The i -th line of the output file should contain one integer -- the volume (in cubic mm) of the hollows made by the machine controlled by the command block given in the i -th data set.

Example

Sample input:

```
1
[^2;@S2;]
```

Sample output:

```
3
```

Added by: Adrian Kosowski

Date: 2004-06-08

Time limit [s]: 10

Source limit [B]: 50000

Resource: III Polish Collegiate Team Programming Contest (AMPPZ), 1998

SPOJ Problem Set

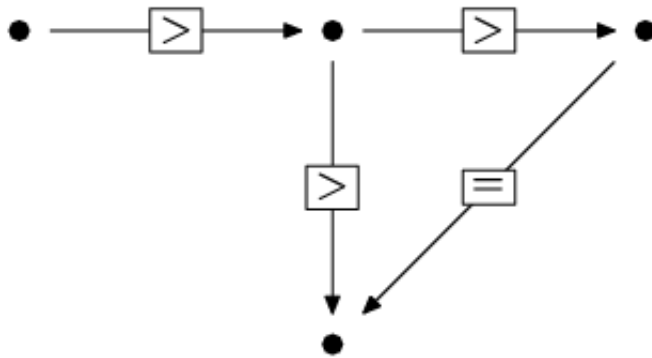
70. Relations

Problem code: RELATS1

You are given a directed graph, whose edges are labeled with relational symbols '<', '>' and '='. For a nonnegative integer k , a k -correct G-labeling is a mapping from vertices of G into integers from interval $[0, k]$ such that numbers at the ends of each edge satisfy the relation described by the label of the edge. We assume that an element on the left side of the relational symbol is a number assigned to the initial vertex. Compute the smallest k for which k -correct G-labeling exists or verify that such labeling doesn't exist for any k .

Illustration

For the graph in the figure the smallest $k = 2$.



Task

Write a program that for each data set from a sequence of several data sets:

- reads a description of a graph G from the input file,
- verifies whether there exist an integer k for which it is possible to label G k -correctly and, if the answer is positive, computes the smallest such k ,
- writes the result to the output file.

Input

The first line of the input file contains one positive integer d not larger than 10. This is the number of data sets. The data sets follow. Each data set is described in two consecutive lines of the input file. In the first line there are two integers n and m separated by a single space. The number n is the number of vertices of G and m is the number of edges of G . Numbers n and m satisfy the inequalities: $1 \leq n \leq 1000$, $0 \leq m \leq 10000$. The vertices are numbered with integers from 1 to n and are identified by these numbers. There are no parallel edges and self-loops in the graph. (Two different edges $u_1 \rightarrow v_1$ and $u_2 \rightarrow v_2$ are parallel iff $u_1 = u_2$ and $v_1 = v_2$.) There are $3m$ integers separated by single spaces in the second line. The numbers at positions $3i-2$ and $3i-1$, $1 \leq i \leq m$, are the ends of the i -th edge,

the beginning and the end, respectively, whereas the number at position $3i$ is a number from the set $\{-1,0,1\}$ and it is the label of the i -th edge: -1 represents ' $<$ ', 0 represents ' $=$ ' and 1 represents ' $>$ '.

Output

For the i -th data set, $1 \leq i \leq d$, your program should write one word NO in the i -th line of the output file if a k -correct labeling doesn't exist for any k , or the smallest integer k for which such a labeling exists.

Example

Sample input:

```
4
4 4
1 2 -1 2 3 0 2 4 -1 3 4 -1
2 2
1 2 -1 2 1 -1
2 2
1 2 -1 2 1 1
3 3
1 2 0 3 2 0 3 1 0
```

Sample output:

```
2
NO
1
0
```

Added by: Adrian Kosowski

Date: 2004-06-08

Time limit [s]: 10

Source limit [B]: 50000

Resource: III Polish Collegiate Team Programming Contest (AMPPZ), 1998

SPOJ Problem Set

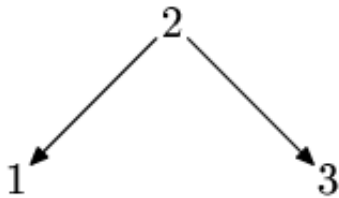
71. Tree

Problem code: TREE1

Consider an n -vertex binary search tree T containing n keys $1, 2, \dots, n$. A permutation $p = [p_1, \dots, p_n]$ of the integers $1, 2, \dots, n$ is said to be *consistent with the tree T* if the tree can be built from the empty one as the result of inserting integers p_1, p_2, \dots, p_n . Find how many permutations are consistent with the tree T .

Illustration

Exactly 2 permutations are consistent with the tree in the figure below.



Task

Write a program that for each data set from a sequence of several data sets:

- reads from the input file a description of an input tree T ,
- computes the number of permutations consistent with T ,
- writes the result to output.

Input

The first line of the input file contains one positive integer d not larger than 10. This is the number of data sets. The data sets follow. Each set of data occupies two consecutive lines of the input file. The first line contains only one integer n , $1 \leq n \leq 30$. This is the number of vertices of the tree. The second line contains a sequence of n integers separated by single spaces. The integers are keys in the input tree given in the prefix order. The first integer in the sequence is the key from the root of the tree. It is followed by the keys from the left subtree written in the prefix order. The sequence ends with the keys from the right subtree, also given in the prefix order.

Output

For each $i = 1, \dots, d$, your program should write to the i -th line of output the number of permutations consistent with the tree described in the i -th data set.

Example

Sample input:

```
5
3
2 1 3
3
1 2 3
1
1
4
2 1 3 4
4
1 4 2 3
```

Sample output:

```
2
1
1
3
1
```

Added by: Adrian Kosowski

Date: 2004-06-08

Time limit [s]: 10

Source limit [B]: 50000

Resource: III Polish Collegiate Team Programming Contest (AMPPZ), 1998

SPOJ Problem Set

72. Food Shortage in Byteland

Problem code: BYTEFOOD

Fanatics from the BBFO blew up all the food factories in the Bytelandian capital! Hurry up! There is still some food left in shops. Some shops are located in the centre, others in the suburbs, so Johnny has to decide which of them are worth visiting. Some shops can be very big and have plenty of food in them, others may be so small that food disappears from them at an alarming rate... So? Help Johnny buy as much food as possible.

There are n open shops, each of them located at position (x_i, y_i) , for $i=1, \dots, n$, where $0 \leq x_i, y_i \leq 250$. The distances between shops are measured using the Manhattan metric (i.e. as sums of absolute values of differences of x and y coordinates). Besides, every shop is characterized by a linear time function describing how much food is left in the shop at the moment:

$$f_i = \max\{0, a_i - b_i * \text{time}\}$$

where $0 \leq a_i \leq 1000000$, $0 \leq b_i \leq 1000$, while time is the time (in minutes) that has elapsed from the moment Johnny left the house (assume that Johnny does not live in the same place as any shop). If Johnny decides to stay in a shop, he can buy at most b_i units of food per minute. Otherwise, he can move along the orthogonal system of streets of the city at a constant speed of unit distance per minute. Johnny only ever changes the action he is performing at the full minute. Because his family is slowly beginning to starve, he should be back at home not later than m minutes after he left. Since there are thousands of starving families in the capital, Johnny can't spend more than $1 \leq c_i \leq 10$ minutes in a shop. Moreover, he will never go into the same shop twice for fear of being lynched...

Input

The first line of input contains a single positive integer $t \leq 1000$, the number of test cases. Each test case begins with the number of shops in the city $1 \leq n \leq 1000$ and the deadline $1 \leq m \leq 5000$. Then the following n lines consist of four integers $x_i y_i a_i b_i c_i$ each, describing the position and the parameters of the function for food availability of the i -th shop. At the end of every test case comes a line with two integers $p q$ (between 0 and 250), corresponding to the x and y coordinates of the position of Johnny's house.

All the input data are integers.

Output

Process all test cases. The correct output for the i -th test case takes the following form:

i [the number of the test case, in the input order]

$s m$ [s is the number of the target shop and $m > 0$ is the number of minutes spent in it].

At the end of the series of moves you should always write a line consisting of two zeros ('0 0').

All the output data should be integers.

Scoring

The score of your program is the total amount of food that Johnny bought (summed over all the testcases in which he managed to come back home before the deadline).

Example

Input

```
4
2 20
0 0 100 5 5
10 0 200 10 10
5 0
2 20
0 0 180 15 10
10 0 200 20 10
5 0
4 101
0 0 1000 20 5
20 0 200 1 5
0 20 5000 200 5
20 20 300 5 10
10 10
1 15
1 0 10 1 5
5 0
```

Output

```
1
2 10
0 0
2
1 10
0 0
3
3 5
4 10
2 1
0 0
4
1 5
0 0
```

Score

Score = 1261

Added by: Michal Malafiejski

Date: 2004-06-09

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004 (problemset 2)

SPOJ Problem Set

73. Bacterial

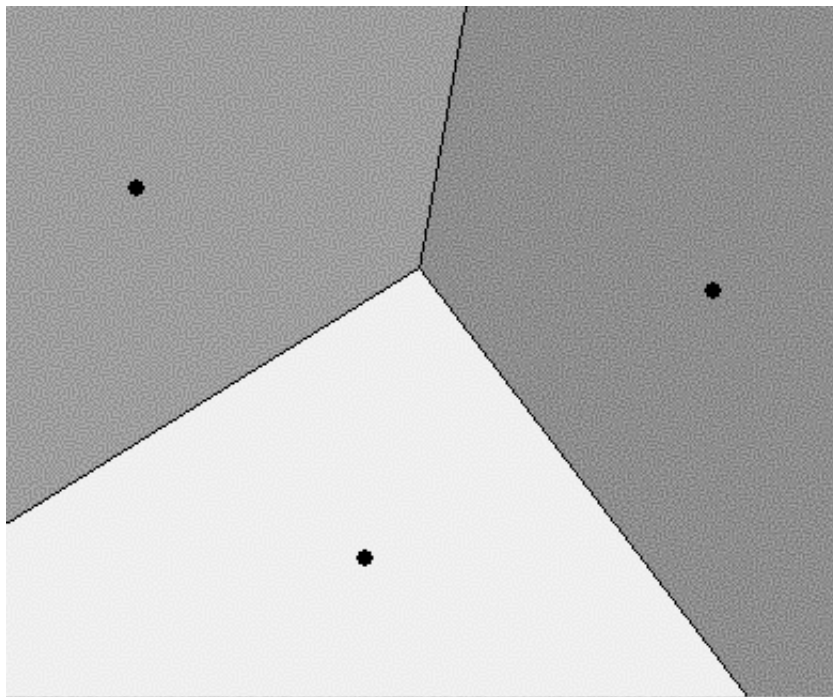
Problem code: BAC

In the biology laboratory we are observing several bacterial samples, and under the microscope we have them shaded with different colors to see them expanding their territory on the plate.

It is interesting to know that the bacterial are quite 'friendly' that once they meet each other, they do not expand into each other's occupation any more. The bacterial samples are expanding at similar speeds and we take them as the same speed.

Since the experiment is tedious and lengthy (Oh My God! there are several thousand samples at our pick), we are going to run a simulation based on this reality, taking the variable that these samples may be planted in different starting spots.

We are using rectangular plates and bacterial racing is bounded within the plate.



Input format

There are multiple test cases (about 20000 of them) each taking the following format:

- one line with two integers between 1 and 1000 inclusive indicating width and height of the plate
- one line with one integer between 1 and 100 inclusive indicating the number of bacterial samples
- for each bacterial sample there is one line with two integers indicating the sample's position: x y, where x, y specify a position within or on the bound of the plate.

The plate lies in such a coordinating system that the lower-left corner of it is (0,0) and the upper-right corner is (width,height).

A test with zero plate area marks the end of the tests and this one shall not be processed.

Between each input block there is a blank line.

Output format

Generate a report having the samples sorted on their domination, with each line taking the following format:

<sample id> <area occupation>

where: 'sample id' takes 3 columns right justified, with '0' padded to the left as necessary, and 'area occupation' takes 14 columns with 2 digit precision, right justified.

The sample occupying more area shall be reported prior to those occupying less. The input data will ensure enough difference in areas to avoid ambiguity.

Between each output block there shall be a blank line.

Example

Sample input:

```
10 10
2
5 5
0 0

0 0
```

Sample output:

```
01234567890123456789 (Do not output this ruler line)
001          87.50
002          12.50
```

Warning: large Input/Output data, be careful with certain languages

Added by: Neal Zane
Date: 2004-06-08
Time limit [s]: 20
Source limit [B]:50000
Resource: Neal Zane

SPOJ Problem Set

74. Divisor Summation

Problem code: DIVSUM

Given a natural number n ($1 \leq n \leq 500000$), please output the summation of all its proper divisors.

Definition: A proper divisor of a natural number is the divisor that is strictly less than the number.

e.g. number 20 has 5 proper divisors: 1, 2, 4, 5, 10, and the divisor summation is: $1 + 2 + 4 + 5 + 10 = 22$.

Input

An integer stating the number of test cases (equal to about 200000), and that many lines follow, each containing one integer between 1 and 500000 inclusive.

Output

One integer each line: the divisor summation of the integer given respectively.

Example

Sample Input:

```
3
2
10
20
```

Sample Output:

```
1
8
22
```

Warning: large Input/Output data, be careful with certain languages

Added by: Neal Zane
Date: 2004-06-10
Time limit [s]: 5
Source limit [B]: 5000
Resource: Neal Zane

SPOJ Problem Set

75. Editor

Problem code: EDIT1

Have you ever programmed in Brainf**k? If yes, then you know how annoying it is to press the same key several times in a row. So what we all need, is a good editor. Here are the functions that the editor should have:

- '\n': begin a new line. If the last line was empty, stop processing and print out all lines.
- 'd': copy all characters from the current line, and append them after the last character in this line. For example, if current line contains ab, and d is pressed two times, the result will be abababab
- any other character: append it to the current line.

Please note, that the solution may only be submitted in Brainfk or Intercal.**

Input

There is exactly one test case. You can assume, that there is no key press of 'd' when the line is still empty.

Output

Print the output that the editor described above would produce on the given input. You can assume, that no line is created with more than 150 characters.

Example

Input:

sample-test-dd-d-ddend signalled by two newlines

Output:

sample-test-----enen signalleenen signalle by two newlines

Added by: Adrian Kuegel

Date: 2004-06-12

Time limit [s]: 5

Source limit [B]:50000

SPOJ Problem Set

76. Editor Inverse

Problem code: EDIT2

You are given a text. Calculate the minimum number of keystrokes needed to produce this text, if the editor described below is used.

If you haven't read the problem "Editor" before, here is a description of the functionality of the editor:

- '\n': begin a new line. If the last line was empty, stop processing and print out all lines.
- 'd': copy all characters from the current line, and append them after the last character in this line.
For example, if current line contains ab, and d is pressed two times, the result will be abababab
- any other character: append it to the current line.

Input

The input consists of **exactly ten** test cases. Each test case consists of a line with at most 600 characters. The character 'd' is not used in any of the lines, but all other printable ascii characters may occur.

Output

For each test case, first print a line containing the minimum number of key strokes to produce the given line of text. In the next lines, write the keys that are pressed to produce the text. If there are several possibilities with minimum number of keystrokes, you should also minimise the number of lines, if there is still more than one possibility, minimise number of keystrokes before the first '\n', then second '\n', ...

Since 'd' is a costly operation in the editor, for each output line you should minimise the number of 'd' characters as the 2nd criterion after minimising number of keystrokes in this line.

The original input line should be the same as the output of the editor (processing the output you produce), if '\n' characters are ignored.

Notice that you have to terminate the input for the editor with two '\n'.

Example

Here only two test cases.

Input :

00001123444456789000011234444446789

Output :

1800d11234444567891800d1123444d6789

Added by: Adrian Kuegel
Date: 2004-06-12
Time limit [s]: 5
Source limit [B]:50000

SPOJ Problem Set

77. New bricks disorder

Problem code: BRICKS

You have n bricks arranged in a line on the table. There is exactly one letter on each of them. Your task is to rearrange those bricks so that letters on them create some specified inscription. While rearranging you can only swap adjacent bricks with specified letters (you are given m pairs $(a_1, b_1), \dots, (a_m, b_m)$ and you are only allowed to swap bricks with a_i on one of them and b_i on the second, for some $i=1, \dots, m$). You should check if it is possible to accomplish this - and if it is - calculate minimal needed number of swaps.

Input

There is a single integer c on the first line of input. Then c test cases follow: each of them consists of two lines of small letters (a..z) with lengths not exceeding 100000 (descriptions of starting and ending configurations), one integer m in the next line and then m lines with two letters a_i, b_i in each of them.

Output

For each test case you should print -1 if it is not possible to rearrange bricks or the minimal number of swaps if it is possible (if so, output this value modulo 2^{32}).

Example

Input :

```
4
ab
ba
0
abc
cba
3
ab
cb
ca
cabbbc
cbabbc
1
ab
abba
baab
1
ab
```

Output :

```
-1
3
1
2
2
```

Warning: large Input/Output data, be careful with certain languages

Added by: Pawel Gawrychowski
Date: 2004-06-17
Time limit [s]: 20
Source limit [B]:10000

SPOJ Problem Set

78. Marbles

Problem code: MARBLES

Hänschen dreams he is in a shop with an infinite amount of marbles. He is allowed to select n marbles. There are marbles of k different colors. From each color there are also infinitely many marbles. Hänschen wants to have at least one marble of each color, but still there are a lot of possibilities for his selection. In his effort to make a decision he wakes up. Now he asks you how many possibilities for his selection he would have had. Assume that marbles of equal color can't be distinguished, and the order of the marbles is irrelevant.

Input

The first line of input contains a number $T \leq 100$ that indicates the number of test cases to follow. Each test case consists of one line containing n and k , where n is the number of marbles Hänschen selects and k is the number of different colors of the marbles. You can assume that $1 \leq k \leq n \leq 1000000$.

Output

For each test case print the number of possibilities that Hänschen would have had. You can assume that this number fits into a signed 64 bit integer.

Example

Input:

```
2
10 10
30 7
```

Output:

```
1
475020
```

Added by: Adrian Kuegel
Date: 2004-06-19
Time limit [s]: 1
Source limit [B]: 10000

SPOJ Problem Set

82. Easy Problem

Problem code: EASYPIE

Last year there were a lot of complaints concerning the set of problems. Most contestants considered our problems to be too hard to solve. One reason for this is that the team members responsible for the problems are not able to evaluate properly whether a particular problem is easy or hard to solve. (We have created until now so many problems, that all seems quite easy.) Because we want our future contests to be better we would like to be able to evaluate the hardness of our problems after the contest using a history of submissions.

There are a few statistics that we can use for evaluating the hardness of a particular problem: the number of accepted solutions of the problem, the average number of submissions of the problem and the average time consumed to solve it (as "General rules" of the contest state "the time consumed for a solved problem is the time elapsed from the beginning of the contest to the submittal of the accepted run"). For the latter two statistics we consider only the teams which solved this particular problem. Needless to say we ask you to write a program that computes aforementioned statistics for all problems.

Task

Write a program that:

- reads a history of submissions during an ACM contest,
- computes for each problem the number of accepted solutions of the problem, the average number of submissions and the average time consumed to solve it,
- writes the result.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case, the first line of the input contains one integer n ($1 \leq n \leq 2000$) being the number of submissions during the contest. Each of the next n lines describes one submission and contains a submission time (measured in seconds from the beginning of the contest), a team identifier, a problem identifier and a result of evaluating the submission separated by single spaces. The submission time is a positive integer not greater than 18000. The team identifier is a non-empty string consisting of at most five small letters or digits. The problem identifier is a capital letter A, B, ..., or I. The result is a capital letter A (the submission is accepted) or R (the submission is rejected).

Submissions are given in nondecreasing order according to submission times and there are 62 teams competing.

Please note that if a problem is accepted all further submission of this problem by the same team are possible but they should not be taken to the statistics.

Output

For each test case the output consists of nine lines. The first line corresponds to problem A, the second line to problem B, and so on. Each line should contain the problem identifier, the number of accepted solutions of the problem, the average number of submissions done by teams that solved that problem and the average time consumed to solve it separated by single spaces. The latter two statistics should be printed only if there was at least one accepted solution of the given problem and should be rounded to two fractional digits (in particular 1.235 should be rounded to 1.23).

Example

Sample input:

```
1
12
10 wawu1 B R
100 chau1 A A
2000 uwr2 B A
2010 wawu1 A R
2020 wawu1 A A
2020 wawu1 B A
4000 wawu2 C R
6000 chau1 A R
7000 chau1 A A
8000 ppl A A
8000 zil2 B R
9000 zil2 B A
```

Sample output:

```
A 3 1.33 3373.33
B 3 1.67 4340.00
C 0
D 0
E 0
F 0
G 0
H 0
I 0
```

Added by: Adrian Kosowski

Date: 2004-06-26

Time limit [s]: 5

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Warsaw 2003

SPOJ Problem Set

83. Bundling

Problem code: BUNDLE

Outel, a famous semiconductor company, recently released a new model of microprocessor called Platinum. Like many modern processors, Platinum can execute many instructions in one clock step providing that there are no dependencies between them (instruction I_2 is dependent on instruction I_1 if for example I_2 reads a register that I_1 writes to). Some processors are so clever that they calculate on the fly which instructions can be safely executed in parallel. Platinum however expects this information to be explicitly specified. A special marker, called simply a stop, inserted between two instructions indicates that some instructions after the stop are possibly dependent on some instructions before the stop. In other words instructions between two successive stops can be executed in parallel and there should not be dependencies between them.

Another interesting feature of Platinum is that an instruction sequence must be split into groups of one, two or three successive instructions. Each group has to be packed into a container called a bundle. Each bundle has 3 slots and a single instruction can be put into each slot, however some slots may stay empty. Each instruction is categorized into one of 10 instruction types denoted by consecutive capital letters from A to J (instructions of the same type have similar functionality, for example type A groups integer arithmetic instructions and type F groups instructions). Only instructions of certain types are allowed to be packed into one bundle. A template specifies one permissible combination of instruction types within a bundle. A template can also specify a position of a stop in the middle of a bundle (there is at most one such stop allowed). In addition, stops are allowed between any two adjoining bundles. A set of templates is called a bundling profile. When packing instructions into bundles, one has to use templates from bundling profile only.

Although Platinum is equipped with an instruction cache it was found that for maximal performance it is most crucial to pack instructions as densely as possible. Second important thing is to use a small number of stops.

Your task is to write a program for bundling Platinum instructions. For the sake of simplicity we assume that the instructions cannot be reordered.

Task

Write a program that:

- reads a bundling profile and a sequence of instructions,
- computes the minimal number of bundles into which the sequence can be packed without breaking the dependencies and the minimal number of all stops that are required for the minimal number of bundles,
- writes the result.

Input

The input begins with the integer z , the number of test cases. Then z test cases follow.

The first line of each test case description contains two integers t and n separated by a single space. Integer t ($1 \leq t \leq 1500$) is the number of templates in the bundling profile. Integer n ($1 \leq n \leq 100000$) is the number of instructions to be bundled.

Each of the next t lines specifies one template and contains 3 capital letters t_1, t_2, t_3 with no spaces in between followed by a space and an integer p . Letter t_i ($A \leq t_i \leq J$) is an instruction type allowed in the i -th slot. Integer p ($0 \leq p \leq 2$) is the index of the slot after which the stop is positioned (0 means no stop within the bundle).

Each of the next n lines specifies one instruction. The i -th line of these n lines contains one capital letter c_i and an integer d_i , separated by a single space. Letter c_i ($A \leq c_i \leq J$) is the type of the i -th instruction. Integer d_i ($0 \leq d_i < i$) is the index of the last instruction (among the previous ones) that the i -th instruction is dependent on (0 means that the instruction is not dependent on any former instruction).

You can assume that for each instruction type c describing an instruction in the instruction sequence there is at least one template containing c .

Output

For each test case, the first and only line of the output contains two integers b and s . Integer b is the minimal number of bundles in a valid packing. Integer s is the minimal number of all stops that are required for the minimal number of bundles.

Example

Sample input:

```
1
4 9
ABB 0
BAD 1
AAB 0
ABB 2
B 0
B 1
A 1
A 1
B 4
D 0
A 0
B 3
B 0
```

Sample output:

```
4 3
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski
Date: 2004-06-26
Time limit [s]: 15
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Warsaw 2003

For each test case, the first line of the input contains one integer n ($3 \leq n \leq 250\,000$) being the number of sections of the route. The second line of the input contains a sequence of n characters N, E, S or W with no spaces in between. Each character is a description of one section of the route. Character N, E, S or W means that Mirek walks 10 meters north, east, south or west respectively. You may assume that at least one shortcut exists for the given route.

Output

The first and only line of the output contains integers l , b , e and character d separated by single spaces. Integer l is the length of the shortest shortcut (measured in 10 m segments). Integers b and e are the numbers of break points where the shortcut begins and ends respectively (we number break points with consecutive integers from 0 for Mirek's home to n for the university). Character d is the direction of the shortcut. If more than one shortcut of the minimal length exists you should output the one that begins earliest on the route. If more than one shortcut of the minimal length begins at the same break point you should output the one that ends furthest on the route.

Example

Sample input:

```
1
12
NNNENNNWWSSW
```

Sample output:

```
2 3 11 W
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-26

Time limit [s]: 30

Source limit [B]: 50000

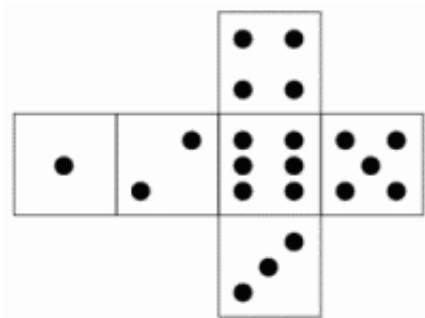
Resource: ACM Central European Programming Contest, Warsaw 2003

SPOJ Problem Set

85. Dice Contest

Problem code: DICE1

Everyone loves gambling in the Dicent City. Every Saturday the whole community meets to attend a dice contest. They started a few years ago with a classic six-sided die with 1 to 6 dots displayed on the sides and had a lot of fun.



However they soon got bored and that's why more sophisticated dice are in use nowadays. They put a sticker on each side and write a positive integer on each sticker.

The contest is run on a strip divided into squares in a chessboard-like manner. The strip is 4 squares wide and infinite to the left and to the right (is anyone going to say it can't exist in the real world, huh?). The rows of the strip are numbered from 1 to 4 from the bottom to the top and the columns are numbered by consecutive integers from the left to the right. Each square is identified by a pair (x,y) where x is a column number and y is a row number.

The game begins with a die placed on a square chosen by a contest committee with one-dot side on the top and two-dots side facing the player. To move the die the player must roll the die over an edge to an adjacent (either horizontally or vertically) square. The number displayed on the top of the die after a roll is the cost of the move. The goal of the game is to roll the die from the starting square to the selected target square so that the sum of costs of all moves is minimal.

Task

Write a program that:

- reads the description of a die, a starting square and a target square,
- computes the minimal cost of rolling the die from the starting square to the target square,
- writes the result.

Note: all teams participating in the contest received dice from the organisers.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first line of the input contains six integers $l_1, l_2, l_3, l_4, l_5, l_6$ ($1 \leq l_i \leq 50$) separated by single spaces. Integer l_i is the number written on a side having originally i dots. The second line of the input contains four integers x_1, y_1, x_2, y_2 ($-10^9 \leq x_1, x_2 \leq 10^9, 1 \leq y_1, y_2 \leq 4$) separated by single spaces. Integers x_1, y_1 are the column and the row number of the starting square respectively. Integers x_2, y_2 are the column and the row number of the target square respectively.

Output

For each test case the first and the only line of the output should contain the minimal cost of rolling the die from the starting square to the target square.

Example

Sample input:

```
1
1 2 8 3 1 4
-1 1 0 2
```

Sample output:

```
7
```

Added by: Adrian Kosowski

Date: 2004-06-26

Time limit [s]: 30

Source limit [B]: 50000

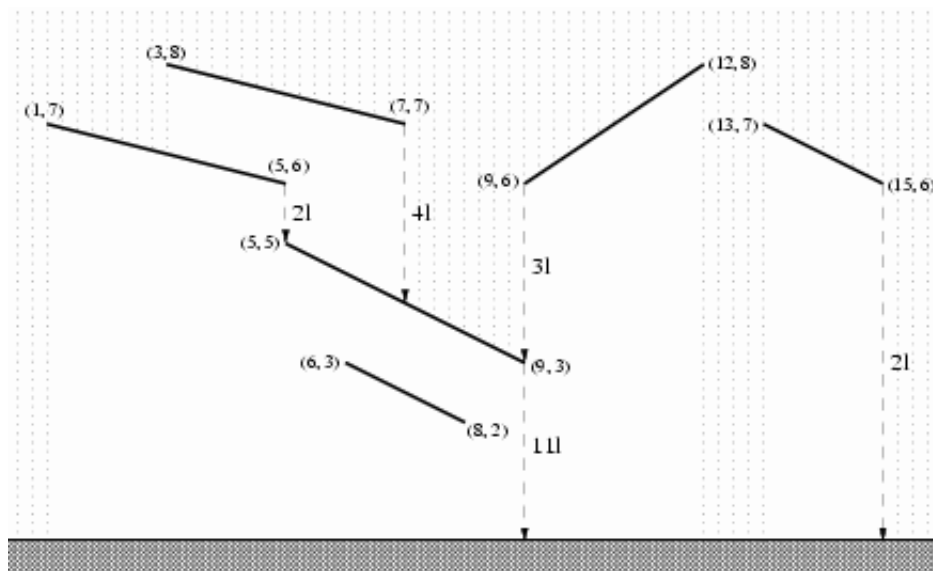
Resource: ACM Central European Programming Contest, Warsaw 2003

SPOJ Problem Set

86. November Rain

Problem code: RAIN1

Contemporary buildings can have very complicated roofs. If we take a vertical section of such a roof it results in a number of sloping segments. When it is raining the drops are falling down on the roof straight from the sky above. Some segments are completely exposed to the rain but there may be some segments partially or even completely shielded by other segments. All the water falling onto a segment as a stream straight down from the lower end of the segment on the ground or possibly onto some other segment. In particular, if a stream of water is falling on an end of a segment then we consider it to be collected by this segment.



For the purpose of designing a piping system it is desired to compute how much water is down from each segment of the roof. To be prepared for a heavy November rain you should count one liter of rain water falling on a meter of the horizontal plane during one second.

Task

Write a program that:

- reads the description of a roof,
- computes the amount of water down in one second from each segment of the roof,
- writes the results.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first line of the input contains one integer n ($1 \leq n \leq 40000$) being the number of segments of the roof. Each of the next n lines describes one segment of the roof and contains four integers x_1, y_1, x_2, y_2 ($0 \leq x_1, y_1, x_2, y_2 \leq 1000000, x_1 < x_2, y_1 \leq y_2$) separated by single spaces. Integers x_1, y_1 are respectively the horizontal position and the height of the left end of the segment. Integers x_2, y_2 are respectively the horizontal position and the height of the right end of the segment. The segments don't have common points and there are no horizontal segments. You can also assume that there are at most 25 segments placed above any point on the ground level.

Output

For each test case the output consists of n lines. The i -th line should contain the amount of water (in liters) down from the i -th segment of the roof in one second.

Example

Sample input:

```
1
6
13 7 15 6
3 8 7 7
1 7 5 6
5 5 9 3
6 3 8 2
9 6 12 8
```

Sample output:

```
2
4
2
11
0
3
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-26

Time limit [s]: 30

Source limit [B]: 50000

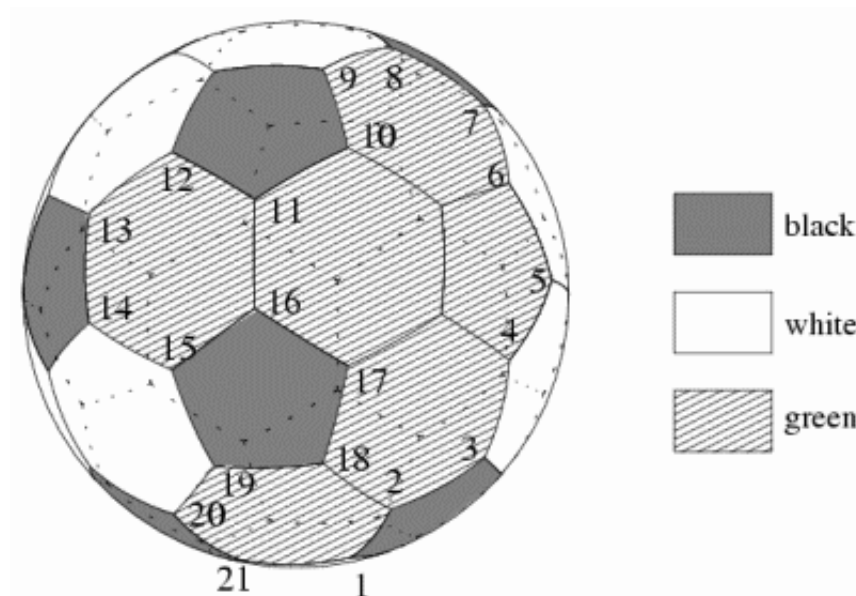
Resource: ACM Central European Programming Contest, Warsaw 2003

SPOJ Problem Set

87. Football

Problem code: FOOTBALL

Eric has a classic football that is made of 32 pieces of leather: 12 black pentagons and 20 white hexagons. Each pentagon adjoins 5 hexagons and each hexagon adjoins 3 pentagons and 3 hexagons. Eric drew a polygon (i.e. a closed line without intersections) along the edges of the pieces. The polygon divided the ball into two parts and Eric painted one of them green.



He is curious if given a description of the polygon you are able to compute the number of black, white and green pieces?

Task

Write a program that:

- reads the description of a polygon,
- computes the number of black, white and green pieces,
- writes the result.

Contest note: the first accepted solution will be awarded with the original football used for preparing the problem, signed by Eric, the author of the problem!

SPOJ note: the first accepted solution will be awarded some other sphere, without anybody's signatures, sent in PNG format to the author's email address [the offer is invalid, the sphere has already been presented to Robin Nittka, University of Ulm, Germany].

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case, the first line of the input contains one integer n being the number of vertices of the polygon. The second line of the input contains n integers a_1, a_2, \dots, a_n separated by single spaces. Integer a_i (equal 1 or 2) is the number of green pieces adjoining the i -th vertex of the polygon. The side of the polygon connecting the n -th and the first vertex always lies between two hexagons.

Output

For each test case the first and only line of the output contains three integers b , w and g - the numbers of black, white and green pieces respectively.

Example

Sample input:

```
1
21
1 2 1 2 1 2 1 1 1 2 2 1 1 1 1 2 2 2 1 1 1
```

Sample output:

```
11 15 6
```

Added by: Adrian Kosowski

Date: 2004-06-26

Time limit [s]: 3

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Warsaw 2003 (E. Kopczynski)

SPOJ Problem Set

88. Which is Next

Problem code: TREE2

Every computer science student knows binary trees. Here is one of many possible definitions of binary trees. Binary trees are defined inductively. A binary tree t is either an external node (leaf) \circ or an ordered pair $t = (t_1, t_2)$ representing an internal node \bullet with two subtrees attached, left subtree t_1 and right subtree t_2 . Under this definition the number of nodes in any binary tree is odd. Given an odd integer n let $B(n)$ denote the set of all binary trees with n nodes, both internal and external. For instance $B(1)$ consists of only one tree \circ , $B(3) = \{(\circ, \circ)\}$ and $B(5) = \{(\circ, (\circ, \circ)), ((\circ, \circ), \circ)\}$. The trees of $B(5)$ are depicted in the figure below.



Denote by $|t|$ the number of nodes in a tree t . Given a tree t we define its unique integer identifier $N(t)$ as follows:

- $N(\circ) = 0$
- $N(t_1, t_2) = 2^{|t_1|+|t_2|} + 2^{|t_2|} * N(t_1) + N(t_2)$

For instance, $N(\circ, \circ) = 2^2 + 2^1 * 0 + 0 = 4$, $N(\circ, (\circ, \circ)) = 2^4 + 2^3 * 0 + 4 = 20$,
 $N((\circ, \circ), \circ) = 2^4 + 2^1 * 4 + 0 = 24$.

Consider the following linear order on all binary trees:

- 1) $\circ \leq t$
- 2) $(t_1, t_2) \leq (u_1, u_2)$ when $t_1 < u_1$, or $t_1 = u_1$ and $t_2 \leq u_2$

In this order a single leaf \circ is the smallest tree and given two nonleaf trees, the smaller one is that with the smaller left tree, if the left subtrees are different, and that with the smaller right subtree, otherwise. Hence for instance $(\circ, (\circ, \circ)) < ((\circ, \circ), \circ)$, since we have $\circ < (\circ, \circ)$. Assume now that the trees in $B(n)$ were sorted using the relation \leq . Then, for each tree t in $B(n)$ we define the successor of t as the tree that immediately follows t in $B(n)$. If t is the largest one in $B(n)$ then the successor of t is the smallest tree in set $B(n)$. For instance, the successor of (\circ, \circ) in $B(3)$ is the same tree (\circ, \circ) and the successor of $(\circ, (\circ, \circ))$ in $B(5)$ is $((\circ, \circ), \circ)$. Given the integer identifier of some tree t can you give the identifier of the successor of t in $B(|t|)$?

Task

Write a program that:

- reads the identifier of some binary tree t ,
- computes the identifier of the successor of t in $B(|t|)$,
- writes the result.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first and only line of the input contains one integer n ($0 \leq n < 2^{30}$) - the identifier of some binary tree t .

Output

For each test case the first and only line of the output should contain one integer s - the identifier of the successor of t in $B(|t|)$.

Example

Sample input:

1
20

Sample output:

24

Added by: Adrian Kosowski

Date: 2004-06-26

Time limit [s]: 3

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Warsaw 2003

SPOJ Problem Set

89. Hang or not to hang

Problem code: HANGLET

Little Tom is learning how to program. He has just written some programs but is afraid to run them, because he does not know if they will ever stop. Please write a program to help him. This task is not as easy as it may seem, because Tom's programs are possibly not deterministic. Given a program written by Tom, your program should tell him whether his program can stop and if so, what is the shortest possible time before it stops.

Tom's computer consists of 32 1-bit registers and the program consists of n instructions. The registers are numbered from 0 to 31 and the instructions are numbered from 0 to $n-1$.

Below, $\text{MEM}[a]$ stands for the contents of the a -th register, $0 \leq a, b < 32$, $0 \leq x < n$, $0 \leq c \leq 1$.

The instruction set is as follows:

Instruction	Semantics
AND a b	$\text{MEM}[a] := \text{MEM}[a] \text{ and } \text{MEM}[b]$
OR a b	$\text{MEM}[a] := \text{MEM}[a] \text{ or } \text{MEM}[b]$
XOR a b	$\text{MEM}[a] := \text{MEM}[a] \text{ xor } \text{MEM}[b]$
NOT a	$\text{MEM}[a] := \text{not } \text{MEM}[a]$
MOV a b	$\text{MEM}[a] := \text{MEM}[b]$
SET a c	$\text{MEM}[a] := c$
RANDOM a	$\text{MEM}[a] := \text{random value (0 or 1)}$
JMP x	jump to the instruction with the number x
JZ x a	jump to the instruction with the number x if $\text{MEM}[a] = 0$
STOP	stop the program

The last instruction of a program is always STOP (although there can be more than one STOP instruction). Every program starts with the instruction number 0. Before the start, the contents of the registers can be arbitrary values. Each instruction (including STOP) takes 1 processor cycle to execute.

Task

Write a program that:

- reads the program,
- computes the shortest possible running time of the program,
- writes the result.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first line of the input contains an integer n ($1 \leq n \leq 16$) being the number of instructions of the program. Each of the next n lines contains one instruction of the program in the format given above. You may assume that the only white characters in the program are single spaces between successive tokens of each instruction.

Output

For each test case the first and only line of the output should contain the shortest possible running time of the program, measured in processor cycles. If the program cannot stop, output should contain the word HANGS.

Example

Sample input:

```
2
5
SET 0 1
JZ 4 0
RANDOM 0
JMP 1
STOP
5
MOV 3 5
NOT 3
AND 3 5
JZ 0 3
STOP
```

Sample output:

```
6
HANGS
```

Added by: Adrian Kosowski

Date: 2004-06-26

Time limit [s]: 5

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Warsaw 2003

SPOJ Problem Set

90. Minimizing maximizer

Problem code: MINIMAX

The company Chris Ltd. is preparing a new sorting hardware called Maximizer. Maximizer has n inputs numbered from 1 to n . Each input represents one integer. Maximizer has one output which represents the maximum value present on Maximizer's inputs.

Maximizer is implemented as a pipeline of sorters $\text{Sorter}(i_1, j_1), \dots, \text{Sorter}(i_k, j_k)$. Each sorter has n inputs and n outputs. $\text{Sorter}(i, j)$ sorts values on inputs $i, i+1, \dots, j$ in non-decreasing order and lets the other inputs pass through unchanged. The n -th output of the last sorter is the output of the Maximizer.

An intern (a former ACM contestant) observed that some sorters could be excluded from the pipeline and Maximizer would still produce the correct result. What is the length of the shortest subsequence of the given sequence of sorters in the pipeline still producing correct results for all possible combinations of input values?

Task

Write a program that:

- reads a description of a Maximizer, i.e. the initial sequence of sorters in the pipeline,
- computes the length of the shortest subsequence of the initial sequence of sorters still producing correct results for all possible input data,
- writes the result.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first line of the input contains two integers n and m ($2 \leq n \leq 50000$, $1 \leq m \leq 500000$) separated by a single space. Integer n is the number of inputs and integer m is the number of sorters in the pipeline. The initial sequence of sorters is described in the next m lines. The k -th of these lines contains the parameters of the k -th sorter: two integers i_k and j_k ($1 \leq i_k < j_k \leq n$) separated by a single space.

Output

For each test case the output consists of only one line containing an integer equal to the length of the shortest subsequence of the initial sequence of sorters still producing correct results for all possible data.

Example

Sample input:

```
1
40 6
20 30
1 10
10 20
20 30
15 25
30 40
```

Sample output:

```
4
```

Warning: enormous Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-06-26

Time limit [s]: 30

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Warsaw 2003

SPOJ Problem Set

91. Two squares or not two squares

Problem code: TWOSQRS

Given integer n decide if it is possible to represent it as a sum of two squares of integers.

Input

First line of input contains one integer $c \leq 100$ - number of test cases. Then c lines follow, each of them consisting of exactly one integer $0 \leq n \leq 10^{12}$.

Output

For each test case output Yes if it is possible to represent given number as a sum of two squares and No if it is not possible.

Example

Input :

```
10
1
2
7
14
49
9
17
76
2888
27
```

Output :

```
Yes
Yes
No
No
Yes
Yes
Yes
No
Yes
No
```

Added by: Pawel Gawrychowski

Date: 2004-06-29

Time limit [s]: 3

Source limit [B]: 50000

SPOJ Problem Set

92. Cutting off Squares

Problem code: CUTSQRS

Problem

Two players take it in turns to cut off squares from a rectangle. If the lengths of the sides of the rectangle are a and b ($a \leq b$) at the beginning of a player's turn, he may cut off as many squares with a side of length a as he likes (but at least 1 square), provided the square he is cutting off has at least three of its sides lying on the sides of the rectangle he is trimming. After every cut, the cut off square is removed from the rectangle. When the last part of the rectangle is removed, the game ends and the person who cut it off wins.

Michael, a friend of the players', is taking down a log of the games they are playing in the form of a sequence of consecutive numbers, each number denoting how many squares a player cut off in his turn. Since the game is rather slow, Michael is getting a little bored and he has started writing a detailed analysis of the game in his notebook. For given starting dimensions a and b , he always writes down:

- the number of different possible game sequences,
- the number of different possible game sequences in which the starting player wins,
- the word 'first' if the starting player can win (provided he does not make any mistakes) regardless of what the other player does, and the word 'second' in all other cases.

After writing for several hours Michael began to worry whether he had enough room left in his notebook for all the information he wanted to write down. Please help him answer this question.

Input

An integer t denoting the number of test cases, ($t \leq 10000$) followed by t pairs of integers a , b , ($1 \leq a \leq b \leq 10^9$) given in separate lines.

Output

For each test case, output the number of characters Michael has to write down (excluding spaces).

Example

Sample input:

```
2
1 1
2 3
```

Sample output:

```
7
8
```

(In the first case Michael has to write '1 1 first', in the second case '2 1 second'.)

Added by: Adrian Kosowski

Date: 2004-06-22

Time limit [s]: 5

Source limit [B]:50000

Resource: DASM Programming League 2004 (problemset 1)


SPOJ Problem Set

94. Numeral System of the Maya

Problem code: MAYA

The Maya lived in Central America during the first millennium. In many regards, they constituted one of the most developed and most fascinating cultures of this epoch. Even though draught animals and the wheel were unknown to the Mayas, they excelled in the fields of weaving, architecture and pottery. But truly breath-taking were their achievements in the fields of astronomy and mathematics. Whilst Europe was trudging through the dark Middle Ages, the Maya determined the solar year to 365.242 days (modern-day measurement: 365.242198) and the lunar cycle to 29.5302 days (modern-day measurement: 29.53059). Such astonishingly precise findings were hardly possible without a powerful numeral system. In this task we will explore the Maya's numeral system.

Maya priests and astronomers used a numerical system to the base of 20. Unusual to their time, their system also included the concepts of digits and of the zero. Both concepts were completely unknown to the Europeans at this time. The first nineteen numbers of the vigesimal system were represented by dots and dashes according to the following table:

0	1	2	3	4	5	6	7	8	9
	—	—.	—.	—.	—.
10	11	12	13	14	15	16	17	18	19
==	==.	==.	==.	==.	==	==.	==.	==.	==.

The zero was written down as a symbol resembling a shell. Multi-digit numbers (i.e. the numbers bigger than 19) were written in vertical arrangement, with the highest-value digit on top. For example, the number 79 was written as



As can be seen, the second digit possesses a value of 20.

Due to an interference of the two calendar systems of the Maya, the third digit did not hold the value 400 (20×20), as would be expected, but 360. All the following digits were again treated regularly, i.e. the fourth digit counted 7200 (360×20), the fifth 144000 (7200×20), and so on.

Hence, the number 13495 ($=1 \times 7200 + 17 \times 360 + 8 \times 20 + 15$) was written as follows:



Write a program to convert Maya numbers to decimal numbers!

Input

The input file contains a list of numbers written down in Maya fashion. Of course, dots are represented as points (.), and dashes are represented as hyphens (-). The zero digit, the shell symbol, is written as a capital letter S (S). Description of a Maya number starts with n - the number of the Maya digits. The following n lines contain one digit each. One digit is written from top to bottom using spaces as vertical separators.

One number will not have more than seven digits. Each two numbers are separated by a blank line. Input terminates with $n = 0$

Output

Your program has to output the value of the number in the input file in the nowadays more common decimal system. One number per line.

Example

Sample input:

1
..

5
... -
. - -
S
S
S

0

Sample output:

2
1231200

Added by: Michal Czuczman
Date: 2004-07-11
Time limit [s]: 5
Source limit [B]: 50000
Resource: Swiss Olympiad in Informatics 2004

SPOJ Problem Set

95. Street Parade

Problem code: STPAR

For sure, the love mobiles will roll again on this summer's street parade. Each year, the organisers decide on a fixed order for the decorated trucks. Experience taught them to keep free a side street to be able to bring the trucks into order.

The side street is so narrow that no two cars can pass each other. Thus, the love mobile that enters the side street last must necessarily leave the side street first. Because the trucks and the ravers move up closely, a truck cannot drive back and re-enter the side street or the approach street.

You are given the order in which the love mobiles arrive. Write a program that decides if the love mobiles can be brought into the order that the organisers want them to be.

Input

There are several test cases. The first line of each test case contains a single number n , the number of love mobiles. The second line contains the numbers 1 to n in an arbitrary order. All the numbers are separated by single spaces. These numbers indicate the order in which the trucks arrive in the approach street. No more than 1000 love mobiles participate in the street parade. Input ends with number 0.

Output

Your program has to output for each test case "yes" (without the quotes, of course), if the love mobiles can be re-ordered with the help of the side street. The output must be "no", if this is not possible.

Example

Sample input:

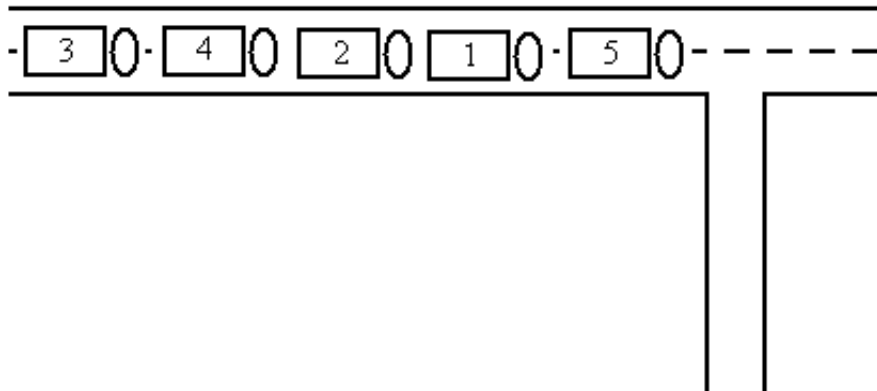
```
5
5 1 2 4 3
0
```

Sample output:

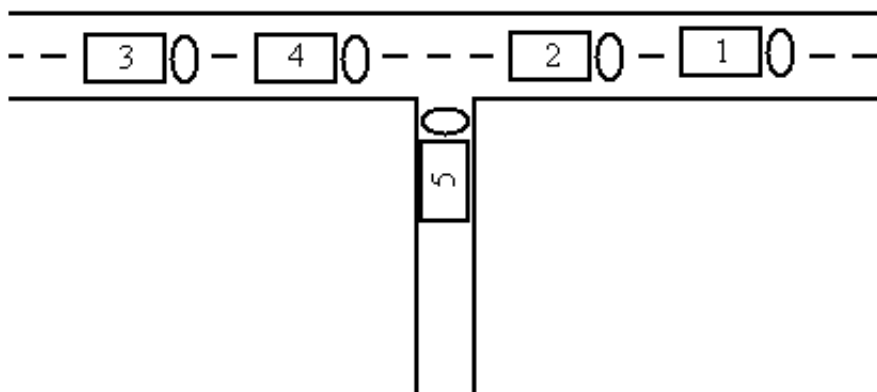
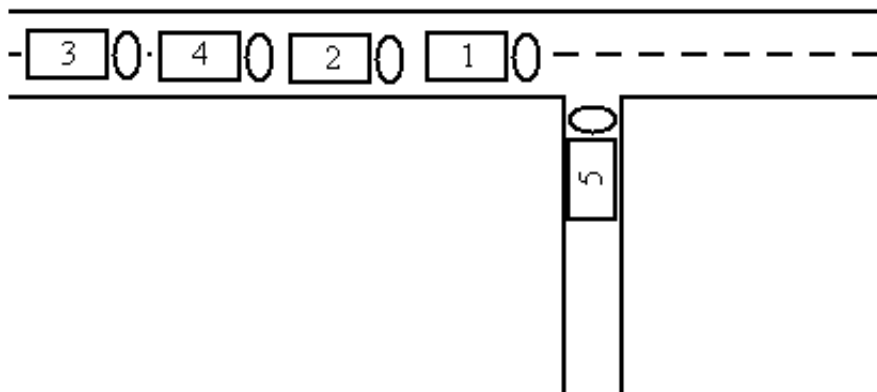
```
yes
```

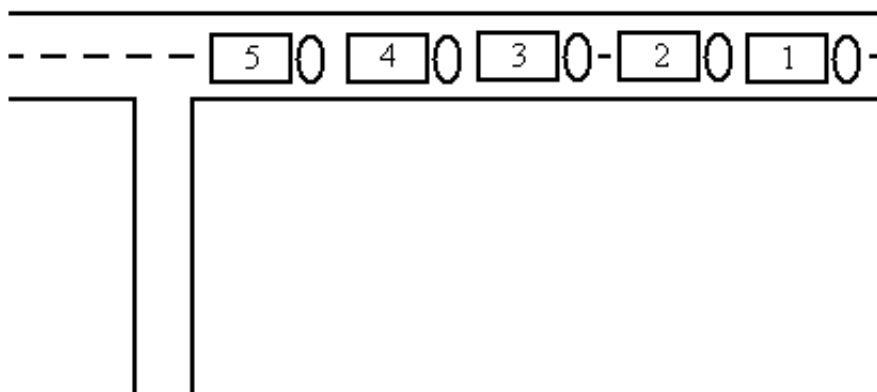
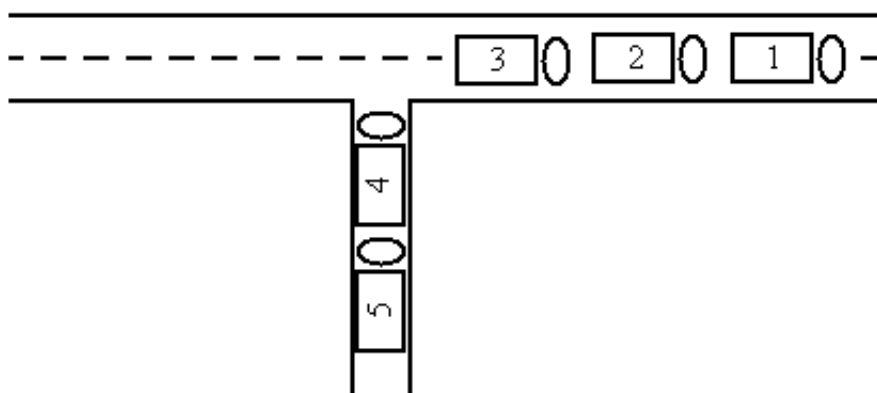
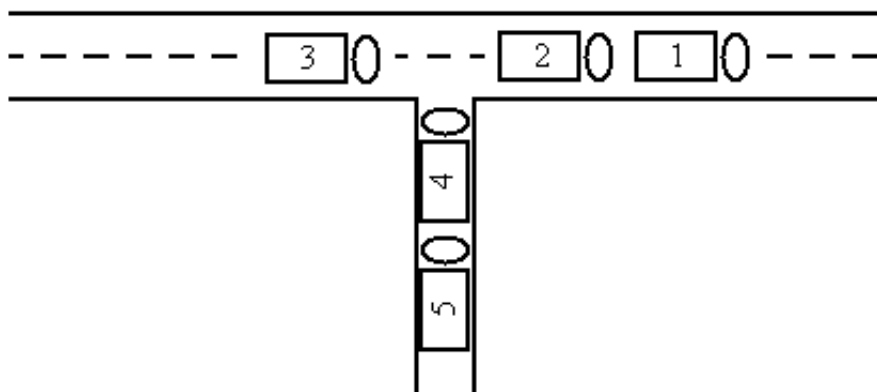
Illustration

The sample input reflects the following situation:



The five trucks can be re-ordered in the following way:





Added by: Patryk Pomykalski
 Date: 2004-07-01
 Time limit [s]: 3
 Source limit [B]: 50000
 Resource: Swiss Olympiad in Informatics 2004

SPOJ Problem Set

96. Shopping

Problem code: SHOP



The old tube screen to your computer turned out to be the cause of your chronic headaches. You therefore decide to buy one of these new flat TFT monitors. At the entrance of the computer shop you see that it is quite full with customers.

In fact, the shop is rather packed with customers and moving inside involves a certain amount of elbowing. Since you want to return home quickly to complete your half finished SPOJ tasks, you want to sidestep the crowd as much as possible. You examine the situation somewhat closer and realise that the crowding is less in some parts of the shop. Thus, there is reason for hope that you can reach your goal in due time, provided that you take the shortest way. But which way is the shortest way?

You sketch the situation on a piece of paper but even so, it is still a tricky affair. You take out your notebook from your pocket and start to write a program which will find the shortest way for you.

Input

The first line of the input specifies the width w and height h of the shop. Neither dimension exceeds 25.

The following h lines contain w characters each. A letter X symbolises a shelf, the letter S marks your starting position, and the letter D marks the destination (i.e. the square in front of the monitors). All free squares are marked with a digit from 1 to 9, meaning the number of seconds needed to pass this square.

There are many test cases separated by an empty line. Input terminates with width and height equal 0 0.

Output

Your program is to output the minimum number of seconds needed to reach to destination square. Each test case in a separate line. Movements can only be vertical and horizontal. Of course, all movements must take place inside the grid. There will always be a way to reach the destination.

Example

Sample input:

```
4 3
X1S3
42X4
X1D2
```

```
5 5
S5213
2X2X5
51248
4X4X2
1445D
```

```
0 0
```

Sample output:

```
4
23
```

Added by: Michal Czuczman

Date: 2004-07-01

Time limit [s]: 5

Source limit [B]:50000

Resource: Swiss Olympiad in Informatics 2004

SPOJ Problem Set

97. Party Schedule

Problem code: PARTY

You just received another bill which you cannot pay because you lack the money.

Unfortunately, this is not the first time to happen, and now you decide to investigate the cause of your constant monetary shortness. The reason is quite obvious: the lion's share of your money routinely disappears at the entrance of party localities.

You make up your mind to solve the problem where it arises, namely at the parties themselves. You introduce a limit for your party budget and try to have the most possible fun with regard to this limit.

You inquire beforehand about the entrance fee to each party and estimate how much fun you might have there. The list is readily compiled, but how do you actually pick the parties that give you the most fun and do not exceed your budget?

Write a program which finds this optimal set of parties that offer the most fun. Keep in mind that your budget need not necessarily be reached exactly. Achieve the highest possible fun level, and do not spend more money than is absolutely necessary.

Input

The first line of the input specifies your party budget and the number n of parties.

The following n lines contain two numbers each. The first number indicates the entrance fee of each party. Parties cost between 5 and 25 francs. The second number indicates the amount of fun of each party, given as an integer number ranging from 0 to 10.

The budget will not exceed 500 and there will be at most 100 parties. All numbers are separated by a single space.

There are many test cases. Input ends with 0 0.

Output

For each test case your program must output the sum of the entrance fees and the sum of all fun values of an optimal solution. Both numbers must be separated by a single space.

Example

Sample input:

```
50 10
12 3
15 8
16 9
16 6
10 2
21 9
18 4
12 4
17 8
18 9

50 10
```

13 8
19 10
16 8
12 9
10 2
12 8
13 5
15 5
11 7
16 2

0 0

Sample output:

49 26
48 32

Added by: Patryk Pomykalski
Date: 2004-07-01
Time limit [s]: 5
Source limit [B]:50000
Resource: Swiss Olympiad in Informatics 2004

SPOJ Problem Set

98. Dance Floor

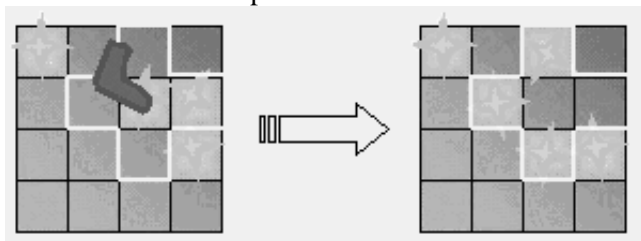
Problem code: DFLOOR

You recently watched a video clip in which a singer danced on a grid of colourful tiles enlightened from below. Each step on a tile flipped the tile's state, i.e. light on or off. In addition to that, all the neighbouring tiles flipped their states, too.

In this task, you are supposed to come up with a short program that decides if it is possible for the singer to switch on the lights of all the tiles, provided that he dances on the appropriate tiles.

The dance floor has rectangular shape. At the beginning, some of the tiles are already alight. Your program may temporarily switch off some tiles, if it deems that necessary to reach its goal. Stepping on a tile toggles its own state as well as the states of the four neighbouring tiles directly above, below, to the left and to the right. Of course, in the case of a peripheral tile, there will be only three or two neighbouring tiles.

Here comes an example:



If the dancer steps on the tile indicated by the brown shoe, all the tiles within the white area change their states. The resulting dance floor is depicted on the right.

You may assume that the singer is fit enough to jump from any tile to any other tile, even if the destination tile lies on the opposite side of the dance floor.

Input

There are several test cases. The first line of each case contains two integer numbers x and y , indicating the width and the height of the dance floor grid. The numbers are separated by a single space and satisfy $3 \leq x, y \leq 15$.

The following y lines containing x characters each describe the initial on/off states of the tiles. A zero means "the tile is switched off", a one digit means "the tile is alight".

Input ends with 0 0.

Output

For each test case your program should output the number of steps needed to switch all the lights on, followed by exactly that many lines with two space-separated numbers i and j . Each individual line commands the singer to step on the i -th tile of the j -th row. Starting with the situation of the input file and executing all the commands in the output file, all the tiles must be switched on.

If more than one solution exist, your program should output an arbitrary one of them. If, on the other hand, no solution exists, your program should write the number "-1".

Example

Sample input

```
4 3
0111
1010
1000
```

```
0 0
```

Sample output

```
3
1 2
1 3
4 3
```

Added by: Michal Czuczman

Date: 2004-07-01

Time limit [s]: 5

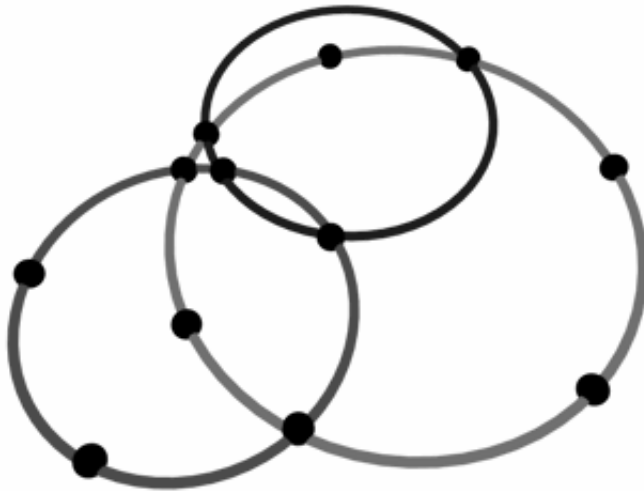
Source limit [B]:50000

Resource: Swiss Olympiad in Informatics 2004

SPOJ Problem Set

99. Bus

Problem code: BUS



The city Buscelona (as the name suggests) has a great bus transport system. All buses have circular lines. The bus drivers in Buscelona like to chat. Fortunately most bus lines have some stops in common. If a bus driver meets a colleague on a bus stop they chat a bit and exchange all news they know.

The operation of buses is highly synchronized. The time necessary to get from one stop to the next stop is always exactly 1 minute.

Each morning each bus driver has some important news that only he knows. When a busdriver meets a colleague he will tell him all news he knows. If two bus drivers share the same start station, they will exchange their news there already (before they start working). Note that exchanging news and stopping does not take any time.

Input

The first line of a test case contains the number of bus lines n ($0 < n < 50$). The following n lines start with a number s ($0 < s < 50$) indicating the stops of a busline. On the same line follow s numbers representing a bus station each. A bus starts at the first station. When a bus reaches the last station, the bus will drive to the first station again.

There are many test cases separated by an empty line. Input data terminates with $n = 0$.

Output

For each test case you should output the time in minutes which it takes until all bus drivers know all news. If that never happens, your program should write the word "NEVER" (without quotes).

Example

Sample input:

```
3
3 1 2 3
3 2 3 1
4 2 3 4 5
```

```
2
2 1 2
2 5 8
```

```
0
```

Sample output:

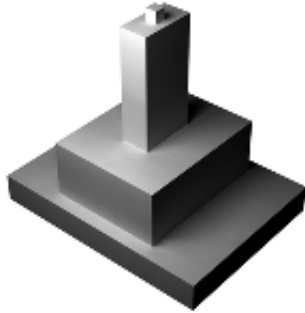
```
12
NEVER
```

Added by: Michal Czuczman
Date: 2004-07-03
Time limit [s]: 15
Source limit [B]: 50000
Resource: Swiss Olympiad in Informatics 2004

SPOJ Problem Set

100. Tower of Babylon

Problem code: **BABTWR**



Apart from the Hanging Gardens the Babylonians (around 3000-539 b.c.) built the Tower of Babylon as well. The tower was meant to reach the sky, but the project failed because of a confusion of language imposed from much higher above.

For the 2638th anniversary a model of the tower will be rebuilt. n different types of blocks are available. Each one of them may be duplicated as many times as you like. Each type has a height y , a width x and a depth z . The blocks are to be stacked one upon each other so that the resulting tower is as high as possible. Of course the blocks can be rotated as desired before stacking. However for reasons of stability a block can only be stacked upon another if *both* of its baselines are shorter.

Input

The number of types of blocks n is located in the first line of each test case. On the subsequent n lines the height y_i , the width x_i and the depth z_i of each type of blocks are given. There are never more than 30 different types available.

There are many test cases, which come one by one. Input terminates with $n = 0$.

Output

For each test case your program should output one line with the height of the highest possible tower.

Example

Sample input:

```
5
31 41 59
26 53 58
97 93 23
84 62 64
33 83 27
1
1 1 1
0
```

Sample output:

342

1

Added by: Michal Czuczman

Date: 2004-07-06

Time limit [s]: 5

Source limit [B]:50000

Resource: Swiss Olympiad in Informatics 2004

SPOJ Problem Set

101. Fishmonger

Problem code: FISHER



A fishmonger wants to bring his goods from the port to the market. On his route he has to traverse an area with many tiny city states. Of course he has to pay a toll at each border.

Because he is a good business man, he wants to choose the route in such a way that he has to pay as little money for tolls as possible. On the other hand, he has to be at the market within a certain time, otherwise his fish start to smell.

Input

The first line contains the number of states n and available time t . The first state is the port, the last state is the market. After this line there are n lines with n numbers each, specifying for each state the travel time to the i -th state. This table is terminated with an empty line. The table of the tolls follows in the same format.

n is at least 3 and at most 50. The time available is less than 1000. All numbers are integers.

There are many test cases separated by an empty line. Input terminates with number of states and time equal 0 0.

Output

For each test case your program should print on one line the total amount of tolls followed by the actual travelling time.

Example

Sample input:

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

0 2 2 7
2 0 1 2
```

```

2 2 0 5
7 2 5 0

0 0

```

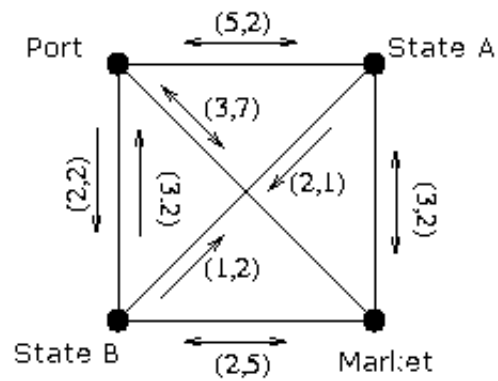
Sample output:

```

6 6

```

This corresponds to the following situation, the connections are labeled with (time, toll):



Added by: Michal Czuczman
 Date: 2004-07-07
 Time limit [s]: 5
 Source limit [B]: 50000
 Resource: Swiss Olympiad in Informatics 2004

SPOJ Problem Set

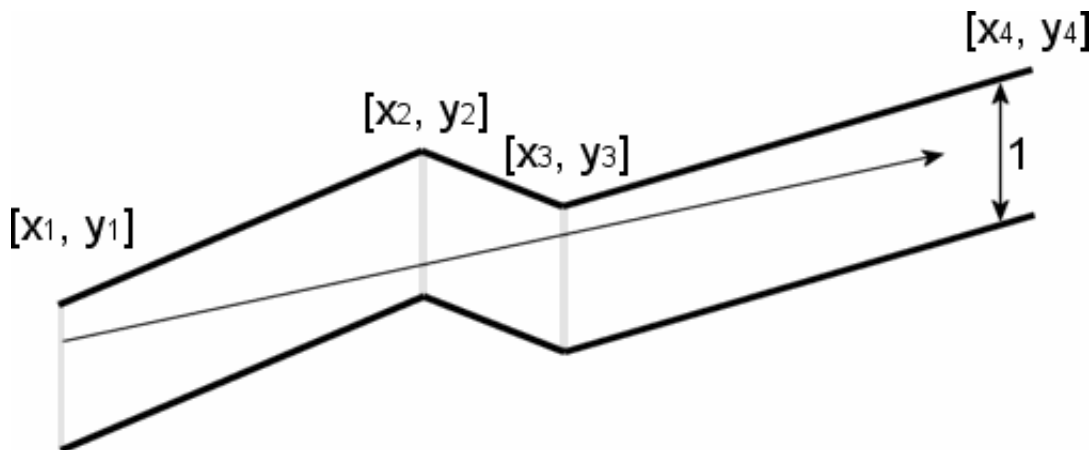
102. GX Light Pipeline Inc

Problem code: LITEPIPE

The GX Light Pipeline Inc. started to prepare bent pipes for the new transgalactic light pipeline. However during the design of the pipeline they ran into the problem of determining how far the light can reach inside the pipe. In order to improve your scarce budget you decided to fill a summer job at the GX Light Pipeline Inc. Now it's your task to create a program which computes how far the light reaches in the pipeline.

The pipeline consists of seamlessly welded together segments made of non-reflecting opaque materials. The upper points of the pipe contour are described by a sequence of points $[x_1, y_1]$, $[x_2, y_2]$, $[x_3, y_3]$, ..., $[x_n, y_n]$, where $x_k < x_{k+1}$. The bottom points of the pipe contour are the same points with y -coordinate decreased by 1.

The company wants to find the points with maximal x -coordinate that the light will reach. The light is emitted by a segment source with endpoints $[x_1, y_1]$ and $[x_1, y_1 - 1]$ (endpoints are emitting light too). Assume that the light is not bent at the pipe bent points and the bent points do not stop the light beam.



Input

Each test case starts with the number of bent points n . Each of the next n lines contains a pair of real values x_i, y_i separated by space.

The number of bent points never exceeds 200.

There are many test cases. Input terminates with $n = 0$.

Output

For each test case your program should output on a single line the maximal x -coordinate of the point where the light can reach from the source segment, written with precision of two decimal places. If the light goes through all the pipe, your program should output x_n .

Example

Sample input:

```
4
0.00 1.00
2.00 2.00
4.00 1.00
6.00 4.00
0
```

Sample output:

```
4.67
```

Added by: Michal Czuczman

Date: 2004-07-11

Time limit [s]: 5

Source limit [B]: 50000

Resource: Swiss Olympiad in Informatics 2004

SPOJ Problem Set

104. Highways

Problem code: HIGH

In some countries building highways takes a lot of time... Maybe that's because there are many possibilities to construct a network of highways and engineers can't make up their minds which one to choose. Suppose we have a list of cities that can be connected directly. Your task is to count how many ways there are to build such a network that between every two cities there exists exactly one path. Two networks differ if there are two cities that are connected directly in the first case and aren't in the second case. At most one highway connects two cities. No highway connects a city to itself. Highways are two-way.

Input

The input begins with the integer t , the number of test cases (equal to about 1000). Then t test cases follow. The first line of each test case contains two integers, the number of cities ($1 \leq n \leq 12$) and the number of direct connections between them. Each next line contains two integers a and b , which are numbers of cities that can be connected. Cities are numbered from 1 to n . Consecutive test cases are separated with one blank line.

Output

The number of ways to build the network, for every test case in a separate line. Assume that when there is only one city, the answer should be 1. The answer will fit in a signed 64-bit integer.

Example

Sample input:

```
4
4 5
3 4
4 2
2 3
1 2
1 3

2 1
2 1

1 0

3 3
1 2
2 3
3 1
```

Sample output:

```
8
1
1
3
```

Added by: Piotr Lowiec
Date: 2004-07-02
Time limit [s]: 15
Source limit [B]:50000

SPOJ Problem Set

105. Alice and Bob

Problem code: ALICEBOB

This is a puzzle for two persons, let's say Alice and Bob. Alice draws an n -vertex convex polygon and numbers its vertices with integers $1, 2, \dots, n$ in an arbitrary way. Then she draws a number of noncrossing diagonals (the vertices of the polygon are not considered to be crossing points). She informs Bob about the sides and the diagonals of the polygon but not telling him which are which. Each side and diagonal is specified by its ends. Bob has to guess the order of the vertices on the border of the polygon. Help him solve the puzzle.

If $n = 4$ and $(1,3), (4,2), (1,2), (4,1), (2,3)$ are the ends of four sides and one diagonal then the order of the vertices on the border of this polygon is $1, 3, 2, 4$ (with the accuracy to shifting and reversing).

Task

Write a program that:

- reads the description of sides and diagonals given to Bob by Alice,
- computes the order of the vertices on the border of the polygon,
- writes the result.

Input

The first line of the input contains exactly one positive integer d equal to the number of data sets, $1 \leq d \leq 20$. The data sets follow.

Each data set consists of exactly two consecutive lines.

The first of those lines contains exactly two integers n and m separated by a single space, $3 \leq n \leq 1000$, $0 \leq m \leq n-3$. Integer n is the number of vertices of a polygon and integer m is the number of its diagonals, respectively.

The second of those lines contains exactly $2(m+n)$ integers separated by single spaces. Those are ends of all sides and some diagonals of the polygon. Integers a_j, b_j on positions $2j-1$ and $2j$, $1 \leq j \leq m+n$, $1 \leq a_j \leq n$, $1 \leq b_j \leq n$, $a_j \neq b_j$, specify ends of a side or a diagonal. The sides and the diagonals can be given in an arbitrary order. There are no duplicates. Alice does not cheat, i.e. the puzzle always has a solution.

Output

Line i , $1 \leq i \leq d$, should contain a sequence of n integers separated by single spaces - a permutation of $1, 2, \dots, n$, i.e. the numbers of subsequent vertices on the border of the polygon from the i -th data set, the sequence should always start from 1 and its second element should be the smaller vertex of the two border neighbours of vertex 1.

Example

Sample input:

```
1
4 1
1 3 4 2 1 2
4 1 2 3
```

Sample output:

```
1 3 2 4
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-07-02

Time limit [s]: 15

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Warsaw 2001

SPOJ Problem Set

106. Binary Stirling Numbers

Problem code: BINSTIRL

The Stirling number of the second kind $S(n, m)$ stands for the number of ways to partition a set of n things into m nonempty subsets. For example, there are seven ways to split a four-element set into two parts: $\{1, 2, 3\} \cup \{4\}$, $\{1, 2, 4\} \cup \{3\}$, $\{1, 3, 4\} \cup \{2\}$, $\{2, 3, 4\} \cup \{1\}$, $\{1, 2\} \cup \{3, 4\}$, $\{1, 3\} \cup \{2, 4\}$, $\{1, 4\} \cup \{2, 3\}$.

There is a recurrence which allows you to compute $S(n, m)$ for all m and n .

$$S(0, 0) = 1,$$

$$S(n, 0) = 0, \text{ for } n > 0,$$

$$S(0, m) = 0, \text{ for } m > 0,$$

$$S(n, m) = m \cdot S(n-1, m) + S(n-1, m-1), \text{ for } n, m > 0.$$

Your task is much "easier". Given integers n and m satisfying $1 \leq m \leq n$, compute the parity of $S(n, m)$, i.e. $S(n, m) \bmod 2$.

For instance, $S(4, 2) \bmod 2 = 1$.

Task

Write a program that:

- reads two positive integers n and m ,
- computes $S(n, m) \bmod 2$,
- writes the result.

Input

The first line of the input contains exactly one positive integer d equal to the number of data sets, $1 \leq d \leq 200$. The data sets follow.

Line $i + 1$ contains the i -th data set - exactly two integers n_i and m_i separated by a single space, $1 \leq m_i \leq n_i \leq 10^9$.

Output

The output should consist of exactly d lines, one line for each data set. Line i , $1 \leq i \leq d$, should contain 0 or 1, the value of $S(n_i, m_i) \bmod 2$.

Example

Sample input:

1

4 2

sample output:

1

Added by: Adrian Kosowski

Date: 2004-07-02

Time limit [s]: 5

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Warsaw 2001

SPOJ Problem Set

107. Calendar of the Maya

Problem code: MAYACAL

The Classical Maya civilization developed in what is today southern Mexico, Guatemala, Belize and northern Honduras. During its height they developed a sophisticated system for time keeping which they used both to record history and for divinatory rituals. Their calendar consisted of 3 components. the Tzolkin, the Haab and the Long Count.

For divinatory purposes the Maya used the Tzolkin which was composed of 20 day names to which numeric coefficients from 1 to 13 were attached giving a total of 260 distinct combinations. This is the size of the Tzolkin, or ritual, year. From Spanish colonial sources, we know the names of the days: Imix, Ik, Akbal, Kan, Chikchan, Kimi, Manik, Lamat, Muluk, Ok, Chuen, Eb, Ben, Ix, Men, Kib, Kaban, Etznab, Kawak, Ajaw. The sequence of days developed as follows (starting for example at 9 Imix):

9 Imix, 10 Ik, 11 Akbal, 12 Kan, 13 Chikchan, 1 Kimi, 2 Manik, ...

The Haab calendar was an astronomical one. It had 365 days divided into 19 months each with 20 days, except the last one which had only 5 days. In a manner similar to the Tzolkin each month name had a number from 1 to 20 indicating the day number within the month. Again, from Spanish colonial sources, we know the names of the months: Pohp, Wo, Sip, Zotz, Sek, Xul, Yaxkin, Mol, Chen, Yax, Sak, Keh, Mak, Kankin, Muan, Pax, Kayab, Kumku, Wayeb. The month Wayeb had just 5 days and was considered an unlucky time of the year.

The Tzolkin and Haab were combined in the inscriptions to create the Calendar Round, combining the 260 day cycle of the Tzolkin and the 365 day cycle of the Haab. A typical Calendar Round date in the inscriptions might be. 3 Lamat 6 Pax. Note that not all of the combination of days, months and coefficients are possible.

A typical sequence of days in the Calendar Round (starting for example at 3 Lamat 6 Pax):

3 Lamat 6 Pax, 4 Muluk 7 Pax, 5 Ok 8 Pax, 6 Chuen 9 Pax, 7 Eb 10 Pax,
8 Ben 11 Pax, 9 Ix 12 Pax, 10 Men 13 Pax, 11 Kib 14 Pax, 12 Kaban 15 Pax,
13 Etznab 16 Pax, 1 Kawak 17 Pax, 2 Ajaw 18 Pax, 3 Imix 19 Pax, 4 Ik 20 Pax,
5 Akbal 1 Kayab, 6 Kan 2 Kayab, ...

Finally, at the beginning of the Classic Period (AD 200 - 900), the Maya developed an absolute calendar called Long Count which counted the days from a fixed date in the past (the date when the current world was created according to Maya belief). Dates in the Long Count are given (for simplicity) in 5-tuples of the form. 9.2.3.4.5. Such a date one reads "9 baktuns 2 katuns 3 tuns 4 winals 5 kins since the zero date". A "kin" is just one day. A winal is a group of 20 days. A tun is a group of 18 winals (thus a tun has $20 \times 18 = 360$ days, 5 days short of a year). From here on all units come in multiples of 20. Thus a katun is equal to 20 tuns (almost 20 years) and a baktun means 20 katuns (almost 400 years). Thus 9.2.3.4.5 really means " $9 \times 144000 + 2 \times 7200 + 3 \times 360 + 4 \times 20 + 5$ days since the zero date". Note that for every Long Count date b.k.t.w.i we have $0 \leq k < 20$; $0 \leq t < 20$; $0 \leq w < 18$; $0 \leq i < 20$. Given the periodicity of the Calendar Round, a legal date such as 3 Lamat 6 Pax has multiple occurrences in the Long Count. Thus, one difficulty in reading inscriptions is in establishing a date for the inscription when the date is given only in terms of a Calendar Round (very common). In

this case one must compute "all" the possible Long Count dates associated with the particular Calendar Round and based in some other context information deduce (for example, the text mentions a king for which other dates are known) which one applies.

We limit our interest to the Long Count dates in the baktuns 8 and 9 (they cover all the Classic Period). We know that the Long Count date 8.0.0.0.0 fell on the Calendar Round 9 Ajaw 3 Sip.

Task

Write a program that:

- reads a Calendar Round date,
- computes all Long Count dates in the baktuns 8 and 9 for the given Calendar Round date if this date is legal,
- writes the result.

Input

The first line of the input contains exactly one positive integer d equal to the number of data sets, $1 \leq d \leq 30$. The data sets follow.

Each data set consists of exactly one line that contains exactly one Calendar Round date (maybe illegal). Tzolkin day number, Tzolkin day name, Haab day number and Haab month name separated by single spaces.

Output

For every data set your program must output an ascending sequence of Long Count dates computed for a given Calendar Round date. The first line of the output for the given input set should contain exactly one integer n equal to the length of the sequence (0, if the input date is illegal).

Each of the next n lines should contain exactly one Long Count date specified by exactly 5 integers (meaning the numbers of baktuns, katuns, tuns, winals and kins respectively) separated by single dots.

Example

Sample input:

```
2
3 Lamat 6 Pax
1 Ajaw 9 Chen
```

Sample output:

```
15
8.0.17.17.8
8.3.10.12.8
8.6.3.7.8
8.8.16.2.8
8.11.8.15.8
8.14.1.10.8
8.16.14.5.8
8.19.7.0.8
9.1.19.13.8
9.4.12.8.8
9.7.5.3.8
```


9.9.17.16.8
9.12.10.11.8
9.15.3.6.8
9.17.16.1.8
0

Added by: Adrian Kosowski
Date: 2004-07-02
Time limit [s]: 5
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Warsaw 2001

SPOJ Problem Set

108. Decoding Morse Sequences

Problem code: MORSE

Before the digital age, the most common "binary" code for radio communication was the Morse code. In Morse code, symbols are encoded as sequences of short and long pulses (called dots and dashes respectively). The following table reproduces the Morse code for the alphabet, where dots and dashes are represented as ASCII characters "." and "-":

A	.-	B	-...	C	-.-.	D	-..
E	.	F	..-.	G	--.	H
I	..	J	.-.-	K	-.-	L	.-..
M	--	N	-.	O	---	P	.-.-
Q	--.-	R	.-.	S	...	T	-
U	..-	V	...-	W	.-.-	X	-.--
Y	-.--	Z	--..				

Notice that in the absence of pauses between letters there might be multiple interpretations of a Morse sequence. For example, the sequence -.-.- could be decoded both as CAT or NXT (among others). A human Morse operator would use other context information (such as a language dictionary) to decide the appropriate decoding. But even provided with such dictionary one can obtain multiple phrases from a single Morse sequence.

Task

Write a program that:

- reads a Morse sequence and a list of words (a dictionary),
- computes the number of distinct phrases that can be obtained from the given Morse sequence using words from the dictionary,
- writes the result.

Notice that we are interested in full matches, i.e. the complete Morse sequence must be matched to words in the dictionary.

Input

The first line of the input contains exactly one positive integer d equal to the number of data sets, $1 \leq d \leq 20$. The data sets follow.

The first line of each data set contains a Morse sequence - a nonempty sequence of at most 10000 characters "." and "-" with no spaces in between.

The second line contains exactly one integer n , $1 \leq n \leq 10000$, equal to the number of words in a dictionary. Each of the following n lines contains one dictionary word - a nonempty sequence of at most 20 capital letters from "A" to "Z". No word occurs in the dictionary more than once.

Output

The output should consist of exactly d lines, one line for each data set. Line i should contain one integer equal to the number of distinct phrases into which the Morse sequence from the i -th data set can be parsed. You may assume that this number is at most $2 \cdot 10^9$ for every single data set.

Example

Sample input:

```
1
.---.---.---.---.---.
6
AT
TACK
TICK
ATTACK
DAWN
DUSK
```

Sample output:

```
2
```

Added by: Adrian Kosowski

Date: 2004-07-02

Time limit [s]: 15

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Warsaw 2001

SPOJ Problem Set

109. Exchanges

Problem code: EXCHNG

Given n integer registers r_1, r_2, \dots, r_n we define a Compare-Exchange Instruction $CE(a,b)$, where a, b are register indices ($1 \leq a < b \leq n$):

```
CE(a, b)::  
  if content( $r_a$ ) > content( $r_b$ ) then  
    exchange the contents of registers  $r_a$  and  $r_b$ ;
```

A Compare-Exchange program (shortly CE-program) is any finite sequence of Compare-Exchange instructions. A CE-program is called a Minimum-Finding program if after its execution the register r_1 always contains the smallest value among all values in the registers. Such a program is called reliable if it remains a Minimum-Finding program after removing any single Compare-Exchange instruction. Given a CE-program P , what is the smallest number of instructions that should be added at the end of program P in order to get a reliable Minimum-Finding program?

For instance, consider the following CE-program for 3 registers: $CE(1, 2)$, $CE(2, 3)$, $CE(1, 2)$. In order to make this program a reliable Minimum-Finding program it is sufficient to add only two instructions: $CE(1, 3)$ and $CE(1, 2)$.

Task

Write a program that:

- reads the description of a CE-program,
- computes the smallest number of CE-instructions that should be added to make this program a reliable Minimum-Finding program,
- writes the result.

Input

The first line of the input contains exactly one positive integer d equal to the number of data sets, $1 \leq d \leq 10$. The data sets follow.

Each data set consists of exactly two consecutive lines. The first of those lines contains exactly two integers n and m separated by a single space, $2 \leq n \leq 10000$, $0 \leq m \leq 25000$. Integer n is the number of registers and integer m is the number of program instructions.

The second of those lines contains exactly $2m$ integers separated by single spaces - the program itself. Integers a_j, b_j on positions $2j-1$ and $2j$, $1 \leq j \leq m$, $1 \leq a_j < b_j \leq n$, are parameters of the j -th instruction in the program.

Output

The output should consist of exactly d lines, one line for each data set. Line i , $1 \leq i \leq d$, should contain only one integer - the smallest number of instructions that should be added at the end of the i -th input program in order to make this program a reliable Minimum-Finding program.

Example

Sample input:

```
1
3 3
1 2 2 3 1 2
```

Sample output:

```
2
```

Added by: Adrian Kosowski

Date: 2004-07-02

Time limit [s]: 10

Source limit [B]: 50000

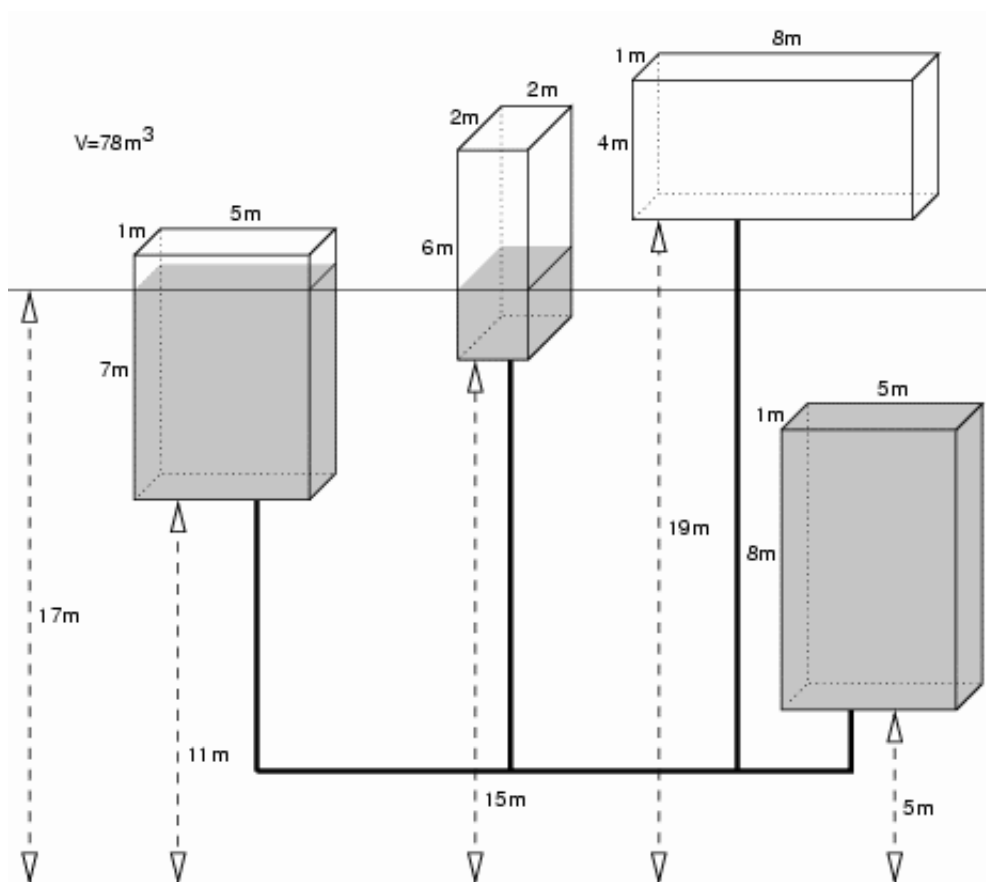
Resource: ACM Central European Programming Contest, Warsaw 2001

SPOJ Problem Set

110. Fill the Cisterns

Problem code: CISTFILL

During the next century certain regions on earth will experience severe water shortages. The old town of Uqbar has already started to prepare itself for the worst. Recently they created a network of pipes connecting the cisterns that distribute water in each neighbourhood, making it easier to fill them at once from a single source of water. But in case of water shortage the cisterns above a certain level will be empty since the water will flow to the cisterns below.



You have been asked to write a program to compute the level to which cisterns will be filled with a certain volume of water, given the dimensions and position of each cistern. To simplify we will neglect the volume of water in the pipes.

Task

Write a program that:

- reads the description of cisterns and the volume of water,
- computes the level to which the cisterns will be filled with the given amount of water,
- writes the result.

Input

The first line of the input contains the number of data sets k , $1 \leq k \leq 30$. The data sets follow.

The first line of each data set contains one integer n , the number of cisterns, $1 \leq n \leq 50000$. Each of the following n lines consists of 4 nonnegative integers, separated by single spaces: b, h, w, d - the base level of the cistern, its height, width and depth in meters, respectively. The integers satisfy $0 \leq b \leq 10^6$ and $1 \leq h*w*d \leq 40000$. The last line of the data set contains an integer V - the volume of water in cubic meters to be injected into the network. Integer V satisfies $1 \leq V \leq 2*10^9$.

Output

The output should consist of exactly d lines, one line for each data set. Line i , $1 \leq i \leq d$, should contain the level that the water will reach, in meters, rounded up to two fractional digits, or the word 'OVERFLOW', if the volume of water exceeds the total capacity of the cisterns.

Example

Sample input:

```
3
2
0 1 1 1
2 1 1 1
1
4
11 7 5 1
15 6 2 2
5 8 5 1
19 4 8 1
132
4
11 7 5 1
15 6 2 2
5 8 5 1
19 4 8 1
78
```

Sample output:

```
1.00
OVERFLOW
17.00
```

Warning: enormous Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-07-02

Time limit [s]: 30

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Warsaw 2001

SPOJ Problem Set

112. Horizontally Visible Segments

Problem code: SEGVIS

There is a number of disjoint vertical line segments in the plane. We say that two segments are horizontally visible if they can be connected by a horizontal line segment that does not have any common points with other vertical segments. Three different vertical segments are said to form a triangle of segments if each two of them are horizontally visible. How many triangles can be found in a given set of vertical segments?

Task

Write a program that:

- reads the description of a set of vertical segments,
- computes the number of triangles in this set,
- writes the result.

Input

The first line of the input contains exactly one positive integer d equal to the number of data sets, $1 \leq d \leq 20$. The data sets follow.

The first line of each data set contains exactly one integer n , $1 \leq n \leq 8000$, equal to the number of vertical line segments.

Each of the following n lines consists of exactly 3 nonnegative integers separated by single spaces: y'_i, y''_i, x_i (that is the y -coordinate of the beginning of a segment, y -coordinate of its end and its x -coordinate, respectively). The coordinates satisfy: $0 \leq y'_i < y''_i \leq 8000$, $0 \leq x_i \leq 8000$. The segments are disjoint.

Output

The output should consist of exactly d lines, one line for each data set. Line i should contain exactly one integer equal to the number of triangles in the i -th data set.

Example

Sample input:

```
1
5
0 4 4
0 3 1
3 4 2
0 2 2
0 2 3
```

Sample output:

```
1
```

Added by: Adrian Kosowski
Date: 2004-07-02
Time limit [s]: 30
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Warsaw 2001

SPOJ Problem Set

115. Family

Problem code: FAMILY

We want to find out how much related are the members of a family of monsters. Each monster has the same number of genes but the genes themselves may differ from monster to monster. It would be nice to know how many genes any two given monsters have in common. This is impossible, however, since the number of genes is very large. Still, we do know the family tree (well, not actually a tree, but you cannot really blame them, these are monsters, right?) and we do know how the genes are inherited so we can estimate the number of common genes quite well.

The inheritance rule is very simple: if a monster C is a child of monsters A and B then each gene of C is identical to the corresponding gene of either A or B, each with probability 50%. Every gene of every monster is inherited independently.

Let us define the degree of relationship of monsters X and Y as the expected number of common genes. For example consider a family consisting of two completely unrelated (i.e. having no common genes) monsters A and B and their two children C and D. How much are C and D related? Well, each of C's genes comes either from A or from B, both with probability 50%. The same is true for D. Thus, the probability of a given gene of C being the same as the corresponding gene of D is 50%. Therefore the degree of relationship of C and D (the expected number of common genes) is equal to 50% of all the genes. Note that the answer would be different if A and B were related. For if A and B had common genes, these would be necessarily inherited by both C and D.

Your task is to write a program that, given a family graph and a list of pairs of monsters, computes the degree of relationship for each of these pairs.

Task

Write a program that:

- reads the description of a family and a list of pairs of its members from the standard input,
- computes the degree of relationship (in percentages) for each pair on the list,
- writes the result to the standard output.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first line of the input contains two integers n and k separated by a single space. Integer n ($2 \leq n \leq 300$) is the number of members in a family. Family members are numbered arbitrarily from 1 to n . Integer k ($0 \leq k \leq n - 2$) is the number of monsters that do have parents (all the other monsters were created by gods and are completely unrelated to each other).

Each of the next k lines contains three different integers a , b , c separated by single spaces. The triple a , b , c means that the monster a is a child of monsters b and c .

The next input line contains an integer m ($1 \leq m \leq n^2$) - the number of pairs of monsters on the list. Each of the next m lines contains two integers separated by a single space - these are the numbers of two monsters.

You may assume that no monster is its own ancestor. You should not make any additional assumptions on the input data. In particular, you should not assume that there exists any valid sex assignment.

Output

For each test case the output consists of m lines. The i -th line corresponds to the i -th pair on the list and should contain single number followed by the percentage sign. The number should be the exact degree of relationship (in percentages) of the monsters in the i -th pair. Unsignificant zeroes are not allowed in the output (please note however that there must be at least one digit before the period sign so for example the leading zero in number 0.1 is significant and you cannot print it as .1). Confront the example output for the details of the output format.

Example

Sample input:

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

Sample output:

```
0%
50%
81.25%
100%
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-07-07

Time limit [s]: 15

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Warsaw 2002

SPOJ Problem Set

116. Intervals

Problem code: INTERVAL

You are given n closed integer intervals $[a_i, b_i]$ and n integers c_1, \dots, c_n .

Task

Write a program that:

- reads the number of intervals, their endpoints and integers c_1, \dots, c_n from the standard input,
- computes the minimal size of a set Z of integers which has at least c_i common elements with interval $[a_i, b_i]$, for each $i = 1, 2, \dots, n$,
- writes the answer to the standard output.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first line of the input contains an integer n ($1 \leq n \leq 50000$) - the number of intervals. The following n lines describe the intervals. Line $(i+1)$ of the input contains three integers a_i, b_i and c_i separated by single spaces and such that $0 \leq a_i \leq b_i \leq 50000$ and $1 \leq c_i \leq b_i - a_i + 1$.

Output

For each test case the output contains exactly one integer equal to the minimal size of set Z sharing at least c_i elements with interval $[a_i, b_i]$, for each $i = 1, 2, \dots, n$.

Example

Sample input:

```
1
5
3 7 3
8 10 3
6 8 1
1 3 1
10 11 1
```

Sample output:

```
6
```

Warning: enormous Input/Output data, be careful with certain languages

Added by: Adrian Kosowski
Date: 2004-07-07
Time limit [s]: 15
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Warsaw 2002

SPOJ Problem Set

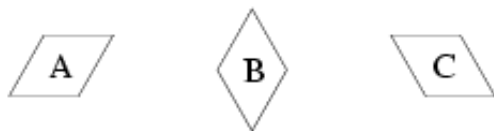
118. Rhombs

Problem code: RHOMBS

An unbounded triangular grid is a plane covered by equilateral triangles:

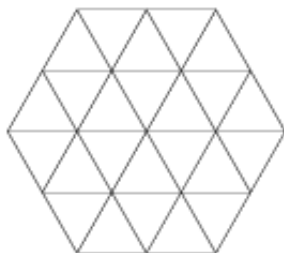


Two neighboring triangles in the grid form a rhomb. There are 3 types of such rhombs:

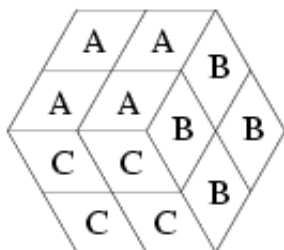


A grid polygon is a simple polygon which sides consist entirely of sides of triangles in the grid. We say that a grid polygon is rhombastic if it can be partitioned into internally disjoint rhombs of types A, B and C.

As an example let's consider the following grid hexagon:



This hexagon can be partitioned into 4 rhombs of type A, 4 rhombs of type B and 4 rhombs of type C:



For a given rhombastic grid polygon P compute the numbers of rhombs of types A, B and C in some correct partition.

Task

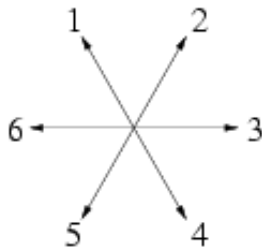
Write a program that:

- reads a description of a rhombastic grid polygon from the standard input,
- computes the numbers of rhombs of types A, B and C in some correct partition of the polygon,
- writes the results to the standard output.

Input

The input begins with the integer t, the number of test cases. Then t test cases follow.

For each test case the first line of the input contains an integer n ($3 \leq n \leq 50000$) - the number of sides of a rhombastic grid polygon. Each of the next n lines contains a description of one side of the polygon. The sides are given one by one in the clockwise order. No two consecutive sides of the polygon lie on the same straight line. The description of a side consists of two integers d and k. Integer d says what is the direction of the side according to the following figure:



Integer k is the length of the polygon side measured in the number of sides of grid triangles. Sum of all numbers k is not larger than 100000.

Output

For each test case the first and only line of the output contains three integers separated by single spaces denoting the number of rhombs of type A, B and C respectively, in some partition of the input polygon.

Example

Sample input:

```
1
6
1 2
2 2
3 2
4 2
5 2
6 2
```

Sample output:

```
4 4 4
```

Added by: Adrian Kosowski
Date: 2004-07-07
Time limit [s]: 15
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Warsaw 2002

SPOJ Problem Set

119. Servers

Problem code: SERVERS

The Kingdom of Byteland decided to develop a large computer network of servers offering various services.

The network is built of n servers connected by bidirectional wires. Two servers can be directly connected by at most one wire. Each server can be directly connected to at most 10 other servers and every two servers are connected with some path in the network. Each wire has a fixed positive data transmission time measured in milliseconds. The distance (in milliseconds) $D(V, W)$ between two servers V and W is defined as the length of the shortest (transmission time-wise) path connecting V and W in the network. For convenience we let $D(V, V) = 0$ for all V .

Some servers offer more services than others. Therefore each server V is marked with a natural number $r(V)$, called a rank. The bigger the rank the more powerful a server is.

At each server, data about nearby servers should be stored. However, not all servers are interesting. The data about distant servers with low ranks do not have to be stored. More specifically, a server W is interesting for a server V if for every server U such that $D(V, U) \leq D(V, W)$ we have $r(U) \leq r(W)$.

For example, all servers of the maximal rank are interesting to all servers. If a server V has the maximal rank, then exactly the servers of the maximal rank are interesting for V . Let $B(V)$ denote the set of servers interesting for a server V .

We want to compute the total amount of data about servers that need to be stored in the network being the total sum of sizes of all sets $B(V)$. The Kingdom of Byteland wanted the data to be quite small so it built the network in such a way that this sum does not exceed $30 \cdot n$.

Task

Write a program that:

- reads the description of a server network from the standard input,
- computes the total amount of data about servers that need to be stored in the network,
- writes the result to the standard output.

Input

The input begins with the integer z , the number of test cases. Then z test cases follow.

For each test case, in the first line there are two natural numbers n, m , where n is the number of servers in the network ($1 \leq n \leq 30000$) and m is the number of wires ($1 \leq m \leq 5n$). The numbers are separated by single space.

In the next n lines the ranks of the servers are given. Line i contains one integer r_i ($1 \leq r_i \leq 10$) - the rank of i -th server.

In the following m lines the wires are described. Each wire is described by three numbers a, b, t ($1 \leq t \leq 1000, 1 \leq a, b \leq n, a < b$), where a and b are numbers of the servers connected by the wire and t is the transmission time of the wire in milliseconds.

Output

For each test case the output consists of a single integer equal to the total amount of data about servers that need to be stored in the network.

Example

Sample input:

```
1
4 3
2
3
1
1
1 4 30
2 3 20
3 4 20
```

Sample output:

```
9
```

(because $B(1) = \{1, 2\}$, $B(2) = \{2\}$, $B(3) = \{2, 3\}$, $B(4) = \{1, 2, 3, 4\}$)

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-07-07

Time limit [s]: 15

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Warsaw 2002

SPOJ Problem Set

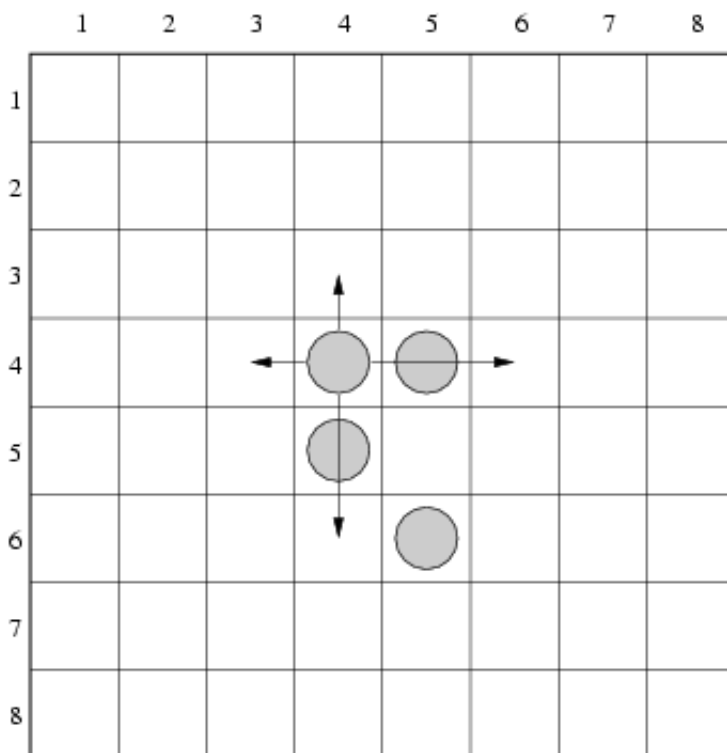
120. Solitaire

Problem code: SOLIT

Solitaire is a game played on an 8x8 chessboard. The rows and columns of the chessboard are numbered from 1 to 8, from the top to the bottom and from left to right respectively.

There are four identical pieces on the board. In one move it is allowed to:

- move a piece to an empty neighboring field (up, down, left or right),
- jump over one neighboring piece to an empty field (up, down, left or right).



There are 4 moves allowed for each piece in the configuration shown above. As an example let's consider a piece placed in the row 4, column 4. It can be moved one row up, two rows down, one column left or two columns right.

Task

Write a program that:

- reads two chessboard configurations from the standard input,
- verifies whether the second one is reachable from the first one in at most 8 moves,
- writes the result to the standard output.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case, each of two input lines contains 8 integers a_1, a_2, \dots, a_8 separated by single spaces and describes one configuration of pieces on the chessboard. Integers a_{2j-1} and a_{2j} ($1 \leq j \leq 4$) describe the position of one piece - the row number and the column number respectively.

Output

For each test case the output should contain one word for each test case - 'YES' if a configuration described in the second input line is reachable from the configuration described in the first input line in at most 8 moves, or one word 'NO' otherwise.

Example

Sample input:

```
1
4 4 4 5 5 4 6 5
2 4 3 3 3 6 4 6
```

Sample output:

```
YES
```

Added by: Adrian Kosowski

Date: 2004-07-07

Time limit [s]: 15

Source limit [B]: 50000

Resource: ACM Central European Programming Contest, Warsaw 2002

SPOJ Problem Set

121. Timetable

Problem code: TTABLE

You are the owner of a railway system between n cities, numbered by integers from 1 to n . Each train travels from the start station to the end station according to a very specific timetable (always on time), not stopping anywhere between. On each station a departure timetable is available. Unfortunately each timetable contains only direct connections. A passenger that wants to travel from city p to city q is not limited to direct connections however - he or she can change trains. Each change takes zero time, but a passenger cannot change from one train to the other if it departs before the first one arrives. People would like to have a timetable of all optimal connections. A connection departing from city p at A o'clock and arriving in city q at B o'clock is called optimal if there is no connection that begins in p not sooner than at A , ends in q not later than at B , and has strictly shorter travel time than the considered connection. We are only interested in connections that can be completed during same day.

Task

Write a program that:

- reads the number n and departure timetable for each of n cities from the standard input,
- creates a timetable of optimal connections from city 1 to city n ,
- writes the answer to the standard output.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first line of the input contains an integer n ($2 \leq n \leq 100000$). The following lines contain n timetables for cities 1, 2, ..., n respectively.

The first line of the timetable description contains only one integer m . Each of the following m lines corresponds to one position in the timetable and contains: departure time A , arrival time B ($A < B$) and destination city number t ($1 \leq t \leq n$) separated by single spaces. Departure time A and arrival time B are written in format $hh : mm$, where hh are two digits representing full hours ($00 \leq hh \leq 23$) and mm are two digits representing minutes ($00 \leq mm \leq 59$). Positions in the timetable are given in non-decreasing order according to the departure times. The number of all positions in all timetables does not exceed 1000000.

Output

For each test case the first line of the output contains an integer r - the number of positions in the timetable being the solution. Each of the following r lines contains a departure time A and an arrival time B separated by single space. The time format should be like in the input and positions in the timetable should be ordered increasingly according to the departure times. If there is more then one optimal connection with the same departure and arrival time, your program should output just one.

Example

Sample input:

```
1
3
3
09:00 15:00 3
10:00 12:00 2
11:00 20:00 3
2
11:30 13:00 3
12:30 14:00 3
0
```

Sample output:

```
2
10:00 14:00
11:00 20:00
```

Warning: enormous Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-07-07

Time limit [s]: 20

Source limit [B]:50000

Resource: ACM Central European Programming Contest, Warsaw 2002

SPOJ Problem Set

122. Voracious Steve

Problem code: STEVE

Steve and Digit bought a box containing a number of donuts. In order to divide them between themselves they play a special game that they created. The players alternately take a certain, positive number of donuts from the box, but no more than some fixed integer. Each player's donuts are gathered on the player's side. The player that empties the box eats his donuts while the other one puts his donuts back into the box and the game continues with the "loser" player starting. The game goes on until all the donuts are eaten. The goal of the game is to eat the most donuts. How many donuts can Steve, who starts the game, count on, assuming the best strategy for both players?

Task

Write a program that:

- reads the parameters of the game from the standard input,
- computes the number of donuts Steve can count on,
- writes the result to the standard output.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first and only line of the input contains exactly two integers n and m separated by a single space, $1 \leq m \leq n \leq 100$ - the parameters of the game, where n is the number of donuts in the box at the beginning of the game and m is the upper limit on the number of donuts to be taken by one player in one move.

Output

For each test case the output contains exactly one integer equal to the number of donuts Steve can count on.

Example

Sample input:

```
1
5 2
```

Sample output:

```
3
```

Added by: Adrian Kosowski
Date: 2004-07-07
Time limit [s]: 15
Source limit [B]:50000
Resource: ACM Central European Programming Contest, Warsaw 2002

SPOJ Problem Set

123. Paying in Byteland

Problem code: PAYING

There are infinitely many coin denominations in the Byteland. They have values of 2^i for $i=0,1,2,\dots$. We will say that set of coins c_1, c_2, \dots, c_k is perfect when it is possible to pay every amount of money between 0 and $c_1 + \dots + c_k$ using some of them (so $\{4, 2, 2, 1\}$ is perfect while $\{8, 1\}$ is not). The question is - is it always possible to change given sum n into a perfect set of coins? Of course it is possible ;). Your task will be more complicated: for a sum n you should find minimal number of coins in its perfect representation.

Input

First line of input contains one integer $c \leq 50$ - number of test cases. Then c lines follow, each of them consisting of exactly one integer $n \leq 10^{1000}$.

Output

For each test case output minimal number of coins.

Example

Input :

```
5
507
29
8574
233
149
```

Output :

```
14
7
21
11
10
```

Added by: Pawel Gawrychowski
Date: 2004-07-07
Time limit [s]: 15
Source limit [B]: 50000

SPOJ Problem Set

126. Solovay-Strassen Inverted

Problem code: SOLSTRAS

Let us denote the set of all prime numbers by the symbol \mathbf{P} . The Solovay-Strassen algorithm determines whether a given positive odd integer $n > 2$ belongs to \mathbf{P} .

The *Legendre function* sig for number $n \in \mathbf{N}$ with parameter $s \in \mathbf{N}$ ($s < n$) is defined by the formula $\text{sig}(n,s) = s^{(n-1)/2} \bmod n$. The symbol \bmod is defined in such a way as to return the result with the smallest possible absolute value, from the range $(-n/2, n/2]$.

The *Jacobi function* jac for number $n \in \mathbf{N}$ with parameter $s \in \mathbf{N}$ ($s < n$) is given as:

$$\text{jac}(n,s) = \begin{cases} \text{sig}(n,s), & \text{if } n \in \mathbf{P} \\ \prod_{i=1}^k \text{sig}(p_i,s), & \text{if } n = \prod_{i=1}^k p_i, \text{ where all } p_i \in \mathbf{P} \end{cases}$$

It is interesting to note that for given n and s , the values of $\text{sig}(n,s)$ and $\text{jac}(n,s)$ can be computed in $O((\log_2 n)^2)$ time. For particulars consult an encyclopedia, such as MathWorld.

The deterministic version of the Solovay-Strassen primality-test algorithm is given below.

```
algorithm Solovay-Strassen ( $n$ )  
var  $s$ ;  
begin  
  for  $s$  in  $\{1,2,3,4,\dots,n\}$  do  
    if  $\text{sig}(n,s) \neq \text{jac}(n,s)$   
      then return " $n$  is composite (detected at attempt  $\langle s \rangle$ )";  
  return " $n$  is prime";  
end.
```

Task

We are not asking you to implement the Solovay-Strassen algorithm, this would be far too easy :). Instead, try to find values of n , for which the output of the algorithm would be " n is composite (detected at attempt 1)", " n is composite (detected at attempt 2)", and so on. Write out as many of these values as you can in consecutive lines, and try to keep them as small as possible.

Scoring

The score awarded to your program is the total of all points given for its individual lines.

The i -th line output by your program should contain exactly one positive odd integer $n > 2$ of no more than 500 decimal digits. If $\text{Solovay-Strassen}(n)$ yields the answer " n is composite (detected at attempt i)", you will receive $i/\log_{10} n$ points for this line, if not - your program will be considered incorrect. Output 0 if you don't want a line to be assessed. Only the first 1000 lines of output are taken into

account.

Example

A program outputting:

```
0
0
561
```

will receive $3/\log_{10} 561 = 1.091$ points.

Added by: Konrad Piwakowski

Date: 2004-07-10

Time limit [s]: 5

Source limit [B]:50000

SPOJ Problem Set

127. Johnny Goes Shopping

Problem code: JOHNNY

Johnny visited his favourite supermarket to purchase as many sweets as he could afford. Since daddy had left his credit card at home untended, this was not really a problem. Once he had (barely) managed to push the trolley laden with chocolate bars past the cash desk, he began to wonder how to carry all the shopping home without breaking his back.

You must know that Johnny is a perfectly normal child, and has exactly 2 hands. Help him distribute his load between both hands so as to minimise the difference in load between both hands.

Input

The first line of input contains a single integer $n \leq 10000$ denoting the number of sweet packets Johnny has bought. In each of the next n lines a single positive integer is given, describing the weight of the respective packet (the weight is a scalar value never exceeding 10^{14}).

Output

In separate lines, output the numbers of the packets which Johnny should carry in his left hand. Assume packets are numbered in the input order from 1 to n .

Scoring

Your program shall receive $\log_{10} (s/(|d|+1))$ points, where s is the total weight of all parcels, while d denotes the difference in weight between the load carried in Johnny's left and right hand.

Example

For the sample input:

```
3
5
8
4
```

a program outputting:

```
2
3
```

will score $\log_{10} ((5+8+4)/(|8+4-5|+1)) = 0.327$ points.

Added by: Adrian Kosowski
Date: 2004-07-10
Time limit [s]: 10
Source limit [B]:50000

SPOJ Problem Set

130. Rent your airplane and make money

Problem code: RENT

"ABEAS Corp." is a very small company that owns a single airplane. The customers of ABEAS Corp are large airline companies which rent the airplane to accommodate occasional overcapacity.

Customers send renting orders that consist of a time interval and a price that the customer is ready to pay for renting the airplane during the given time period.

Orders of all the customers are known in advance. Of course, not all orders can be accommodated and some orders have to be declined. Eugene LAWLER, the Chief Scientific Officer of ABEAS Corp would like to maximize the profit of the company.

You are requested to compute an optimal solution.

Small Example

Consider for instance the case where the company has 4 orders:

- Order 1 (start time 0, duration 5, price 10)
- Order 2 (start time 3, duration 7, price 8)
- Order 3 (start time 5, duration 9, price 7)
- Order 4 (start time 6, duration 9, price 8)

The optimal solution consists in declining Order 2 and 3 and the gain is $10+8 = 18$.

Note that the solution made of Order 1 and 3 is feasible (the airplane is rented with no interruption from time 0 to time 14) but non-optimal.

Input

The first line of the input contains a number $T \leq 30$ that indicates the number of test cases to follow. The first line of each test case contains the number of orders n ($n \leq 10000$). In the following n lines the orders are given. Each order is described by 3 integer values: The start time of the order st ($0 \leq st < 1000000$), the duration d of the order ($0 < d < 1000000$), and the price p ($0 < p < 100000$) the customer is ready to pay for this order.

Output

You are required to compute an optimal solution. For each test case your program has to write the total price paid by the airlines.

Example

Input:

```
1
4
0 5 10
3 7 14
5 9 7
6 9 8
```

Output:

```
18
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kuegel

Date: 2004-07-13

Time limit [s]: 5

Source limit [B]:50000

Resource: ACM Southwestern European Regional Contest, Paris 2003

SPOJ Problem Set

131. Square dance

Problem code: SQDANCE

You are hired by french NSA to break the RSA code used on the Pink Card. The easiest way to do that is to factor the public modulus and you have found the fastest algorithm to do that, except that you have to solve a subproblem that can be modeled in the following way.

Let $\mathcal{P} = \{p_1, p_2, \dots, p_n\}$ be a set of prime numbers. If $\mathcal{S} = \{s_1, s_2, \dots, s_u\}$ and

$\mathcal{T} = \{t_1, \dots, t_v\}$ are formed with elements of \mathcal{P} , then $\mathcal{S} * \mathcal{T}$ will denote the quantity

$$s_1 * s_2 * \dots * s_u * t_1 * t_2 * \dots * t_v.$$

We call relation a set of two primes p, q , where p and q are distinct elements of \mathcal{P} . You dispose of a collection of R relations $\mathcal{S}_i = \{p_i, q_i\}$ and you are interested in finding sequences of these,

$\mathcal{S}_{i_1}, \mathcal{S}_{i_2}, \dots, \mathcal{S}_{i_k}$ such that

$$\mathcal{S}_{i_1} * \mathcal{S}_{i_2} * \dots * \mathcal{S}_{i_k}$$

is a perfect square.

The way you look for these squares is the following. The ultimate goal is to count squares that appear in the process. Relations arrive one at a time. You maintain a collection \mathcal{C} of relations that do not contain any square subproduct. This is easy: at first, \mathcal{C} is empty. Then a relation arrives and \mathcal{C} begins to grow. Suppose a new relation $\{p, q\}$ arrives. If no square appears when adding $\{p, q\}$ to \mathcal{C} , then

$\{p, q\}$ is added to the collection. Otherwise, a square is about to appear, we increase the number of

squares, but we do not store this relation, hence \mathcal{C} keeps the desired property.

Let us consider an example. First arrives $\mathcal{S}_1 = \{2, 3\}$ and we put it in \mathcal{C} ; then arrives

$\mathcal{S}_2 = \{5, 11\}, \mathcal{S}_3 = \{3, 7\}$ and they are stored in \mathcal{C} . Now enters the relation $\mathcal{S}_4 = \{2, 7\}$. This

relation could be used to form the square:

$$\mathcal{S}_1 * \mathcal{S}_3 * \mathcal{S}_4 = (2 * 3) * (3 * 7) * (2 * 7) = (2 * 3 * 7)^2.$$

So we count 1 and do not store \mathcal{S}_4 in \mathcal{C} . Now we consider $\mathcal{S}_5 = \{5, 11\}$ that could make a square

with S_2 , so we count 1 square more. Then $S_6 = \{2, 13\}$ is put into C . Now $S_7 = \{7, 13\}$ could make the square $S_1 * S_3 * S_6 * S_7$. Eventually, we get 3 squares.

Input

The first line of the input contains a number $T \leq 30$ that indicates the number of test cases to follow. Each test case begins with a line containing two integers P and R : $P \leq 10^5$ is the number of primes occurring in the test case; $R (\leq 10^5)$ is the number of sets of primes that arrive. The subsequent R lines each contain two integers i and j making a set $\{p_i, q_i\} (1 \leq i, j \leq P)$. Note that we actually do not deal with the primes, they are irrelevant to the solution.

Output

For each test case, output the number of squares that can be formed using the preceding rules.

Example

Input :

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

Output :

```
3
2
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kuegel

Date: 2004-07-13

Time limit [s]: 10

Source limit [B]: 50000

Resource: ACM Southwestern European Regional Contest, Paris 2003

SPOJ Problem Set

132. Help R2-D2!

Problem code: HELPR2D2

In Episode III of Star Wars (whose alleged title is "How I became Vader"), R2-D2 (Artoo-Detoo) is again confronted to a tedious work. He is responsible for the loading of the republic transport starships in the fastest way. Imagine a huge space area where n starships are parked. Each starship has a capacity of K cubic femtoparsec. Containers C_i arrive one at a time with some volume v_i (expressed in cubic femtoparsec). R2-D2 wants to minimize the number of starships used for a given sequence of containers.

Smart as he is, R2-D2 knows for sure that the problem is a hard one, even with the force being around. Here is the heuristics he selected to solve his problem. Start with all starships ready to load, and numbered $S_0, S_1, \text{etc.}$ When a container C_j arrives, select the starship of minimal index i that can contain C_j and put it in S_i . In some sense, this heuristic minimizes the move of the container arriving before its loading.

At the end of the n arrivals, R2-D2 counts the number s of starships used and he measures the total waste w of the sequence. For $i=0..s-1$, the waste in starship i is given by the unused volume.

Your task is to simulate the algorithm of R2-D2.

Input

The first line of the input contains a number $T \leq 10$ that indicates the number of test cases to follow. Each test case begins with K on a line ($K \leq 1000$), followed by the number of containers in the sequence, n , on the second line ($1 \leq n \leq 1000000$). There are two possible formats for the remaining lines. If it contains one integer, then this is the next v_i . If it begins with the character b (for block), it is followed by 2 integers r and v . This means that the r next containers arriving have volume v .

Output

Your program must output the number s of starships used, followed by a blank, followed by the total waste w .

You can assume, that at most 100000 starships are needed, and R2-D2 has to change the starships in which the next container is loaded at most 100000 times.

Example

Input:

```
2
100
3
50
25
70
100
4
50
b 2 40
20
```

Output:

2 55
2 50

Added by: Adrian Kuegel

Date: 2004-07-14

Time limit [s]: 15

Source limit [B]: 50000

Resource: ACM Southwestern European Regional Contest, Paris 2003

SPOJ Problem Set

134. Phony Primes

Problem code: PHONY

You are chief debugger for Poorly Guarded Privacy, Inc. One of the top selling product, ReallySecureAgent©, seems to have a problem with its prime number generator. It produces from time to time bogus primes N .

After a while, you realize that the problem is due to the way primes are recognized.

Every phony prime N you discover can be characterized as follows. It is odd and has distinct prime factors, say $N = p_1 * p_2 * \dots * p_k$ with $p_i \neq p_j$, where the number k of factors is at least 3.

Moreover, for all $i=1..k$, $p_i - 1$ divides $N-1$. For instance, $561 = 3*11*17$ is a phony prime.

Intrigued by this phenomenon, you decide to write a program that enumerates all such N 's in a given interval $[N_{\min}, N_{\max}]$ with $1 \leq N_{\min} < N_{\max} < 2^{31}$, $N_{\max} - N_{\min} < 10^6$.

Please note, that the source code limit for this problem is 2000 Bytes to avoid precalculated tables.

Input

Each test case contains one line. On this line are written two integers N_{\min} and N_{\max} separated by a blank. The end of the input is signalled by a line containing two zeros. The number of test cases is approximately 2000.

Output

For each test case, output the list of phony primes in increasing order, one per line. If there are no phony primes in the interval, then simply output none on a line.

Example

Input:

```
10 2000
20000 21000
0 0
```

Output:

```
561
1105
1729
none
```

Added by: Adrian Kuegel
Date: 2004-07-15
Time limit [s]: 30
Source limit [B]:2000
Resource: ACM Southwestern European Regional Contest, Paris 2003

SPOJ Problem Set

135. Men at work

Problem code: MAWORK

Every morning you have to drive to your workplace. Unfortunately, roads are under constant repair. Fortunately, administration is aware that this may cause trouble and they enforce a strict rule on roadblocks: roads must be blocked only half of the time. However, contractors are free to schedule their working hours, still they must follow regulations:

- Working periods (when the road is blocked) and rest periods (when the road is open) must alternate and be of fixed length.
- The beginning of the day (time zero) must coincide with the beginning of a period.

Write a program that, given a description of the road network and of contractors schedules outputs the minimal time needed to drive from home to work.

Input

The first line of the input contains a number $T \leq 10$ that indicates the number of test cases to follow. The road network is represented on a $N \times N$ grid and the first line of each test case consists in the number N , $2 \leq N \leq 25$.

Then follows N lines of N characters that represent the road network at time zero. Those lines are made of "." (standing for open road) and "*" (standing for roadblock) and they encode the rows of the grid in increasing order, while columns are also presented in increasing order. Conventionally, your home is at the position first row, first column, while your workplace is at the position last row, last column. Furthermore, you leave home at time $t=0$, that is, your starting position is first row, first column at time zero.

At a given time t , your car must be on some "open road" cell. It takes one time unit to drive to any of the four adjacent cells heading toward north, south, west or east, and you may also choose to stay on the same cell for one time unit. Of course, those five moves are valid if and only if the target cell exists and is free at time $t+1$.

Finally comes N lines of N characters that represent the contractors schedules. Those lines match the ones of the grid description and are made of N characters 0,1,...,9 that specify the duration of the working (and rest) period for a given cell. Observe that 0 is a bit special, since it means that the corresponding cell status does not change.

Output

The output consist in a single line for each test case, holding either the requested time, or NO, if driving from home to work is not possible.

Example

Input:

```

2
10
.*****
. ....**
* .....*
* .....*
* .....*
* .....*
* .....*
* .....*
* .....*
* .....*
* .....*
*****.
0000000000
0000000000
0000000000
0000000000
0000000000
0123456780
0000000000
0000000000
0123456780
0000000000
3
...
**
**
021
002
000

```

Output:

34
NO

Added by: Adrian Kuegel
Date: 2004-07-16
Time limit [s]: 20
Source limit [B]: 50000
Resource: ACM Southwestern European Regional Contest, Paris 2003

SPOJ Problem Set

136. Transformation

Problem code: TRANS

You are given two short sequences of numbers, X and Y. Try to determine the minimum number of steps of transformation required to convert sequence X into sequence Y, or determine that such a conversion is impossible.

In every step of transformation of a sequence, you are allowed to replace exactly one occurrence of one of its elements by a sequence of 2 or 3 numbers inserted in its place, according to a rule specified in the input file.

Input

The input begins with the integer t, the number of test cases. Then t test cases follow.

For each test case, the first line of input contains four integers - N, M, U, V ($1 \leq N, M \leq 50$). The next two lines of input contain sequences X and Y, consisting of N and M integers respectively. The next U lines contain three integers: $a \ b \ c$ each, signifying that integer a can be converted to the sequence $b \ c$ in one step of transformation. The next V-U lines contain four integers: $a \ b \ c \ d$ each, signifying that integer a can be converted to the sequence $b \ c \ d$ in one step of transformation. With the exception of N and M, all integers provided at input are positive and do not exceed 30.

The format of one set of input data is illustrated below.

```
N M U V
x1 x2 ... xN
y1 y2 ... yM
a1 b1 c1
⋮
aU bU cU
aU+1 bU+1 cU+1 dU+1
⋮
aV bV cV dV
```

Output

For each test case output -1 if it is impossible to convert sequence X into sequence Y, or the minimum number of steps required to achieve this conversion otherwise.

Example

Sample input:

```
1
3 10 2 3
2 3 1
2 1 1 2 2 1 2 1 2 1
```


3 1 2
3 3 3
3 1 3 2

Sample output:

6

Added by: Adrian Kosowski

Date: 2004-07-18

Time limit [s]: 15

Source limit
[B]: 50000

Resource: based on a problem from the VI Polish Collegiate Team Programming Contest
(AMPPZ), 2001

SPOJ Problem Set

137. Partition

Problem code: PARTIT

A *partition* of positive integer m into n components is any sequence a_1, \dots, a_n of positive integers such that $a_1 + \dots + a_n = m$ and $a_1 \leq a_2 \leq \dots \leq a_n$. Your task is to determine the partition, which occupies the k -th position in the lexicographic order of all partitions of m into n components.

The lexicographic order is defined as follows: sequence a_1, \dots, a_n comes before b_1, \dots, b_n iff there exists such an integer $i, 1 \leq i \leq n$, that $a_j = b_j$ for all $j, 1 \leq j < i$, and $a_i < b_i$.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the input consists of three lines, containing the positive integers m , n and k respectively ($1 \leq n \leq 10$, $1 \leq m \leq 220$, k is not larger than the number of partitions of m into n components).

Output

For each test case output the ordered elements of the sought partition, separated by spaces.

Example

Sample input:

```
1
9
4
3
```

Sample output:

```
1 1 3 4
```

Added by: Adrian Kosowski

Date: 2004-07-19

Time limit [s]: 15

Source limit [B]: 50000

Resource: VI Polish Collegiate Team Programming Contest (AMPPZ), 2001

SPOJ Problem Set

138. Election Posters

Problem code: POSTERS

A parliamentary election was being held in Byteland. Its enterprising and orderly citizens decided to limit the entire election campaign to a single dedicated wall, so as not to ruin the panorama with countless posters and billboards. Every politician was allowed to hang exactly one poster on the wall. All posters extend from top to bottom, but are hung at different points of the wall, and may be of different width. The wall is divided horizontally into sections, and a poster completely occupies two or more adjacent sections.

With time, some of the posters were covered (partially or completely) by those of other politicians. Knowing the location of all the posters and the order in which they were hung, determine how many posters have at least one visible section in the end.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

Each test case begins with a line containing integer n - the number of posters ($1 \leq n \leq 40000$). Then n lines follow, the i -th ($1 \leq i \leq n$) containing exactly two integers l_i r_i , denoting the numbers of the leftmost and rightmost sections covered by the i -th poster ($1 \leq l_i < r_i \leq 10^7$). The input order corresponds to the order of hanging posters.

Output

For each test case output a line containing one integer - the number of posters with visible sections.

Example

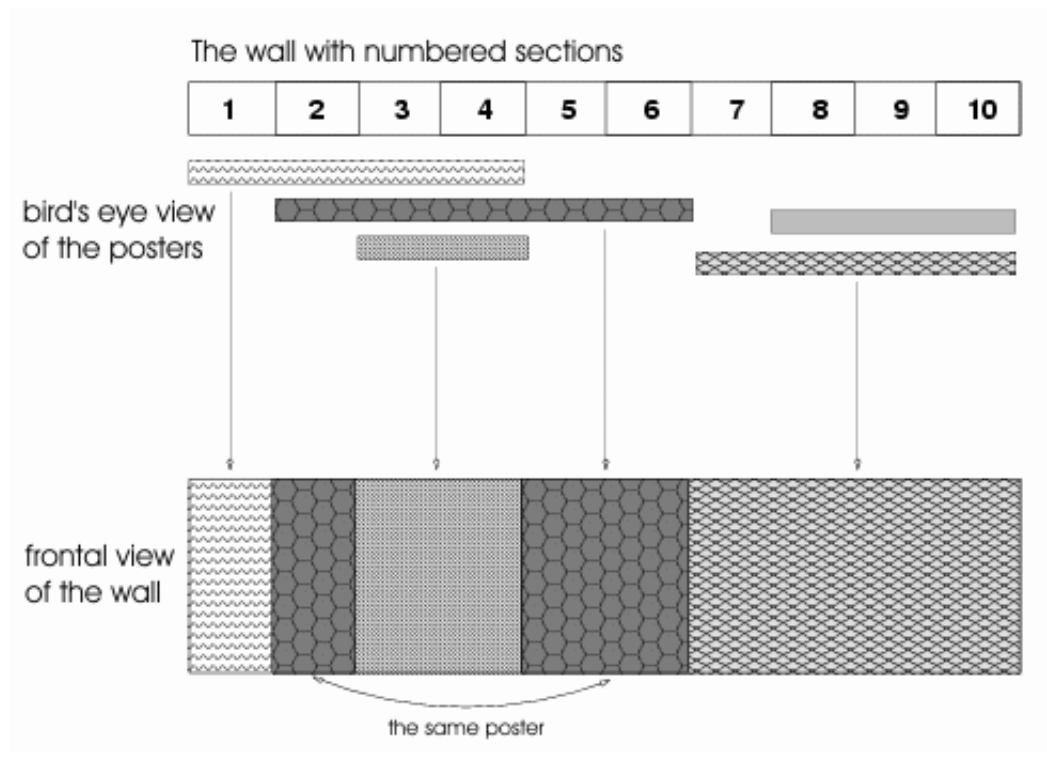
Sample input:

```
1
5
1 4
2 6
8 10
3 4
7 10
```

Sample output:

```
4
```

An illustration of the sample input is given below.



Added by: Adrian Kosowski

Date: 2004-07-19

Time limit [s]: 15

Source limit [B]:50000

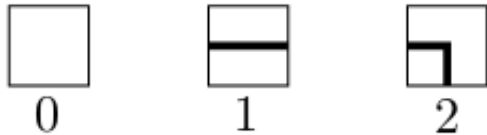
Resource: VI Polish Collegiate Team Programming Contest (AMPPZ), 2001

SPOJ Problem Set

139. The Long and Narrow Maze

Problem code: MAZE

Consider a maze consisting of 3 rows of n square blocks each. The passageways in every block match one of three possible patterns, numbered 0 (empty), 1 (straight) and 2 (bent), as depicted below.



Your task is to determine whether it is possible to create a passage in a given maze, with an entrance at the left end and an outlet at the right end of the maze, only by rotating some of the squares of the maze by a multiple of 90 degrees.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

Each test case begins with a line containing a single integer n - the number of squares in one row of the maze ($1 \leq n \leq 200000$). The next n lines contain three integers each, denoting the types of blocks in consecutive columns of the maze. A column description is of the form $a\ b\ c$ ($0 \leq a, b, c \leq 2$), where a represents the type of the block in the first row, b - in the second row and c - in the third row.

Output

For each test case output the word `yes` if it is possible to rotate the squares so as to form a connection between the left and right edge, and the word `no` in the opposite case.

Example

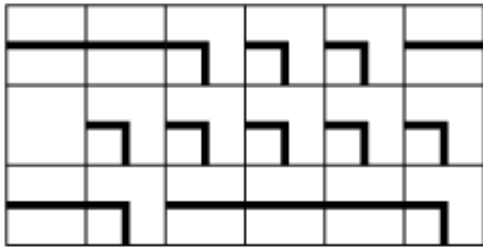
Sample input:

```
1
6
1 0 1
1 2 2
2 2 1
2 2 1
2 2 1
1 2 2
```

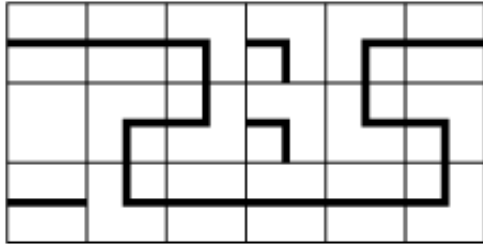
Sample output:

```
yes
```

Indeed, the sample input corresponds to the following maze:



for which there exists a correct solution to the problem:



Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-07-19

Time limit [s]: 23

Source limit [B]: 50000

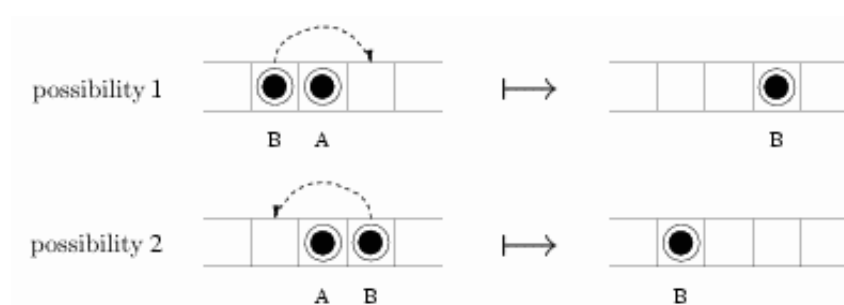
Resource: VI Polish Collegiate Team Programming Contest (AMPPZ), 2001

SPOJ Problem Set

140. The Loner

Problem code: LONER

The *loner* is a one-dimensional board game for a single player. The board is composed of squares arranged in a single line, some of which initially have pawns on them. The player makes a move by jumping with a pawn over a pawn on an adjacent field, to an empty square two fields to the right or left of its initial position. The pawn that was jumped over is removed directly after the move, as illustrated below.



The game is considered won if exactly one pawn remains on the gaming board, and is lost if the player cannot make a move.

Given the initial state of the gaming board, your task is to determine whether it is possible for the player to win the game.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

Each test case begins with the positive integer $n \leq 32000$, denoting the size of the gaming board. The second and last line of the test case description contains a sequence of n characters 0 or 1, without any white spaces. The i -th square of the board is occupied by a pawn at the start of the game iff the i -th character of this sequence is 1.

Output

For each test case output the word `yes` if it is possible for the player to win the game for the presented starting configuration, or the word `no` in the opposite case.

Example

Sample input:

```
2
7
0110011
6
111001
```

Sample output:

yes
no

Added by: Adrian Kosowski

Date: 2004-07-21

Time limit [s]: 15

Source limit [B]: 50000

Resource: VI Polish Collegiate Team Programming Contest (AMPPZ), 2001

SPOJ Problem Set

142. Johnny and the Glue

Problem code: GLUE

Little Johnny decided he needed to stick an open metal box to the floor in the hall of his parents' house, so that all guests coming in would trip on it. He knew that as soon as his parents saw what he had done, they would try to remove it, and he wasn't going to stand for this. So, he chose the strongest glue in his possession and left lots of dabs of it on the floor (from our point of view, these can be regarded as points). Now, the only question that remained was how to stick the box onto the floor. Johnny is very particular about the way he does this: the box is always stuck face down, so that it only touches the floor on the four edges of the rectangle that forms its base. He would like each of these edges to make contact with at least two dabs of glue. Furthermore, he doesn't want any of the dabs to stay outside the box, since this would ruin the fun (there is no way you can trip someone up, if you've glued them to the floor, is there?).

Obviously, Johnny can sometimes reach his objective in more than one way (especially since he has prepared boxes of all possible dimensions for his act of mischief). Depending on how he does this, a different section of floor will be covered by the box. Determine in how many ways Johnny can choose the section of floor to be covered by the box when gluing.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

The first line of each test case contains positive integer $n \leq 10000$ - the number of dabs of glue on the floor. The next n lines contain two integers, x y ($-15000 \leq x, y \leq 15000$), representing the x and y coordinates of the dabs (given in the order in which they were placed by Johnny ;).

Output

For each test case output the number of different sections of floor Johnny may choose to cover (possibly 0).

Example

Sample input:

```
1
8
1 0
1 4
0 3
5 4
5 0
6 1
6 3
0 1
```

Sample output:

```
2
```

Added by: Adrian Kosowski
Date: 2004-07-22
Time limit [s]: 15
Source limit
[B]: 50000
Resource: based on a problem from the VI Polish Collegiate Team Programming Contest
(AMPPZ), 2001

SPOJ Problem Set

145. Aliens

Problem code: ALIENS

Aliens visited our planet with an obvious intention to find some new species for their space zoo. After entering Earth's orbit, they positioned themselves over the town of Belgrade, having detected some life-form activity on the ground. As they approached the surface, they saw a group of half-intelligent beings. Those creatures were actually competitors of the Balkan Olympiad in Informatics who were enjoying the excursion after intense contest. Aliens want to abduct all n ($2 \leq n \leq 100000$) competitors since they are very compassionate, and don't want their creatures to feel lonely in the space zoo.

Aliens use tractor beam to take their prey. Tractor beam works in the following way: it projects a circle-shaped beam from the spacecraft to the ground vertically beneath it, and all beings that are found in that circle or on its boundary are taken. Projecting the tractor beam needs a certain amount of energy to be spent. As the radius of the tractor beam (radius of the circle on the ground) increases, more and more energy is required. Although extremely intelligent, aliens are much more advanced in social sciences than in programming. That's why they are asking you to help them find the position of their spacecraft so that the energy required to take all of the n competitors is minimal.

Help our alien brothers! Write a program that will find the required minimal radius of tractor beam that contains all n competitors and the optimal spacecraft location - which is the same as the center of the circle on the ground.

Input

First line of input contains one integer $c \leq 20$ - number of test cases. Each test case begins with number n ($2 \leq n \leq 100000$). Then n lines follow and i -th of them contains two real numbers x_i and y_i ($-10000.0 \leq x_i, y_i \leq 10000.0$) representing coordinates of the i -th competitor.

Output

For each test case output radius of the tractor beam and coordinates of the spacecraft. Numbers should be rounded to two decimal places.

Example

Input :

```
1
6
8.0 9.0
4.0 7.5
1.0 2.0
5.1 8.7
9.0 2.0
4.5 1.0
```

Output :

```
5.00
5.00 5.00
```

Warning: large Input/Output data, be careful with certain languages

Added by: Pawel Gawrychowski
Date: 2004-07-21
Time limit [s]: 10
Source limit [B]:50000
Resource: Balkan Olympiad in Informatics 2002

SPOJ Problem Set

146. Fast Multiplication Again

Problem code: MULTIPLY

After trying to solve Problem Number 31 (Fast Multiplication) with some script languages that support arbitrary large integers and timing out, you wonder what would be the best language to do fast multiplication of integers. And naturally it comes to your mind: Of course it is brainf**k, because there are only very cheap operations in that language.

Input

Two positive integers, ended with a line feed (ASCII 10) each.

Output

The product of the two integers, terminated by a line feed. You may assume that this number will be less than 10000.

Example

Input :

1
2

Output :

2

Added by: Robin Nittka

Date: 2004-07-21

Time limit [s]: 3

Source limit [B]:5000

SPOJ Problem Set

147. Tautology

Problem code: TAUT

Write a program that checks if the given logical expression is a tautology. The logical expression is a tautology if it is always true, regardless of logical value of its variables.

Input

On the first line there is the number of expressions to check (at most 35). The expression is in a prefix notation, that means that operator precedes its arguments. The following logical operators will be used:

C - and
D - or
I - implies
E - if, and only if
N - not

The variables will be lowercase letters (a-z). There will be no more than 16 different letters in the expression. The length of the expression will not exceed 111 characters.

Output

For each expression write one word: YES if it is a tautology, NO in other case.

Example

Sample input:

```
7
IIpqDpNp
NCNpp
Iaz
NNNNNNNp
IIqrIIpqIpr
Ipp
Ezz
```

Sample output:

```
YES
YES
NO
NO
YES
YES
YES
```

Added by: Piotr Lowiec
Date: 2004-07-25
Time limit [s]: 15
Source limit [B]:50000

SPOJ Problem Set

148. Land for Motorways

Problem code: MLAND

With every year, the plans for the construction of motorways in Poland are more and more advanced. For some time, it seemed as if the building was actually going to start, so the question of purchasing the land under the roads was of some importance. Only certain cities can be connected by a road directly, provided the farmer owning the land under it agrees to sell out. As a result of the constant swing of moods, the price demanded for the land by each farmer changes in a linear fashion, with possibly different coefficients for every road. It may either increase or decrease (and sometimes even be negative, if the owner anticipates future profit from the proximity of a motorway).

It has been decided that the purchase of land will be made at some moment in between two fixed dates. At that moment, the current prices of land will be frozen, and the least costly configuration of bidirectional roads connecting all cities (directly or indirectly) will be chosen. All the land under the selected roads will subsequently be bought at the frozen price.

You act as an intermediary for the purchase and charge a steady commission, proportional to the value of the transaction. Since it is one of your tasks to select the moment of purchase, do so in such a way as to maximise your profit.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first line contains two integers n m , denoting the number of cities to be connected and the number of available potential roads, respectively ($1 \leq n \leq 120, 1 \leq m \leq 820$). The next line contains two integers t_1 t_2 , which stand for the earliest possible and latest possible moments of purchase ($-10000 \leq t_1 \leq t_2 \leq 10000$). Each of the following m lines contains four integers, the i -th being: u_i v_i a_i b_i , which means that the i -th road connects city u_i with city v_i , and the purchase of the land under it costs $b_i + j \cdot a_i$ units of currency at moment j (e.g. at moment 0 the land costs b_i units). Please note that these integers are chosen from the following ranges: $0 \leq u_i, v_i \leq n-1$, $-32000 \leq a_i, b_i \leq 32000$.

Output

For each test case output a line with two floating point numbers, accurate to three digits after the decimal point. The first represents the moment of transaction you ought to choose, the second - the total value of the transaction at that moment. If more than one moment fulfills the conditions of the problem, choose the earliest.

Example

Sample input:

```
2
5 6
0 5
1 0 -6 -4
2 0 3 -3
3 0 1 5
3 1 -2 -3
4 1 -3 -2
4 3 -2 -3
5 7
-20 20
1 0 1 2
2 1 -7 4
3 1 -9 0
3 2 4 9
4 1 0 -2
4 2 2 3
4 3 6 -5
```

Sample output:

```
0.000 -13.000
0.111 -1.000
```

Added by: Adrian Kosowski

Date: 2004-07-24

Time limit [s]: 15

Source limit [B]:50000

Resource: VII Polish Collegiate Team Programming Contest (AMPPZ), 2002

SPOJ Problem Set

149. Fencing in the Sheep

Problem code: FSHEEP

A shepherd is having some trouble penning in his flock of sheep. After several hours of ineffectual efforts he gives up, with some of the sheep within their polygon-shaped pen and some outside. Exhausted, he moves to a place within the pen from which he can see the whole interior of the pen (without any fence getting in the way) and begins to count the sheep which are within it. Please assist him in his task.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

The first line of each test case contains two integers n m , denoting the number of vertices of the polygon forming the fence, and the number of sheep in the whole herd ($3 \leq n \leq 100000$, $0 \leq m \leq 100000$). The next n lines contain two integers each, the i -th being x_i y_i - the x and y coordinates of the i -th vertex of the fence (given in anti-clockwise order, $-32000 \leq x_i, y_i \leq 32000$). The next m lines contain two integers each, the j -th being x_j y_j - the x and y coordinates of the j -th sheep (arranged in decreasing order of seniority, $-32000 \leq x_j, y_j \leq 32000$). The shepherd's observation point is within the pen and has coordinates $(0,0)$.

Output

For each test case output a line with a single integer - the number of sheep within the pen. The sheep which are sitting back on the fence and enjoying a cigarette should be included in the count.

Example

Sample input:

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

Sample output:

```
3
```

Date: 2004-07-24

Source limit 50000

Resource: based on a problem from the VII Polish Collegiate Team Programming Contest (AMPPZ), 2002

SPOJ Problem Set

150. Where to Drink the Plonk?

Problem code: PLONK

Consider a city bounded by a square, whose n horizontal and n vertical streets divide it into $(n+1)^2$ square blocks. However, in tribute to the ancient traditions of the first dwellers (who tended to overindulge in alcohol), all the inhabitants live at crossroads. A group of friends would like to meet for an evening of merriment at the place of residence of one of them. With a good deal of foresight, anticipating the difficulties they might have getting back to their respective homes, they would like to meet in the house of the friend which minimises the total walking distance for all of them. Assume that everybody walks along the streets, turning only at crossroads, and the distance between any pair of adjacent crossroads is 1.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first line of input contains the integer n - the number of friends who want to meet ($1 \leq n \leq 10000$). The next n lines contain two integers each, the i -th being x_i y_i , standing for the x and y coordinates of the crossroads at which the i -th friend lives ($0 \leq x_i, y_i \leq 100000$).

Output

For each test case output the total distance covered by all friends when walking to the meeting place.

Example

Sample input:

```
1
7
1 3
3 2
3 5
6 9
10 1
12 4
5 7
```

Sample output:

```
39
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski
Date: 2004-07-28
Time limit [s]: 15
Source limit
[B]: 50000
Resource: based on a problem from the VII Polish Collegiate Team Programming Contest
(AMPPZ), 2002

SPOJ Problem Set

151. The Courier

Problem code: COURIER

Byteland is a scarcely populated country, and residents of different cities seldom communicate with each other. There is no regular postal service and throughout most of the year a one-man courier establishment suffices to transport all freight. However, on Christmas Day there is somewhat more work for the courier than usual, and since he can only transport one parcel at a time on his bicycle, he finds himself riding back and forth among the cities of Byteland.

The courier needs to schedule a route which would allow him to leave his home city, perform the individual orders in arbitrary order (i.e. travel to the city of the sender and transport the parcel to the city of the recipient, carrying no more than one parcel at a time), and finally return home. All roads are bi-directional, but not all cities are connected by roads directly; some pairs of cities may be connected by more than one road. Knowing the lengths of all the roads and the errands to be performed, determine the length of the shortest possible cycling route for the courier.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

Each test case begins with a line containing three integers: n m b , denoting the number of cities in Byteland, the number of roads, and the number of the courier's home city, respectively ($1 \leq n \leq 100, 1 \leq m \leq 10000$). The next m lines contain three integers each, the i -th being u_i v_i d_i , which means that cities u_i and v_i are connected by a road of length d_i ($1 \leq u_i, v_i \leq 100, 1 \leq d_i \leq 10000$). The following line contains integer z - the number of orders to be transport requests the courier has received ($1 \leq z \leq 5$). After that, z lines with the description of the orders follow. Each consists of three integers, the j -th being u_j v_j b_j , which signifies that b_j parcels should be transported (individually) from city u_j to city v_j . The sum of all b_j does not exceed 12.

Output

For each test case output a line with a single integer - the length of the shortest possible bicycle route for the courier.

Example

Sample input:

```
1
5 7 2
1 2 7
1 3 5
1 5 2
2 4 10
2 5 1
3 4 3
3 5 4
3
1 4 2
```

5 3 1
5 1 1

Sample output:
43

Added by: Adrian Kosowski
Date: 2004-07-28
Time limit [s]: 15
Source limit
[B]: 50000
Resource: based on a problem from the VII Polish Collegiate Team Programming Contest
(AMPPZ), 2002

SPOJ Problem Set

153. Balancing the Stone

Problem code: SCALES

You are given scales for weighing loads. On the left side lies a single stone of known weight $W < 2^N$. You own a set of N different weights, weighing $1, 2, 4, \dots, 2^{N-1}$ units of mass respectively. Determine how many possible ways there are of placing some weights on the sides of the scales, so as to balance them (put them in a state of equilibrium). Output this value modulo a small integer D .

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case, the first line contains three integers: N L D , where N denotes the number of weights at your disposal, L is the length of the binary representation of number W , and D is the modulus ($1 \leq L \leq N \leq 1000000$, $2 \leq D \leq 100$). The second line contains the value of W , encoded in the binary system as a sequence of exactly L characters 0 or 1 without separating spaces.

Output

For each test case, output a single line containing one integer - the calculated number of possible weight placements, modulo D .

Example

Sample input:

```
2
6 4 6
1000
6 6 100
100110
```

Sample output:

```
3
5
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-07-31

Time limit [s]: 15

Source limit
[B]: 50000

Resource: based on a problem from the VII Polish Collegiate Team Programming Contest (AMPPZ), 2002

SPOJ Problem Set

154. Sweet and Sour Rock

Problem code: ROCK

A manufacturer of sweets has started production of a new type of sweet called *rock*. Rock comes in sticks composed of one-centimetre-long segments, some of which are sweet, and the rest are sour. Before sale, the rock is broken up into smaller pieces by splitting it at the connections of some segments.

Today's children are very particular about what they eat, and they will only buy a piece of rock if it contains more sweet segments than sour ones. Try to determine the total length of rock which can be sold after breaking up the rock in the best possible way.

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case, the first line of input contains one integer N - the length of the stick in centimetres ($1 \leq N \leq 200$). The next line is a sequence of N characters '0' or '1', describing the segments of the stick from the left end to the right end ('0' denotes a sour segment, '1' - a sweet one).

Output

For each test case output a line with a single integer: the total length of rock that can be sold after breaking up the rock in the best possible way.

Example

Sample input:

```
2
15
100110001010001
16
0010111101100000
```

Sample output:

```
9
13
```

Added by: Adrian Kosowski

Date: 2004-08-03

Time limit [s]: 15

Source limit
[B]: 50000

Resource: based on a problem from the VII Polish Collegiate Team Programming Contest (AMPPZ), 2002

SPOJ Problem Set

155. Solving the Puzzle

Problem code: SOLVING

The 15 puzzle is a classic puzzle made famous in the 19th century. It consists of 4x4 board with 15 sliding tiles numbered from 1 to 15. The objective is to get them into this pattern:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

Here we will deal with a generalized version of the above puzzle. You should write a program that given some initial state of the $n \times n$ board finds a sequence of moves that transforms it so that in the i -th row there are tiles with numbers $i*n+1, i*n+2, \dots, i*n+n$ (from left to right) - with the exception of the lower right corner where the hole should be. The less moves you use, the more points you get.

Input

The first line of input contains the number of test cases c ($c \leq 200$). Then c test cases follow, each of them begins with a line with a single integer n ($3 \leq n \leq 10$) in it. The next n lines describe the initial state of the board - the i -th line consists of exactly n integers describing the i -th row. The position of the hole is indicated by 0.

Output

For each test case output one line - the found sequence of moves. Write 'D' to move the hole down, 'U' to move it up, 'R' to move it right and 'L' to move it left. You shouldn't use more than 10000 moves. All moves should be valid (so for example don't try to move the hole up when it is in the first row).

Scoring

Your program will receive $n^3/(m+1)$ points for each test case where m is the number of moves.

Example

Input :

2

4

```
1  2  7  3
5  6  0  4
9 10 11  8
13 14 15 12
```

3
0 1 2
4 5 3
7 8 6

Output:
URDDD
RRDD

Added by: Pawel Gawrychowski
Date: 2004-07-27
Time limit [s]: 3
Source limit [B]:10000

SPOJ Problem Set

160. Choosing a Palindromic Sequence

Problem code: PALSEC

Given two sequences of words: $X=(x_1, \dots, x_n)$ and $Y=(y_1, \dots, y_n)$, determine how many binary sequences $P=(p_1, \dots, p_n)$ exist, such that the word concatenation $z_1 z_2 \dots z_n$, where $z_i = x_i$ iff $p_i = 1$ and $z_i = y_i$ iff $p_i = 0$, is a palindrome (a word which is the same when read from left to right and from right to left).

Input

The input begins with the integer t , the number of test cases. Then t test cases follow.

For each test case the first line contains the positive integer n - the number of words in a sequence ($1 \leq n \leq 30$). The following n lines contain consecutive words of the sequence X , one word per line. The next n lines contain consecutive words of the sequence Y , one word per line. Words consist of lower case letters of the alphabet ('a' to 'z'), are non-empty, and not longer than 400 characters.

Output

For each test case output one line containing a single integer - the number of different possible sequences P .

Example

Sample input:

```
1
5
ab
a
a
ab
a
a
baaaa
a
a
ba
```

Sample output:

```
12
```

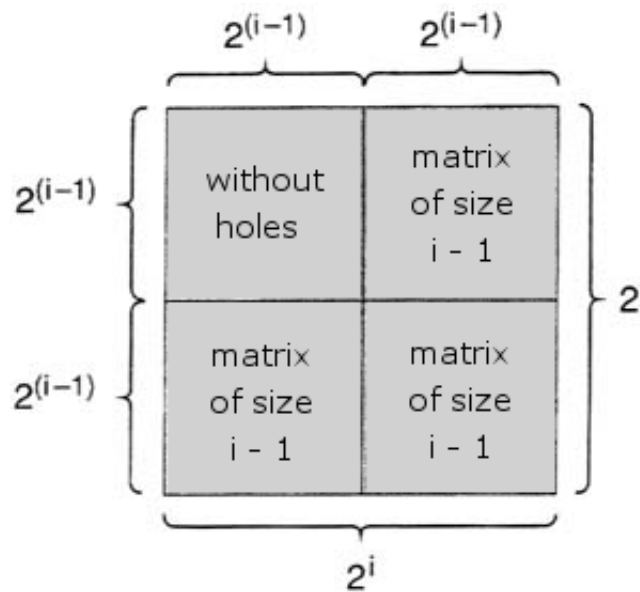
Added by: Adrian Kosowski
Date: 2004-08-07
Time limit [s]: 15
Source limit [B]: 50000
Resource: IV Polish Olympiad in Informatics (Wojciech Rytter)

SPOJ Problem Set

174. Paint templates

Problem code: PAINTTMP

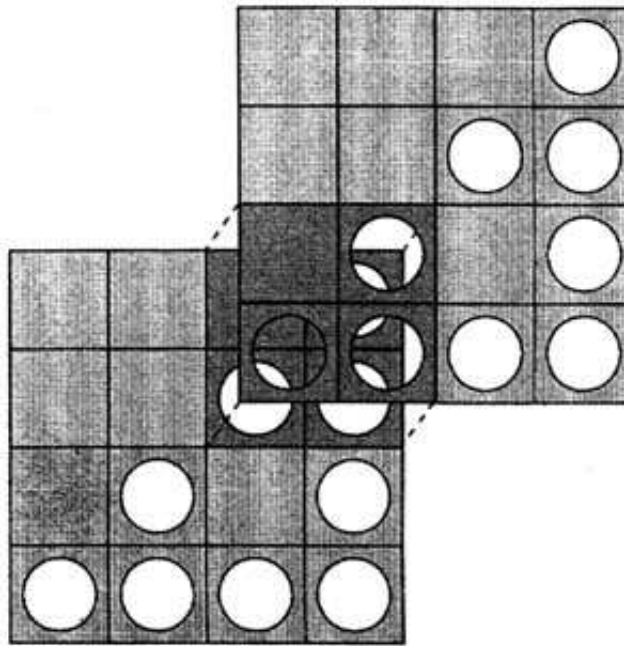
The Painter's Studio is preparing mass production of paintings. Paintings are going to be made with aid of square matrices of various sizes. A matrix of size i consists of 2^i rows and 2^i columns. There are holes on intersections of some rows and columns. Matrix of size 0 has one hole. For $i > 0$, matrix of size i is built of four squares of size $2^{(i-1)} * 2^{(i-1)}$. Look at the following figure:



Both squares on the right side and the bottom-left square are matrices of size $i-1$. Top-left square has no holes. Pictures are constructed in the following way. First, we fix three non-negative integers n , x , y . Next, we take two matrices of size n , place one of them onto the other and shift the upper one x columns right and y rows up. We place such a pattern on a white canvas and cover the common part of matrices with the yellow paint. In this way we get yellow stains on the canvas in the places where the holes in both matrices agree.

Example

Consider two matrices of size 2.



The upper matrix was shifted 2 columns right and 2 rows up. There are three places where holes agree.

Task

Write a program that for each test case:

- reads the sizes of two matrices and the numbers of columns and rows that the upper matrix should be shifted by, from the standard input;
- computes the number of yellow stains on the canvas;
- writes the result to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line

There is one integer n , $0 \leq n \leq 100$ in the first line of each test case. This number is the size of matrices used for production of paintings. In the second line there is one integer x and in the third line one integer y , where $0 \leq x, y \leq 2^n$. The integer x is the number of columns and y is the number of rows that the upper matrix should be shifted by.

Output

For each test case your program should produce one line with exactly one integer - the number of stains on the canvas.

Example

Sample input:

1
2
2
2

Sample output:

3

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 5

Source limit [B]:50000

Resource: 5th Polish Olympiad in Informatics, stage 1 (Wojciech Rytter)

SPOJ Problem Set

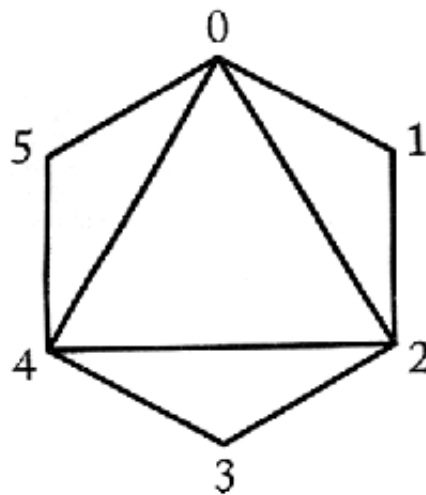
175. Polygon

Problem code: POLY1

We say that two triangles intersect if their interiors have at least one common point. A polygon is called convex if every segment connecting any two of its points is contained in this polygon. A triangle whose vertices are also vertices of a convex polygon is called an elementary triangle of this polygon. A triangulation of a convex polygon is a set of elementary triangles of this polygon, such that no two triangles from the set intersect and a union of all triangles covers the polygon. We are given a polygon and its triangulation. What is the maximal number of triangles in this triangulation that can be intersected by an elementary triangle of this polygon?

Example

Consider the following triangulation:



The elementary triangle (1,3,5) intersects all the triangles in this triangulation.

Task

Write a program that for each test case:

- reads the number of vertices of a polygon and its triangulation;
- computes the maximal number of triangles intersected by an elementary triangle of the given polygon;
- writes the result to standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line

In the first line of a test case there is a number n , $3 \leq n \leq 1000$, which equals the number of vertices of the polygon. The vertices of the polygon are numbered from 0 to $n-1$ clockwise. The following $n-2$ lines describe the triangles in the triangulation. There are three integers separated by single spaces in the $(i+1)$ -st line, where $1 \leq i \leq n-2$. These are the numbers of the vertices of the i -th triangle in the triangulation.

Output

For each test case your program should produce one line with exactly one integer - the maximal number of triangles in the triangulation, that can be intersected by a single elementary triangle of the input polygon.

Example

Sample input:

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

Sample output:

```
4
```

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 10

Source limit [B]:50000

Resource: 5th Polish Olympiad in Informatics, stage 1 (Krzysztof Diks)

SPOJ Problem Set

176. Sum of one-sequence

Problem code: SUM1SEQ

We say that a sequence of integers is a one-sequence if the difference between any two consecutive numbers in this sequence is 1 or -1 and its first element is 0. More precisely: $[a_1, a_2, \dots, a_n]$ is a one-sequence if

- for any k , such that $1 \leq k < n : |a_k - a_{k+1}| = 1$ and
- $a_1 = 0$

Task

Write a program that for each test case:

- reads two integers describing the length of the sequence and the sum of its elements;
- finds a one-sequence of the given length whose elements sum up to the given value or states that such a sequence does not exist;
- writes the result to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line.

In the first line of a test case there is a number n , such that $1 \leq n \leq 10\,000$, which is the number of elements in the sequence. In the second line there is a number S , which is the sum of the elements of the sequence, such that $|S| \leq 50\,000\,000$.

Output

For each test case there should be written n integers (each integer in a separate line) that are the elements of the sequence (k -th element in the k -th line) whose sum is S or the word "No" if such a sequence does not exist. If there is more than one solution your program should output any one.

Consequent test cases should be separated by an empty line.

Example

Sample input:

```
1
8
4
```

Sample output:

```
0
1
2
```

1
0
-1
0
1

Added by: Michal Czuczman
Date: 2004-08-10
Time limit [s]: 5
Source limit [B]:50000
Resource: 5th Polish Olympiad in Informatics, stage 1 (Grzegorz Jakacki)

SPOJ Problem Set

177. AB-words

Problem code: ABWORDS

Every sequence of small letters a and b (also the empty sequence) is called an ab-word. If $X = [x_1, \dots, x_n]$ is an ab-word and i, j are integers such that $1 \leq i \leq j \leq n$ then $X[i..j]$ denotes the subword of X consisting of the letters x_i, \dots, x_j . We say that an ab-word $X = [x_1..x_n]$ is nice if it has as many letters a as b and for all $i = 1, \dots, n$ the subword $X[1..i]$ has at least as many letters a as b.

Now, we give the inductive definition of the similarity between nice ab-words.

- Every two empty ab-words (i.e. words with no letters) are similar
- Two non-empty nice ab-words $X = [x_1, \dots, x_n]$ and $Y = [y_1, \dots, y_m]$ are similar if they have the same length ($n = m$) and one of the following conditions is fulfilled:
 1. $x_1 = y_1, x_n = y_n$ and $X[2..n-1]$ and $Y[2..n-1]$ are similar ab-words and they are both nice;
 2. there exists $i, 1 \leq i \leq n$, such that $X[1..i], X[i+1..n]$ are nice ab-words and
 - a) $Y[1..i], Y[i+1..n]$ are nice ab-words and $X[1..i]$ is similar to $Y[1..i]$ and $X[i+1..n]$ is similar to $Y[i+1..n]$, or
 - b) $Y[1..n-i], Y[n-i+1..n]$ are nice ab-words and $X[1..i]$ is similar to $Y[n-i+1..n]$ and $X[i+1..n]$ is similar to $Y[1..n-i]$.

A **level of diversity** of a non-empty set S of nice ab-words is the maximal number of ab-words that can be chosen from S in such a way that for each pair w_1, w_2 of chosen words, w_1 is not similar to w_2 .

Task

Write a program that for each test case:

- reads elements of S from standard input;
- computes the level of diversity of the set S ;
- writes the result to standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line.

In the first line of a test case there is a number n of elements of the set S , $1 \leq n \leq 1000$; in the following n lines there are elements of the set S , i.e. nice ab-words (one word in each line); the first letter of every ab-word is the first symbol in line and there are no spaces between two consecutive letters in the word; the length of every ab-word is an integer from the range $[1..200]$.

Output

For each test case your program should output one line with one integer - the level of diversity of S .

Example

Sample input:

```
1
3
aabaabbbab
abababaabb
abaaabbabb
```

Sample output:

```
2
```

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 30

Source limit [B]: 50000

Resource: 5th Polish Olympiad in Informatics, stage 1 (Krzysztof Diks)

SPOJ Problem Set

178. Road net

Problem code: ROADNET

A diskette was enclosed to a road map. The diskette contains the table of the shortest ways (distances) between each pair of towns on the map. All the roads are two-way. The location of towns on the map has the following interesting property: *if the length of the shortest way from town A to town B equals the sum of the lengths of the shortest ways from A to C and C to B then town C lies on (certain) shortest way from A to B*. We say that towns A and B are neighbouring towns if there is no town C such that the length of the shortest way from A to B equals the sum of the lengths of the shortest ways from A to C and C to B. Find all the pairs of neighbouring towns.

Example

For the table of distances:

	A	B	C
A	0	1	2
B	1	0	3
C	2	3	0

the neighbouring towns are A, B and A, C.

Task

Write a program that for each test case:

- reads the table of distances from standard input;
- finds all the pairs of neighbouring towns;
- writes the result to standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line.

In the first line of each test case there is an integer n , $1 \leq n \leq 200$, which equals the number of towns on the map. Towns are numbered from 1 to n .

The table of distances is written in the following n lines. In the $(i+1)$ -th line, $1 \leq i \leq n$, there are n non-negative integers not greater than 200, separated by single spaces. The j -th integer is the distance between towns i and j .

Output

For each test case your program should write all the pairs of the neighbouring towns (i.e. their numbers). There should be one pair in each line. Each pair can appear only once. The numbers in each pair should be given in increasing order. Pairs should be ordered so that if the pair (a, b) precedes the pair (c, d) then $a < c$ or $(a = c \text{ and } b < d)$.

Consequent test cases should be separated by an empty line.

Example

Sample input:

```
1
3
0 1 2
1 0 3
2 3 0
```

Sample output:

```
1 2
1 3
```

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 10

Source limit [B]: 50000

Resource: 5th Polish Olympiad in Informatics, stage 2 (Piotr Chrzastowski-Wachtel)

SPOJ Problem Set

179. Word equations

Problem code: WORDEQ

Every non-empty sequence of elements 0 and 1 is called a binary word. A word equation is an equation of the form $x_1x_2...x_l = y_1y_2...y_r$, where x_i and y_j are binary digits (0 or 1) or variables i.e. small letters of English alphabet. For every variable there is a fixed length of the binary words that can be substituted for this variable. This length is called a length of variable. In order to solve a word equation we have to assign binary words of appropriate length to all variables (the length of the word assigned to the variable x has to be equal to the length of this variable) in such a way that if we substitute words for variables then both sides of the equation (which are binary words after substitution) become equal.

For a given equation compute how many distinct solutions it has.

Example

Let a, b, c, d, e be variables and let 4, 2, 4, 4, 2 be their lengths (4 is the length of a, 2 is the length of b etc.). Consider the equation:

1bad1 = acbe

This equation has 16 distinct solutions.

Input

The number of equations t is in the first line of input, then t descriptions of equations follow separated by an empty line.

Each description consists of 6 lines. An equation is described in the following way: in the first line of the description there is an integer k , $0 \leq k \leq 26$, which denotes the number of distinct variables in the equation. We assume that variables are the first k small letters of English alphabet. In the second line there is a sequence of k positive integers separated by single spaces. These numbers denote the lengths of variables a, b, ... from the equation (the first number is the length of a, the second - b, etc.). There is an integer l in the third line of the description, which is the length of the left size of equation, i.e. the length of the word built of digits 0 or 1 and variables (single letters). The left side of the equation is written in the next line as a sequence of digits and variables with no spaces between them. The next two lines contain the description of the right side of the equation. The first of these lines contains a positive integer r , which is the length of the right side of the equation. The second line contains the right side of the equation which is encoded in the same way as the left side. The number of digits plus sum of the lengths of variables (we count all appearances of variables) on each side of the equation is not greater than 10000.

Output

For each equation your program should output one line with the number of distinct solutions.

Example

Sample input:

```
1
5
4 2 4 4 2
5
1bad1
4
acbe
```

Sample output:

```
16
```

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 10

Source limit [B]:50000

Resource: 5th Polish Olympiad in Informatics, stage 2 (Wojciech Rytter)

SPOJ Problem Set

180. How to pack containers

Problem code: CONTPACK

Products of a factory are packed into cylindrical boxes. All boxes have the same bases. A height of a box is a non-negative integer being a power of 2, i.e. it is equal to 2^i for some $i = 0, 1, 2, \dots$. The number i (exponent) is called a size of a box. All boxes contain the same goods but their value may be different. Goods produced earlier are cheaper. The management decided, that the oldest (cheapest) goods should be sold out first. From the warehouse goods are transported in containers. Containers are also cylindrical. A diameter of each container is a little bigger than a diameter of a box, so that boxes can be easily put into containers. A height of a container is a non-negative power of 2. This number is called a size of a container. For safe transport containers should be tightly packed with boxes, i.e. the sum of heights of boxes placed in a container have to be equal to the height of this container. A set of containers was delivered to the warehouse. Check if it is possible to pack all the containers tight with boxes that are currently stored in the warehouse. If so, find the minimal value of goods that can be tightly packed into these containers.

Consider a warehouse with 5 boxes. Their sizes and values of their contents are given below:

```
1 3
1 2
3 5
2 1
1 4
```

Two containers of size 1 and 2 can be tightly packed with two boxes of total value 3, 4 or 5, or three boxes with total value 9. The container of size 5 cannot be tightly packed with boxes from the warehouse.

Task

Write a program that for each test case:

- reads descriptions of boxes (size, value) from a warehouse and descriptions of containers (how many containers of a given size we have);
- checks if all containers can be tightly packed with boxes from the warehouse and if so, computes the minimal value of goods that can be tightly packed into these containers;
- writes the result.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line.

In the first line of a test case there is an integer n , $1 \leq n \leq 10000$, which is the number of boxes in the warehouse. In each of the following n lines there are written two non-negative integers separated by a single space. These numbers describe a single box. First of them is the size of the box and the second - the value of goods contained in this box. The size is not greater than 1000 and the value is not

greater than 10000. The next line contains a positive integer q , which is the number of different sizes of containers delivered to the warehouse. In each of the following q lines there are two positive integers separated by a single space. The first integer is the size of a container and the second one is the number of containers of this size. The maximal number of containers is 5000, a size of a container is not greater than 1000.

Output

For each test case your program should output exactly one line containing:

- a single word "No" if it is not possible to pack the containers from the given set tight with the boxes from the warehouse, or
- a single integer equal to the minimal value of goods in boxes with which all the containers from the given set can be packed tight.

Example

Sample input:

```
1
5
1 3
1 2
3 5
2 1
1 4
2
1 1
2 1
```

Sample output:

```
3
```

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 10

Source limit [B]: 50000

Resource: 5th Polish Olympiad in Informatics, stage 2 (Wojciech Rytter)

SPOJ Problem Set

181. Scuba diver

Problem code: SCUBADIV

A scuba diver uses a special equipment for diving. He has a cylinder with two containers: one with oxygen and the other with nitrogen. Depending on the time he wants to stay under water and the depth of diving the scuba diver needs various amount of oxygen and nitrogen. The scuba diver has at his disposal a certain number of cylinders. Each cylinder can be described by its weight and the volume of gas it contains. In order to complete his task the scuba diver needs specific amount of oxygen and nitrogen. What is the minimal total weight of cylinders he has to take to complete the task?

Example

The scuba diver has at his disposal 5 cylinders described below. Each description consists of: volume of oxygen, volume of nitrogen (both values are given in litres) and weight of the cylinder (given in decagrams):

```
3 36 120
10 25 129
5 50 250
1 45 130
4 20 119
```

If the scuba diver needs 5 litres of oxygen and 60 litres of nitrogen then he has to take two cylinders of total weight 249 (for example the first and the second ones or the fourth and the fifth ones).

Task

Write a program that for each test case:

- reads scuba diver's demand for oxygen and nitrogen, the number of accessible cylinders and their descriptions;
- computes the minimal total weight of cylinders the scuba diver needs to complete his task;
- outputs the result.

Note: the given set of cylinders always allows to complete the given task.

Input

The number of test cases c is in the first line of input, then c test cases follow separated by an empty line.

In the first line of a test case there are two integers t, a separated by a single space, $1 \leq t \leq 21$ and $1 \leq a \leq 79$. They denote volumes of oxygen and nitrogen respectively, needed to complete the task. The second line contains one integer n , $1 \leq n \leq 1000$, which is the number of accessible cylinders. The following n lines contain descriptions of cylinders; i -th line contains three integers t_i, a_i, w_i separated by single spaces, ($1 \leq t_i \leq 21$, $1 \leq a_i \leq 79$, $1 \leq w_i \leq 800$). These are respectively: volume of oxygen and nitrogen in the i -th cylinder and the weight of this cylinder.

Output

For each test case your program should output one line with the minimal total weight of cylinders the scuba diver should take to complete the task.

Example

Sample input:

```
1
5 60
5
3 36 120
10 25 129
5 50 250
1 45 130
4 20 119
```

Sample output:

```
249
```

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 10

Source limit [B]:50000

Resource: 5th Polish Olympiad in Informatics, stage 2

SPOJ Problem Set

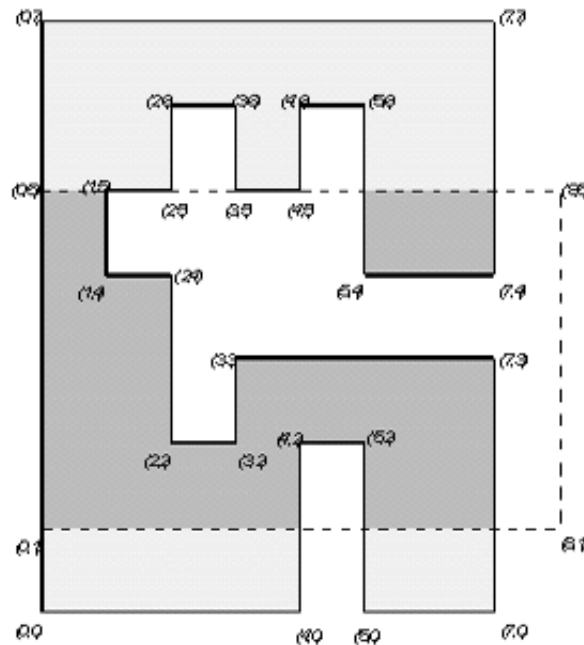
182. Window

Problem code: WINDOW1

We have a polygon chosen in the cartesian coordinate system. Sides of the polygon are parallel to the axes of coordinates. Every two consecutive sides are perpendicular and coordinates of every vertex are integers. We have also given a window that is a rectangle whose sides are parallel to the axes of coordinates. The interior of the polygon (but not its periphery) is coloured red. What is the number of separate red fragments of the polygon that can be seen through the window?

Example

Look at the figure below:



There are two separate fragments of the polygon that can be seen through the window.

Task

Write a program that for each test case:

- reads descriptions of a window and a polygon;
- computes the number of separate red fragments of the polygon that can be seen through the window;
- outputs the result.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line.

In the first line of a test case there are four integers x_1, y_1, x_2, y_2 from the range $[0..10000]$, separated by single spaces. The numbers x_1, y_1 are the coordinates of the top-left corner of the window. The numbers x_2, y_2 are the coordinates of the bottom-right corner of the window. The next line of the input file contains one integer n , $4 \leq n \leq 5000$, which equals the number of vertices of the polygon. In the following n lines there are coordinates of polygon's vertices given in anticlockwise direction, i.e. the interior of the polygon is on the left side of its periphery when we move along the sides of the polygon according to the given order. Each line contains two integers x, y separated by a single space, $0 \leq x \leq 10000, 0 \leq y \leq 10000$. The numbers in the i -th line, are coordinates of the i -th vertex of the polygon.

Output

For each test case you should output one line with the number of separate red fragments of the polygon that can be seen through the window.

Example

Sample input:

```
1
0 5 8 1
24
0 0
4 0
4 2
5 2
5 0
7 0
7 3
3 3
3 2
2 2
2 4
1 4
1 5
2 5
2 6
3 6
3 5
4 5
4 6
5 6
5 4
7 4
7 7
0 7
```

Sample output:

```
2
```

Added by: Michal Czuczman
Date: 2004-08-10
Time limit [s]: 10
Source limit [B]:50000
Resource: 5th Polish Olympiad in Informatics, stage 2 (Wojciech Guzicki)

SPOJ Problem Set

183. Assembler circuits

Problem code: ASCIRC

Bytetel Company decided to improve computers they produce. They want to replace assembler programs with special systems called assembler circuits. Assembler programs consist solely of assignments. Each assignment is determined by four elements:

- two registers from which data are taken,
- elementary operation that should be performed on the data,
- register to which the result should be written.

We assume that there are at most 26 registers. They are represented by small letters of English alphabet. There are at most 4 elementary operations and they are represented by capital letters A, B, C, D.

An assembler circuit has:

- inputs assigned to registers; initial value of appropriate register is passed to the input;
- outputs, also assigned to registers; their final values are passed to these registers.

There are gates inside a circuit. Each gate has two inputs and one output. The gate performs an elementary operation on data delivered on its inputs and passes the result to its output. Inputs of gates and outputs of the whole circuit are connected to outputs of other gates or inputs of the circuit. Outputs of gates and inputs of the circuit can be connected to many inputs of other gates or outputs of the circuit. Connections among gates cannot form cycles.

An assembler circuit is equivalent to an assembler program if for any initial state of registers the final state of registers produced by the program and the circuit are the same.

Task

Write a program that for each test case:

- reads a description of an assembler program;
- computes the minimal number of gates in an assembler circuit equivalent to the given program;
- writes the result.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line.

In the first line of each test case there is one integer n ($1 \leq n \leq 1000$), which is the number of instructions in the program.

In the following n lines there are descriptions of consecutive instructions in the program. Each description is a four-letter word beginning with an elementary operation symbol: A, B, C or D. The second and the third letter (which are small letters of English alphabet) are names of registers, in which data are placed. The fourth letter is a name of a register, in which the result should be placed.

Output

For each test case you should output one line with the minimal number of gates in an assembler circuit equivalent to the given program.

Example

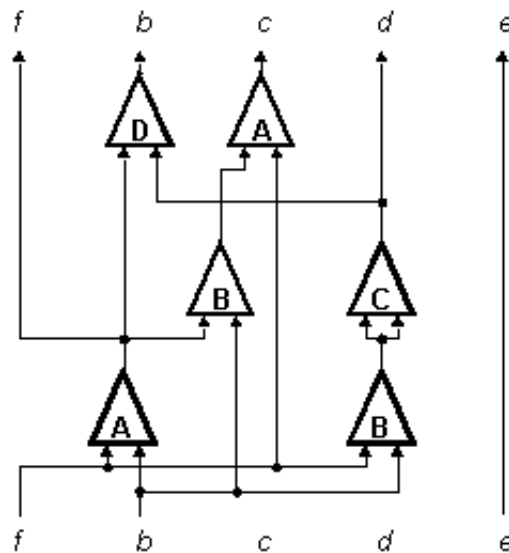
Sample input:

```
1
8
Afbc
Bfbd
Cddd
Bcbc
Afcc
Afbf
Cfbb
Dfdb
```

Sample output:

```
6
```

A circuit equivalent to the given program is shown in the figure.



Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 10

Source limit [B]: 50000

Resource: 5th Polish Olympiad in Informatics, stage 3 (Marcin Kubica)

SPOJ Problem Set

184. Automatic Teller Machines

Problem code: ATMS

Every member of Byteland Credit Society is entitled to loan any amount of Bytelandish ducats unless it is 10^{30} or more, but he has to return the whole amount within seven days. There are 100 ATMs in the Client Service Room of the Society. They are numbered from 0 to 99. Every ATM can perform one action only: it can pay or receive a fixed amount. The i -th ATM pays 2^i ducats if i is even or it receives 2^i ducats if i is odd. If a client is going to loan a fixed sum of money it is necessary to check if he is able to get the money using every ATM at most once. If so, numbers of ATMs he has to use should be determined. It is also necessary to check if the client can return the money in a similar way, and if so, to determine numbers of ATMs he has to use.

Example

A client who is going to loan 7 ducats gets 16 ducats from the ATM # 4 and 1 ducat from the ATM # 0 and then he returns 8 ducats in the ATM # 3 and 2 ducats in the ATM # 1. In order to return the amount of 7 ducats he receives 1 ducat from the ATM # 0 and then he returns 8 ducats in ATM # 3.

Task

Write a program that:

- reads the number of clients n , for every client reads from the same file the amount of money he is going to loan;
- checks for every client if he is able to get the money using every ATM at most once and if so, determines the numbers of ATMs he has to use;
- outputs the results.

Input

In the first line of input there is one positive integer $n \leq 10000$, which equals the number of clients.

In each of the following n lines there is one positive integer less than 10^{30} (at most 30 decimal digits). The number in the i -th line describes the amount of ducats which the client i is going to loan.

Output

For each client you should output two lines with a decreasing sequence of positive integers from the range $[0..99]$ separated by single spaces, or one word "No":

In the first line of the i -th pair of lines there should be numbers of ATMs (in decreasing order) that the client i should use to get his loan or one word "No" if the loan cannot be received according to the rules;

In the second line of the i -th pair there should be numbers of ATMs (in decreasing order) which the client i should use to return his loan or the word "No".

Example

Sample input:

```
2
7
633825300114114700748351602698
```

Sample output:

```
4 3 1 0
3 0
No
99 3 1
```

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 10

Source limit [B]:50000

Resource: 5th Polish Olympiad in Informatics, stage 3 (Piotr Chrzastowski-Wachtel)

SPOJ Problem Set

185. Chase

Problem code: CHASE1

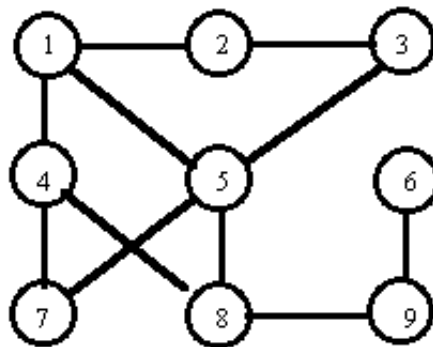
Chase is a two-person board game. A board consists of squares numbered from 1 to n . For each pair of different squares it is known if they are adjacent to one another or they are not. Each player has a piece at his disposal. At the beginning of a game pieces of players are placed on fixed, distinct squares. In one turn a player can leave his piece on the square it stands or move it to an adjacent square.

A game board has the following properties:

- it contains no triangles, i.e. there are no three distinct squares such that each pair of them is adjacent,
- each square can be reached by each player.

A game consists of many turns. In one turn each player makes a single move. Each turn is started by player A. We say that player A is caught by player B if both pieces stand on the same square. Decide, if for a given initial positions of pieces, player B can catch player A, independently of the moves of his opponent. If so, how many turns player B needs to catch player A if both play optimally (i.e. player A tries to run away as long as he can and player B tries to catch him as quickly as possible).

Example



Consider the board in the figure. Adjacent squares (denoted by circles) are connected by edges. If at the beginning of a game pieces of players A and B stand on the squares 9 and 4 respectively, then player B can catch player A in the third turn (if both players move optimally). If game starts with pieces on the squares 8 (player A) and 4 (player B) then player B cannot catch player A (if A plays correctly).

Task

Write a program that for each test case:

- reads the description of a board and numbers of squares on which pieces are placed initially.
- decides if player B can catch player A and if so, computes how many turns he needs (we assume that both players play optimally);
- outputs the result.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line.

In the first line of a test case there are four integers n , m , a and b separated by single spaces, where $2 \leq n \leq 3000$, $n-1 \leq m \leq 15000$, $1 \leq a, b \leq n$. These are (respectively): the number of squares of the board, the number of adjacent (unordered) pairs, the number of the square on which the piece of player A is placed, the number of the square on which the piece of player B is placed. In each of the following lines there are two distinct positive integers separated by a single space, which denote numbers of adjacent squares.

Output

For each test case you should output one line containing:

- one word "No", if player B cannot catch player A, or
- one integer - the number of turns needed by B to catch A (if B can catch A).

Example

Sample input:

```
1
9 11 9 4
1 2
3 2
1 4
4 7
7 5
5 1
6 9
8 5
9 8
5 3
4 8
```

Sample output:

```
3
```

Added by: Michal Czuczman
 Date: 2004-08-10
 Time limit [s]: 10
 Source limit [B]: 50000
 Resource: 5th Polish Olympiad in Informatics, stage 3 (Adam Borowski)

SPOJ Problem Set

186. The lightest language

Problem code: LITELANG

Alphabet A_k consists of k initial letters of English alphabet. A positive integer called a weight is assigned to each letter of the alphabet. A weight of a word built from the letters of the alphabet A_k is the sum of weights of all letters in this word. A language over an alphabet A_k is any finite set of words built from the letters of this alphabet. A weight of a language is the sum of weights of all its words. We say that the language is prefixless if for each pair of different words w, v from this language w is not a prefix of v .

We want to find out what is the minimal possible weight of an n -element, prefixless language over an alphabet A_k .

Example

Assume that $k = 2$, the weight of the letter a is $W(a) = 2$ and the weight of the letter b is $W(b) = 5$. The weight of the word ab is $W(ab) = 2 + 5 = 7$. $W(aba) = 2 + 5 + 2 = 9$. The weight of the language $J = \{ab, aba, b\}$ is $W(J) = 21$. The language J is not prefixless, since the word ab is a prefix of aba. The lightest three-element, prefixless language over the alphabet A_2 (assuming that weights of the letters are as before) is $\{b, aa, ab\}$; its weight is 16.

Task

Write a program that for each test case:

- reads two integers n, k and the weights of k letters of an alphabet A_k ;
- computes the minimal weight of a prefixless, n -element language over the alphabet A_k ;
- outputs the result.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line.

In the first line of a test case there are two positive integers n and k separated by a single space, ($2 \leq n \leq 10000$, $2 \leq k \leq 26$). These are the number of words in a language and the number of letters in an alphabet respectively. The second line contains k positive integers separated by single spaces. Each of them is not greater than 10000. The i -th number is the weight of the i -th letter.

Output

For each test case you should output one line with the weight of the lightest prefixless n -element language over the alphabet A_k .

Example

Sample input:

```
1
3 2
2 5
```

Sample output:

```
16
```

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 10

Source limit [B]:50000

Resource: 5th Polish Olympiad in Informatics, stage 3 (Wojciech Rytter)

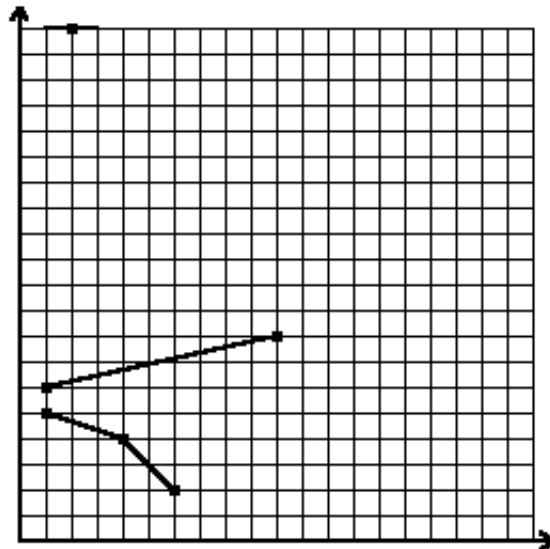
SPOJ Problem Set

187. Flat broken lines

Problem code: FLBRKLIN

We have a cartesian coordinate system drawn on a sheet of paper. Let us consider broken lines that can be drawn with a single pencil stroke from the left to the right side of the sheet. We also require that for each segment of the line the angle between the straight line containing this segment and the OX axis belongs to $[-45^\circ, 45^\circ]$ range. A broken line fulfilling above conditions is called a flat broken line. Suppose we are given n distinct points with integer coordinates. What is the minimal number of flat broken lines that should be drawn in order to cover all the points (a point is covered by a line if it belongs to this line)?

Example



For 6 points whose coordinates are (1,6), (10,8), (1,5), (2,20), (4,4), (6,2) the minimal number of flat broken lines covering them is 3.

Task

Write a program that for each test case:

- reads the number of points and their coordinates;
- computes the minimal number of flat broken lines that should be drawn to cover all the points;
- outputs the result.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line.

In the first line of a test case there is one positive integer n , not greater than 30000, which denotes the number of points. In the following n lines there are coordinates of points. Each line contains two integers x, y separated by a single space, $0 \leq x \leq 30000$, $0 \leq y \leq 30000$. The numbers in the i -th line are the coordinates of the i -th point.

Output

For each test case you should output one line with the minimal number of flat broken lines that should be drawn to cover all the points.

Example

Sample input:

```
1
6
1 6
10 8
1 5
2 20
4 4
6 2
```

Sample output:

```
3
```

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 10

Source limit [B]: 50000

Resource: 5th Polish Olympiad in Informatics, stage 3 (Grzegorz Jakacki, Krzysztof Sobusiak)

SPOJ Problem Set

188. Rectangles

Problem code: RECTNG1

There are n rectangles drawn on the plane. Each rectangle has sides parallel to the coordinate axes and integer coordinates of vertices.

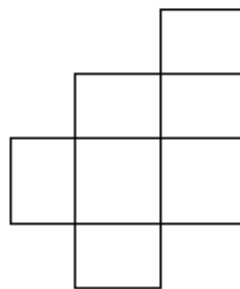
We define a block as follows:

- each rectangle is a block,
- if two distinct blocks have a common segment then they form the new block otherwise we say that these blocks are separate.

Examples

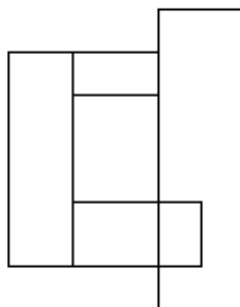
The rectangles in Figure 1 form two separate blocks.

Figure 1



The rectangles in Figure 2 form a single block

Figure 2



Task

Write a program that for each test case:

- reads the number of rectangles and coordinates of their vertices;
- finds the number of separate blocks formed by the rectangles;
- writes the result to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line.

In the first line of a test case there is an integer n , $1 \leq n \leq 7000$, which is the number of rectangles. In the following n lines there are coordinates of rectangles. Each rectangle is described by four numbers: coordinates x, y of the bottom-left vertex and coordinates x, y of the top-right vertex. All these coordinates are non-negative integers not greater than 10000.

Output

For each test case you should output one line with the number of separate blocks formed by the given rectangles.

Example

Sample input:

```
1
9
0 3 2 6
4 5 5 7
4 2 6 4
2 0 3 2
5 3 6 4
3 2 5 3
1 4 4 7
0 0 1 4
0 0 4 1
```

Sample output:

```
2
```

Added by: Michal Czuczman

Date: 2004-08-10

Time limit [s]: 10

Source limit [B]: 50000

Resource: 5th Polish Olympiad in Informatics, stage 3 (Wojciech Rytter)

SPOJ Problem Set

196. Musketeers

Problem code: MUSKET

In the time of Louis XIII and his powerful minister cardinal Richelieu in the Full Barrel Inn n musketeers had consumed their meal and were drinking wine. Wine had not run short and therefore the musketeers were eager to quarrel, a drunken brawl broke out, in which each musketeer insulted all the others.

A duel was inevitable. But who should fight who and in what order? They decided (for the first time since the brawl they had done something together) that they would stay in a circle and draw lots in order. A drawn musketeer fought against his neighbor to the right. A loser "quit the game" and to be more precise his corpse was taken away by servants. The next musketeer who stood beside the loser became the neighbor of a winner.

After years, when historians read memories of the winner they realized that a final result depended in a crucial extent on the order of duels. They noticed that a fence practice had indicated, who against who could win a duel. It appeared that (in mathematical language) the relation "**A** wins **B**" was not transitive! It could happen that the musketeer **A** fought better than **B**, **B** better than **C** and **C** better than **A**. Of course, among three of them the first duel influenced the final result. If **A** and **B** fight as the first, **C** wins eventually. But if **B** and **C** fight as the first, **A** wins finally. Historians fascinated by their discovery decided to verify which musketeers could survive. The fate of France and the whole civilized Europe indeed depended on that!

Task

N persons with consecutive numbers from 1 to n stay in a circle. They fight $n-1$ duels. In the first round one of these persons (e.g. with the number i) fights against its neighbor to the right, i.e. against the person numbered $i+1$ (or, if $i=n$, against the person numbered 1). A loser quits the game, and the circle is tighten so that the next person in order becomes a winner's neighbor. We are given the table with possible duels results, in the form of a matrix. If $A_{i,j} = 1$ then the person with the number i always wins with the person j . If $A_{i,j} = 0$ the person i loses with j . We can say that the person k may win the game if there exists such a series of $n-1$ drawings, that k wins the final duel.

Write a program which:

- reads matrix **A** from the standard input,
- computes numbers of persons, who may win the game,
- writes them into the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case integer n which satisfies the inequality $3 \leq n \leq 100$ is written. In each of the following n lines appears one word consisting of n digits 0 or 1. A digit on j -th position in i -th line denote $A_{i,j}$. Of course $A_{i,j} = 1 - A_{j,i}$, for $i < j$. We assume that $A_{i,i} = 1$, for each i .

Output

For each test case in the first line there should be written m - the number of persons, who may win the game. In the following m lines numbers of these persons should be written in ascending order, one number in each line.

Example

Sample input:

```
1
7
1111101
0101100
0111111
0001101
0000101
1101111
0100001
```

Sample output:

```
3
1
3
6
```

The order of duels: 1-2, 1-3, 5-6, 7-1, 4-6, 6-1 gives a final victory to the person numbered 6. You can also check that only two persons more (1 and 3) may win the game.

Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]: 50000

Resource: 6th Polish Olympiad in Informatics, stage 1

SPOJ Problem Set

199. Empty Cuboids

Problem code: EMPTY

We call a cuboid **regular** if:

- one of its vertices is a point with coordinates (0,0,0),
- edges beginning in this vertex lie on the positive semi-axes of the coordinate system,
- the edges are not longer than 10^6

There is given a set **A** of points of space, whose coordinates are integers from the interval $[1..10^6]$. We try to find a regular cuboid of maximal volume which does not contain any of the points from the set **A**. A point belongs to the cuboid if it belongs to the interior of the cuboid, i.e. it is a point of the cuboid, but not of its wall.

Task

Write a program which:

- reads from the standard input the coordinates of points from the set **A**,
- finds one of the regular cuboids of maximal volume which does not contain any points from the set **A**,
- writes the result to standard output.

Input

Input begins with a line containing integer $t \leq 10$, the number of test cases. t test cases follow.

In the first line of each test case one non-negative integer n is written ($n \leq 5000$). It is the number of elements in the set **A**. In the following n lines of the input there are triples of integers from the interval $[1..10^6]$, which are the X, Y and Z coordinates of points from **A**, respectively. Numbers in each line are separated by single spaces.

Output

For each test case there should be three integers separated by single spaces. These are the X, Y and Z coordinates (respectively) of the vertex of the regular cuboid of maximal volume. If there is more than one such a cuboid, choose whichever. We require that all coordinates be positive.

Example

Sample input:

```
1
4
3 3 300000
2 200000 5
90000 3 2000
```

2 2 1000

Sample output:

1000000 200000 1000

Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]: 50000

Resource: 6th Polish Olympiad in Informatics, stage 1

SPOJ Problem Set

200. Monodigital Representations

Problem code: MONODIG

Let K be a decimal digit different from 0. We say that an arithmetic expression is a **K-representation of the integer X** if a value of this expression is X and if it contains only numbers composed of a digit K . (All the numbers are of course decimal). The following arithmetical operations are allowed in the expression: addition, subtraction, multiplication and division. Round brackets are allowed too. Division may appear only when a dividend is a multiple of a divisor.

Example

Each of the following expressions is the 5-representation of the number 12:

- $5+5+(5:5)+(5:5)$
- $(5+(5))+5:5+5:5$
- $55:5+5:5$
- $(55+5):5$

The **length** of the K -representation is the number of occurrences of digit K in the expression. In the example above the first two representations have the length 6, the third - 5, and the forth - 4.

Task

Write a program which:

- reads the digit K and the series of numbers from the standard input,
- verifies for each number from the series, whether it has a K -representation of length at most 8, and if it does, then the program finds the minimal length of this representation,
- writes results to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. The first line of each test case contains digit K , K is an element of $\{1, \dots, 9\}$. The second line contains number n , $1 \leq n \leq 10$. In the following n lines there is the series of natural numbers a_1, \dots, a_n , $1 \leq a_i \leq 32000$ (for $i=1, \dots, n$), one number in each line.

Output

The output for each test case composes of n lines. The i -th line should contain:

- exactly one number which is the minimal length of K -representation of a_i , assuming that such a representation of length not greater than 8 exists,
- one word NO, if the minimal length of the K -representation of the number a_i is greater than 8.

Example

Sample input:

```
1
5
2
12
31168
```

Sample output

```
4
NO
```

Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]:50000

Resource: 6th Polish Olympiad in Informatics, stage 1

SPOJ Problem Set

201. The Game of Polygons

Problem code: POLYGAME

Two players take part in the game **polygons**. A convex polygon with n vertices divided by $n-3$ diagonals into $n-2$ triangles is necessary. These diagonals may intersect in vertices of the polygon only. One of the triangles is black and the remaining ones are white. Players proceed in alternate turns. Each player, when its turn comes, cuts away one triangle from the polygon. players are allowed to cut off triangles along the given diagonals. The winner is the player who cuts away the black triangle.

NOTE: We call a polygon **convex** if a segment joining any two points of the polygon is contained in the polygon.

Task

Write a program which:

- reads from the standard input the description of the polygon,
- verifies whether the player who starts the game has a winning strategy,
- writes the result to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. The first line of each test case contains an integer n , $4 \leq n \leq 50000$. This is the number of vertices in the polygon. The vertices of the polygon are numbered, clockwise, from 0 to $n-1$.

The next $n-2$ lines comprise descriptions of triangles in the polygon. In the $i+1$ -th line, $1 \leq i \leq n-2$, there are three non-negative integers a, b, c separated by single spaces. These are numbers of vertices of the i -th triangle. The first triangle in a sequence is black.

Output

The output for each test case should have one line with the word:

- YES, if the player, who starts the game has a winning strategy,
- NO, if he does not have a winning strategy.

Example

Sample input:

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

Sample output:

```
YES
```

Warning: large Input/Output data, be careful with certain languages

Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]: 50000

Resource: 6th Polish Olympiad in Informatics, stage 1

SPOJ Problem Set

202. Rockets

Problem code: ROCKETS

There are two separate, n -element sets of points of a two dimensional map: **R** and **W**. None triple of points from the set **R** is collinear. Rockets earth-to-earth are located on points from the set **R**. Enemy objects, which should be destroyed, are located on points from the set **W**. The rockets may fly only in the straight line and their trajectories cannot intersect. We are about to find for each rocket a target to destroy.

Task

Write a program which:

- reads from the standard input coordinates of the points from the sets **R** and **W**,
- finds the set of n pairwise not-intersecting segments, so that one end of each segment belongs to the set **R**, while the other belongs to the set **W**,
- writes the result into the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case there is written one integer n , $1 \leq n \leq 10000$, equal to the number of elements of the sets **R** and **W**.

In each of the following $2n$ lines of the input one pair of integer numbers from the interval $[-10000, 10000]$ is written. Numbers in each pair are separated by a single space. They are coordinates of the point on a map (first coordinate x , then y). The first n lines comprise coordinates of the points from the set **R**, the last n lines comprise the points from the set **W**. In the $(i+1)$ -th line there are coordinates of the point r_i , in the $(i+n+1)$ -th line there are coordinates of the point w_i , $1 \leq i \leq n$.

Output

The output for each test case should consist of n lines. In the i -th line there should be one integer $k(i)$, such that the segment $r_i w_{k(i)}$ belongs to the set of segments which your program found. (This means that the rocket from the point r_i destroys an object in the point $w_{k(i)}$).

Example

Sample input:

```
1
4
0 0
1 5
4 2
2 6
1 2
5 4
```

4 5
3 1

Sample output:

2
1
4
3

Warning: large Input/Output data, be careful with certain languages

Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]: 50000

Resource: 6th Polish Olympiad in Informatics, stage 2

SPOJ Problem Set

203. Potholers

Problem code: POTHOLE

A team of speleologists organizes a training in the Grate Cave of Byte Mountains. During the training each speleologist explores a route from Top Chamber to Bottom Chamber. The speleologists may move down only, i.e. the level of every consecutive chamber on a route should be lower then the previous one. Moreover, each speleologist has to start from Top Chamber through a different corridor and each of them must enter Bottom Chamber using different corridor. The remaining corridors may be traversed by more then one speleologist. How many speleologists can train simultaneously?

Task

Write a program which:

- reads the cave description from the standard input,
- computes the maximal number of speleologists that may train simultaneously,
- writes the result to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case there is one integer n ($2 \leq n \leq 200$), equal to the number of chambers in the cave. The chambers are numbered with integers from 1 to n in descending level order - the chamber of grater number is at the higher level than the chamber of the lower one. (Top Chamber has number 1 , and Bottom Chamber has number n). In the following $n-1$ lines (i.e. lines $2, 3, \dots, n$) the descriptions of corridors are given. The $(i+1)$ -th line contains numbers of chambers connected by corridors with the i -th chamber. (only chambers with numbers grater then i are mentioned). The first number in a line, m , $0 \leq m \leq (n-i+1)$, is a number of corridors exiting the chamber being described. Then the following m integers are the numbers of the chambers the corridors are leading to.

Output

Your program should write one integer for each test case. This number should be equal to the maximal number of speleologists able to train simultaneously,

Example

Sample input:

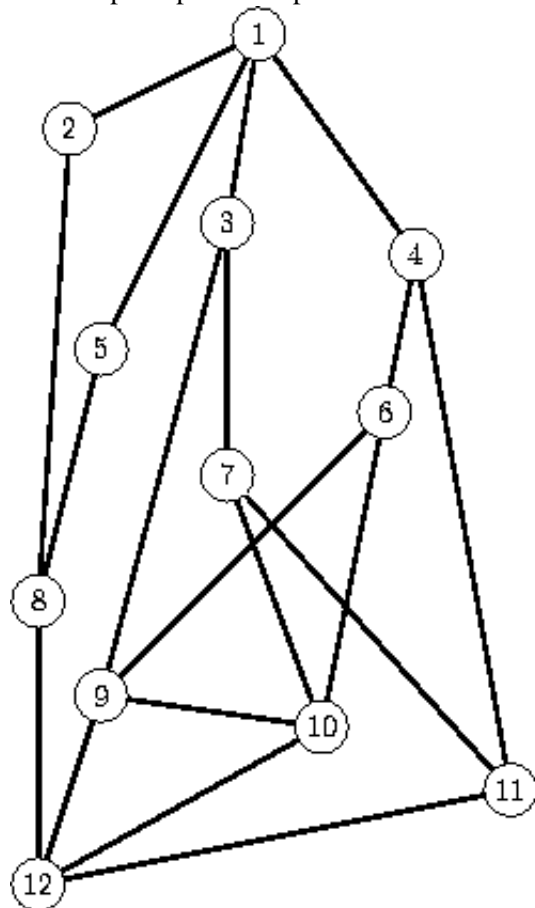
```
1
12
4 3 4 2 5
1 8
2 9 7
2 6 11
1 8
2 9 10
2 10 11
1 12
```

```
2 10 12
1 12
1 12
```

Sample output:

3

The sample input corresponds to the following cave:



Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]: 50000

Resource: 6th Polish Olympiad in Informatics, stage 2

SPOJ Problem Set

204. Sleepwalker

Problem code: SLEEP

There is a building with flat square roof of size $3^k * 3^k$ and sides parallel to north-south and east-west directions. The roof is covered with square tiles of size l (with a side of length 1), but one of the tiles has been removed and there is a hole in the roof (big enough to fall in). The tiles form a rectangular mesh on the roof, so their positions may be specified with coordinates. The tile at the southwestern corner has coordinates $(1,1)$. The first coordinate increases while going eastwards, and the second while going northwards.

Sleepwalker wanders across the roof, in each step moving from the tile he is standing on to the adjacent one on the east(E), west(W), south(S), or north(N). The sleepwalker roof ramble starts from the southwestern corner tile. The description of the path is a word d_k built of the letters N, S, E, W denoting respectively a step to the north, south, east and west. For $k = 1$ the word describing the path of sleepwalker is

$$d_1 = \text{EENNWSWN}$$

For $k = 2$ the word describing the path of sleepwalker is

$$d_2 = \begin{array}{l} \text{NNEESWSEENNEESWSEEEENNWSWNNEENNWSW} - \\ \text{NNEENNWSWNWWWSSSENESSSSWWNENWWSSW} - \\ \text{WNENWNEENNWSWN} . \end{array}$$

(See the picture that shows how the sleepwalker would go across a roof of dimension $3*3$ or $9*9$.) Generally, if $k \geq 1$, the description of a sleepwalker's path on the roof of dimension $3^{k+1} * 3^{k+1}$ is a word:

$$d_{k+1} = a(d_k) E a(d_k) E d_k N d_k N d_k W c(d_k) S b(d_k) W b(d_k) N d_k$$

where functions **a**, **b** and **c** denote the following permutations of letters specifying directions:

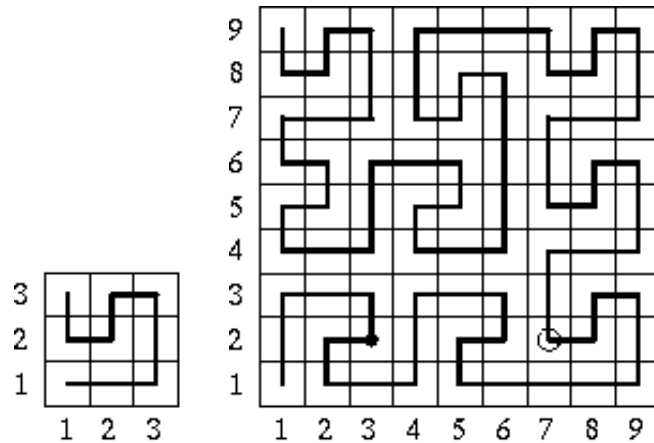
a: E->N W->S N->E S->W
b: E->S W->N N->W S->E
c: E->W W->E N->S S->N

E.g. $a(\text{SEN})=\text{WNE}$, $b(\text{SEN})=\text{ESW}$, $c(\text{SEN})=\text{NWS}$.

We start observing sleepwalker at the time he stands on the tile of coordinates (u_1, u_2) . After how many steps will sleepwalker fall into the hole made after removing the tile of coordinates (v_1, v_2) ?

Example

There are sleepwalker's paths on roofs of dimension $3*3$ and $9*9$ on the picture below. In the second case, the point at which the observation starts and the hole have been marked. The sleepwalker has exactly 20 steps to the hole (from the moment the observation starts).



Task

Write a program which:

- reads from the standard input integer k denoting the size of the roof ($3^k * 3^k$), the position of the sleepwalker at the moment the observation starts and the position of the hole,
- computes the number of steps that the sleepwalker will make before he falls into the hole,
- writes the result to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case one integer k , $1 \leq k \leq 60$, denoting the size of the roof ($3^k * 3^k$) is written. In each of the following two lines of the test case two natural numbers x, y separated with a space are written, $1 \leq x \leq 3^k$, $1 \leq y \leq 3^k$. The numbers in the second line are the coordinates of the tile the sleepwalker is standing on. The numbers in the third line are the coordinates of the hole. You may assume, that with these data the sleepwalker will eventually fall into the hole after some number of steps.

Output

The only line of output for each test case should contain the number of steps on the sleepwalker's path to the hole.

Example

Sample input:

```
1
2
3 2
7 2
```

Sample output

```
20
```

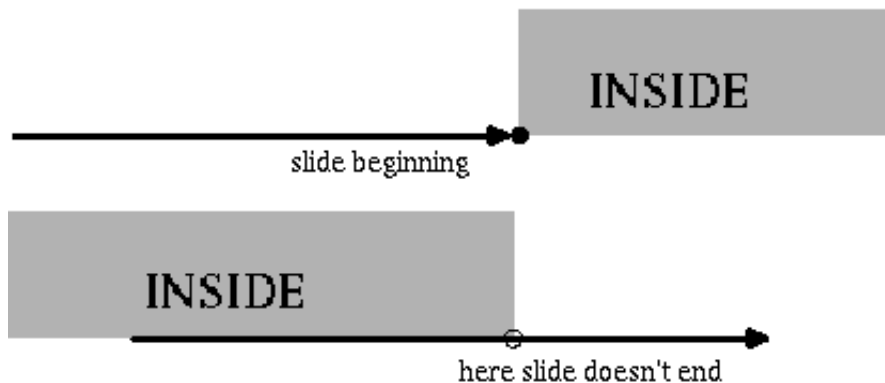
Added by: Piotr Lowiec
Date: 2004-09-13
Time limit [s]: 15
Source limit [B]:50000
Resource: 6th Polish Olympiad in Informatics, stage 2

SPOJ Problem Set

205. Icerink

Problem code: ICERINK

A skating competition was organized on the largest icerink in Byteland. The icerink is a square of size $10000 * 10000$. A competitor begins skating at the START point chosen by referees and his task is to finish sliding at the FINISH point, also chosen by referees. The points of START and FINISH are different. One can slide in directions parallel to the sides of the icerink. There are some obstacles placed on the icerink. Each obstacle is a prism, which base is a polygon with sides parallel to the sides of the icerink. Each two adjacent sides of the base are always perpendicular. The obstacles do not have common points. Each slide finishes up at the point where a competitor, for the first time, meets the wall of an obstacle, which is perpendicular to the direction of the slide. In other words, one can stop only when he crashes on a wall or in the FINISH point. Falling out of the icerink causes disqualification. Competitor may slide along walls of an obstacle.



Decide, whether a competitor who slides according to the given rules may reach the finish point, assuming he begun sliding from the starting point. If so, what is the minimal number of slides he needs to do?

Task

Write a program which:

- reads the description of the icerink, obstacles, and the coordinates of the start and finish point from the standard input,
- verifies, whether a competitor who begins from the starting point and slides according the rules may reach the finish point, and if so, computes the minimal number of slides he needs to do,
- writes the result in the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. We define a system of coordinates to describe positions of objects on a rink. The rink is a square with vertices $(0,0), (10000,0), (10000,10000), (0,10000)$. In the first line of each test case there are two

integers z_1 and z_2 separated by a single space, $0 \leq z_1, z_2 \leq 10000$. The pair (z_1, z_2) denotes coordinates of the START point. In the second line of the file there are two integers t_1 and t_2 separated by single space, $0 \leq t_1, t_2 \leq 10000$. The pair (t_1, t_2) denotes coordinates of the FINISH point. The third line of the file contains one integer s , $1 \leq s \leq 2500$. This is the number of obstacles. The following lines comprise descriptions of s obstacles. Each description of an obstacle begins with the line containing one positive integer r equal to the number of walls (sides of the base) of the obstacle. In each of the following r lines there are two integers x and y separated by a single space. These are the coordinates of the vertices of the obstacle's base, given in a clockwise order. (i.e. when going around the obstacle in this direction the inside is on the left-hand side). The total number of side walls of the obstacles does not exceed 10000.

Output

Your program should write for each test case:

- either one word 'NO' if it's impossible to get from the START point to the FINISH point
- or the minimal number of slides necessary to get to the FINISH point, if it is possible.

Example

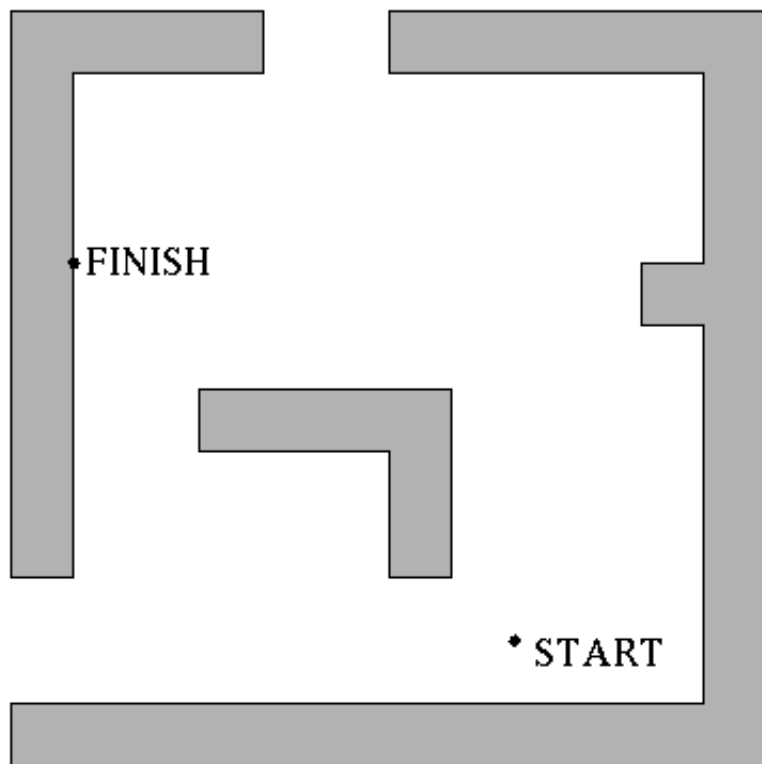
Sample input:

```
1
40 10
5 40
3
6
0 15
0 60
20 60
20 55
5 55
5 15
12
30 55
30 60
60 60
60 0
0 0
0 5
55 5
55 35
50 35
50 40
55 40
55 55
6
30 25
15 25
15 30
35 30
35 15
30 15
```

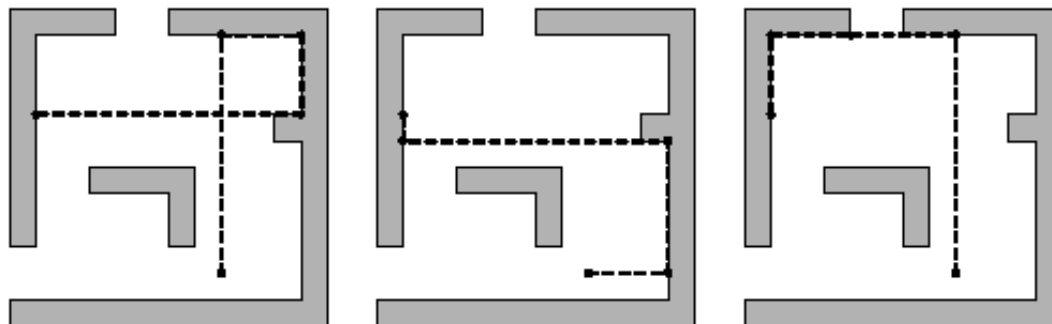
Sample output:

```
4
```

The sample input corresponds to the following situation:



These are the possible sequences of slides of length 4:



Warning: large Input/Output data, be careful with certain languages

Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]: 50000

Resource: 6th Polish Olympiad in Informatics, stage 2

SPOJ Problem Set

206. Bitmap

Problem code: BITMAP

There is given a rectangular bitmap of size $n*m$. Each pixel of the bitmap is either white or black, but at least one is white. The pixel in i -th line and j -th column is called the pixel (i,j) . The distance between two pixels $\mathbf{p}_1=(i_1,j_1)$ and $\mathbf{p}_2=(i_2,j_2)$ is defined as:

$$d(\mathbf{p}_1,\mathbf{p}_2)=|i_1-i_2|+|j_1-j_2|.$$

Task

Write a program which:

- reads the description of the bitmap from the standard input,
- for each pixel, computes the distance to the nearest white pixel,
- writes the results to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case there is a pair of integer numbers n, m separated by a single space, $1 \leq n \leq 182$, $1 \leq m \leq 182$. In each of the following n lines of the test case exactly one zero-one word of length m , the description of one line of the bitmap, is written. On the j -th position in the line $(i+1)$, $1 \leq i \leq n$, $1 \leq j \leq m$, is '1' if, and only if the pixel (i,j) is white.

Output

In the i -th line for each test case, $1 \leq i \leq n$, there should be written m integers $f(i,1), \dots, f(i,m)$ separated by single spaces, where $f(i,j)$ is the distance from the pixel (i,j) to the nearest white pixel.

Example

Sample input:

```
1
3 4
0001
0011
0110
```

Sample output:

```
3 2 1 0
2 1 0 0
1 0 0 1
```

Added by: Piotr Lowiec
Date: 2004-09-13
Time limit [s]: 8
Source limit [B]:50000
Resource: 6th Polish Olympiad in Informatics, stage 2

SPOJ Problem Set

207. Three-coloring of binary trees

Problem code: THREECOL

A **tree** consists of a node and some (zero, one or two) subtrees connected to it. These subtrees are called children.

A **specification** of the tree is a sequence of digits. If the number of children in the tree is:

- zero, then the specification is a sequence with only one element '0';
- one, the specification begins with '1' followed by the specification of the child;
- two, the specification begins with '2' followed by the specification of the first child, and then by the specification of the second child.

Each of the vertices in the tree must be painted either red or green or blue.

However, we need to obey the following rules:

- the vertex and its child cannot have the same color,
- if a vertex has two children, then they must have different colors.

How many vertices may be painted green?

Task

Write a program which:

- reads the specification of the tree from the standard input,
- computes the maximal and the minimal number of vertices that may be painted green,
- writes the results in the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. Each test case consists of one word (no longer than 10000 characters), which is a specification of a tree.

Output

Your program should write for each test case exactly two integers separated by a single space, which respectively denote the maximal and the minimal number of vertices that may be painted green.

Example

Sample input:

1

1122002010

Sample output:

5 2

Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]: 50000

Resource: 6th Polish Olympiad in Informatics, stage 3

SPOJ Problem Set

208. Store-keeper

Problem code: STORE

The floor of a store is a rectangle divided into $n*m$ square fields. Two fields are adjacent, if they have a common side. A parcel lays on one of the fields. Each of the remaining fields is either empty, or occupied by a case, which is too heavy to be moved by a store-keeper. The store-keeper has to shift the parcel from the starting field **P** to the final field **K**. He can move on the empty fields, going from the field on which he stands to a chosen adjacent field. When the store-keeper stays on a field adjacent to the one with the parcel he may push the parcel so that it moves to the next field (i.e. the field on the other side of the parcel), assuming condition that there are no cases on this field.

Task

Write a program, which:

- reads from the standard input a store scheme, a starting position of the store-keeper and a final position of the parcel,
- computes minimal number of the parcel shifts through borders of fields, which are necessary to put the parcel in the final position or decides that it is impossible to put the parcel there,
- writes the result into the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case two positive integers separated by a single space, $n, m \leq 100$, are written. These are dimensions of the store. In each of the following n lines there appears one m -letter word made of letters S, M, P, K, w. A letter on i -th position in j -th word denotes a type of the field with coordinates (i, j) and its meaning is following:

- S - case,
- M - starting position of the store-keeper,
- P - starting position of the parcel,
- K - final position of the parcel,
- w - empty field.

Each letter M, P and K appears in the test case exactly once.

Output

Your program should write to the standard output for each test case:

- exactly one word NO if the parcel cannot be put on the target field,
- exactly one integer, equal to the minimal number of the parcel shifts through borders of the fields, necessary to put a parcel on a final position, if it is possible to put the parcel there.

Example

Sample input:

```
1
10 12
SSSSSSSSSSSS
SwwwwwwwSSSS
SwSSSSwWSSSS
SwSSSSwWSKSS
SwSSSSwWSwSS
SwwwwWpwwwW
SSSSSSSwSwSw
SSSSSSMwSwww
SSSSSSSSSSSS
SSSSSSSSSSSS
```

Sample output

7

Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]:50000

Resource: 6th Polish Olympiad in Informatics, stage 3

SPOJ Problem Set

209. The Map

Problem code: MAP

After a new administrative division of Byteland cartographic office works on a new demographic map of the country. Because of technical reasons only a few colors can be used. The map should be colored so that regions with the same or similar population (number of inhabitants) have the same color. For a given color k let $A(k)$ be the number, such that:

- at least half of regions with color k has population not greater than $A(k)$
- at least half of regions with color k has population not less than $A(k)$

A **coloring error of a region** is an absolute value of the difference between $A(k)$ and the region's population. A **cumulative error** is a sum of coloring errors of all regions. We are looking for an optimal coloring of the map (the one with the minimal cumulative error).

Task

Write a program which:

- reads the population of regions in Byteland from the standard input,
- computes the minimal cumulative error,
- writes the result to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case an integer n is written, which is the number of regions in Byteland, $10 < n < 3000$. In the second line the number m denoting the number of colors used to color the map is written, $2 \leq m \leq 10$. In each of the following n lines there is one non-negative integer - a population of one of the regions of Byteland. No population exceeds 2^{30} .

Output

Your program should write for each test case one integer number equal to a minimal cumulative error, which can be achieved while the map is colored (optimally).

Example

Sample input:

```
1
11
3
21
14
6
18
10
2
```

15
12
3
2
2

Sample output:

15

Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]:50000

Resource: 6th Polish Olympiad in Informatics, stage 3

SPOJ Problem Set

210. The Altars

Problem code: ALTARS

According to Chinese folk believes evil spirits can move only on a straight line. It is of a great importance when temples are built. The temples are constructed on rectangular planes with sides parallel to the north - south or east - west directions. No two of the rectangles have common points. An entrance is situated in the middle of one of four walls and its width is equal to the half of the length of the wall. An altar appears in the center of the temple, where diagonals of the rectangle intersect. If an evil spirit appears in this point, a temple will be profaned. It may happen only if there exists a ray which runs from an altar, through an entrance to infinity and neither intersects nor touches walls of any temple (on a plane parallel to the plane of a construction area), i.e. one can draw at a construction area a line which starts at the altar and runs to the infinity without touching any wall.

Task

Write a program which:

- reads descriptions of the temples from the standard input,
- verifies which temples could be profaned,
- writes their numbers to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case one integer n , the number of temples $1 \leq n \leq 1000$, is written.

In each of the following n lines there is a description of one temple (in i -th line a description of the i -th temple). The description of a temple consists of four non-negative integers, not greater than 8000 and a letter E, W, S or N. Two first numbers are coordinates of a temple's northern-west corner and two following are coordinates of an opposite southern-east corner. In order to specify coordinates of a point first we give its geographical longitude, which increases from the west to the east, and then its latitude, which increases from the south to the north. The fifth element of the description indicates the wall with the entrance (E - Eastern, W - Western, S - Southern, N - Northern). The elements of the temple's description are separated by single spaces.

Output

In the following lines of the output for each test case your program should write in ascending order numbers of the temples, which may be profaned by an evil spirit. Each number is placed in a separate line. If there are no such numbers, you should write one word: NONE.

Example

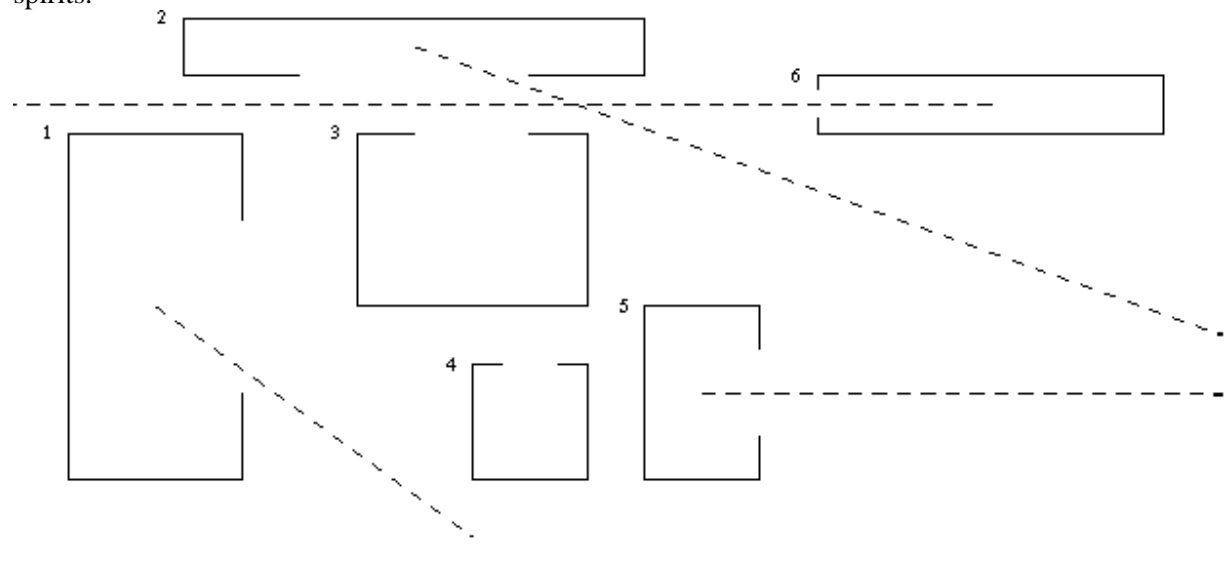
Sample input

```
6
1 7 4 1 E
3 9 11 8 S
6 7 10 4 N
8 3 10 1 N
11 4 13 1 E
14 8 20 7 W
```

Sample output

```
1
2
5
6
```

The picture shows the temples described in the example. The dashed lines show possible routes of evil spirits.



Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]: 50000

Resource: 6th Polish Olympiad in Informatics, stage 3

SPOJ Problem Set

211. Primitivus recurencis

Problem code: PRIMIT

A **genetic code** of the abstract primitivus (*Primitivus recurencis*) is a series of natural numbers $K=(a_1,...,a_n)$. A **feature** of primitivus we call each ordered pair of numbers (l,r) , which appears successively in the genetic code, i.e. there exists such i that $l=a_i$, $r=a_{i+1}$. There are no (p,p) features in a primitivus' genetic code.

Task

Write a program which:

- reads the list of the features from the standard input,
- computes the length of the shortest genetic code having given features,
- writes the results to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case one positive integer number n is written. It is the number of different features of the primitivus. In each of the following n lines there is a pair of natural numbers l and r separated by a single space, $1 \leq l \leq 1000$, $1 \leq r \leq 1000$. A pair (l, r) is one of the primitivus' features. The features do not repeat in the input file

Output

Your program should write for each test case exactly one integer number equal to the length of the shortest genetic code of the primitivus, comprising the features from the input.

Example

Sample input:

```
1
12
2 3
3 9
9 6
8 5
5 7
7 6
4 5
5 1
1 4
4 2
2 8
8 6
```

Sample output:

```
15
```

All the features from the example are written in the following genetic code:
(8, 5, 1, 4, 2, 3, 9, 6, 4, 5, 7, 6, 2, 8, 6)

Warning: enormous Input/Output data, be careful with certain languages

Added by: Piotr Lowiec

Date: 2004-09-13

Time limit [s]: 15

Source limit [B]:50000

Resource: 6th Polish Olympiad in Informatics, stage 3

SPOJ Problem Set

212. Water among Cubes

Problem code: WATER

On a rectangular mesh comprising $n*m$ fields, $n*m$ cuboids were put, one cuboid on each field. A base of each cuboid covers one field and its surface equals to one square inch. Cuboids on adjacent fields adhere one to another so close that there are no gaps between them. A heavy rain pelted on a construction so that in some areas puddles of water appeared.

Task

Write a program which:

- reads from the standard input a size of the chessboard and heights of cuboids put on the fields,
- computes maximal water volume, which may gather in puddles after the rain,
- writes results in the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case two positive integers $1 \leq n \leq 100$, $1 \leq m \leq 100$ are written. They are the size of the mesh. In each of the following n lines there are m integers from the interval $[1..10000]$; i -th number in j -th line denotes a height of a cuboid given in inches put on the field in the i -th column and j -th row of the chessboard.

Output

Your program should write for each test case one integer equal to the maximal volume of water (given in cubic inches), which may gather in puddles on the construction.

Example

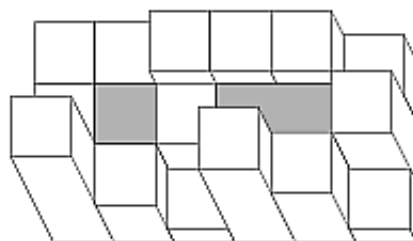
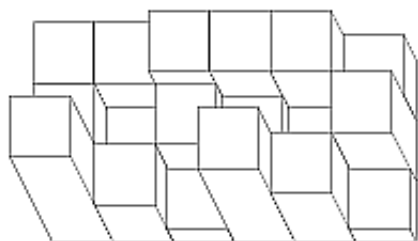
Sample input:

```
1
3 6
3 3 4 4 4 2
3 1 3 2 1 4
7 3 1 6 4 1
```

Sample output:

```
5
```

The picture below shows the mesh after the rain (seen from above). Puddles are drawn in gray.



Added by: Piotr Lowiec
Date: 2004-09-13
Time limit [s]: 15
Source limit [B]:50000
Resource: 6th Polish Olympiad in Informatics, stage 3

SPOJ Problem Set

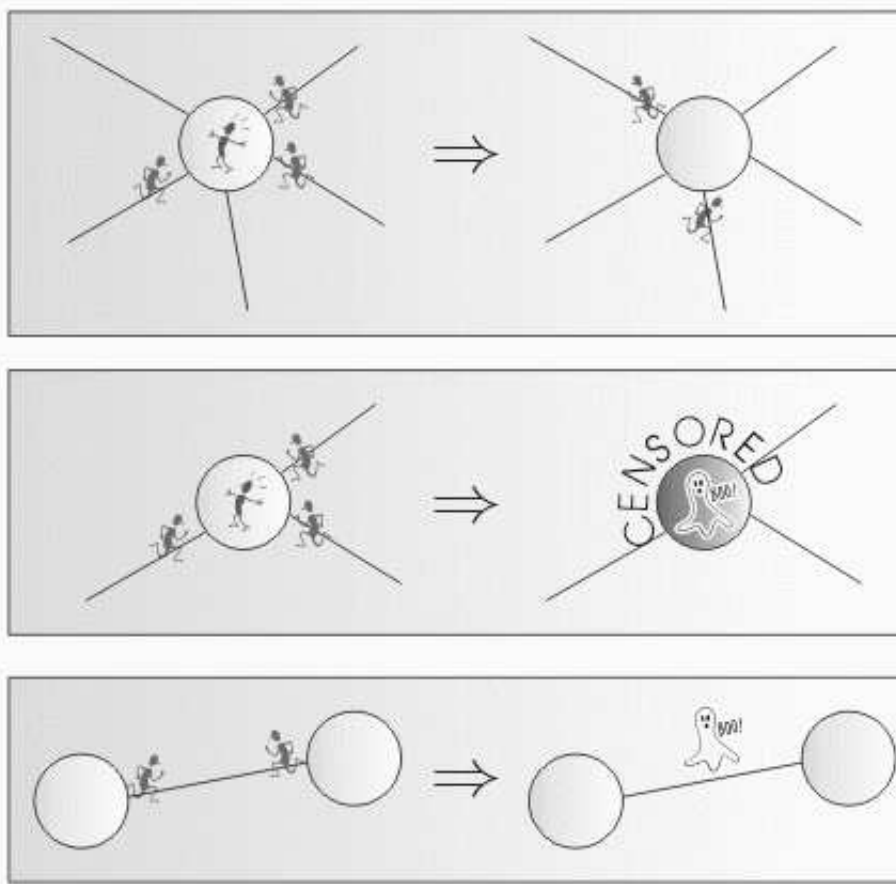
215. Panic in the Plazas

Problem code: PANIC

Have you ever heard of the BBFO? The Bytelandian Bit-eating Fanatic Organisation regards itself as a collection of people with slightly unorthodox views on law and order in the world, and is regarded by others as the most wildly dangerous and unpredictable terrorist organisation which afflicts the small and otherwise peaceful country of Byteland.

Intelligence reports claim that the next act of violence to be performed by the BBFO is a widescale, distributed bomb attack in the Bytelandian capital. Therefore, all precautions have been undertaken to prevent any such action. The BBFO, seeing the futility of their original scheme, decided to change the plan of action. The new idea is endowed with devilish simplicity.

The capital of Byteland is a network of plazas, some of which (but not necessarily all) are connected by bidirectional streets of different length. Crowds of people are sitting at all the plazas, sipping coffee and generally relaxing. The terrorists plan to creep up to some of the plazas armed with inflatable paper bags. Then, exactly at midday, all the bags will be burst in such a way as to simulate the bang of a bomb. Panic will ensue at the plazas where the bags were burst, and will spread throughout some of the city. Panic breaks out at a plaza the moment a bag explodes in it, or immediately after a panicking crowd rushes into the plaza from at least one of the side streets. The people in the plaza then split up into crowds, which rush out by all possible streets except those by which people have just run in. After entering a street, a crowd runs along it at constant speed until it reaches the plaza at the other end, causing panic there, etc. If there is no possible way of escape from a plaza, everybody in it perishes. Similarly, if two crowds rushing in opposite directions collide in mid-street, all the people are lethally trampled.



Despite the panic, people in the city retain a little free will. They don't move at all until the panic reaches them, but when they have to escape, they can always choose the escape route from a plaza that suits them best. Assuming you were to sit in one of the plazas of Byteland at noon that fateful day... which plaza would you choose to sit in? All your normal preferences concerning the quality of coffee in the cafes are temporarily forgotten, and your only aim is to survive as long as possible.

Input

The first line of input contains a single integer $t \leq 500$, the number of test cases. t test cases follow. Each test case begins with a line containing three integers n m k ($1 \leq n \leq 50000$, $0 \leq m \leq 250000$, $0 \leq k \leq n$) denoting the total number of plazas, the number of streets in the city, and the number of plazas in which bags are planted, respectively. Each of the following m lines contains 4 integers u v t_{uv} t_{vu} ($1 \leq u, v \leq n$, $1 \leq t_{uv}, t_{vu} \leq 1000$) representing a single road in the city - leading from plaza u to plaza v and requiring t_{uv} time to cover when running at constant speed from u to v , and t_{vu} time when running the other way. The last line of a test case description contains a list of the k numbers of plazas at which bags explode at noon.

Output

For each test case, the output should contain a single line with a space separated increasing sequence of integers - the numbers of all the plazas which offer the maximum possible survival time to a person sitting there at noon.

Example

Input:

```
2

4 5 2
1 2 10 10
2 4 30 30
3 2 10 10
4 3 50 5
3 1 5 50
1 2
```

```
2 0 1
2
```

Output:

```
2 3 4
1
```

(In the first case the life expectancy is 22.5, in the second case it is more or less infinite.)

Warning: enormous Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-09-27

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004 (problemset 2)

SPOJ Problem Set

217. Soldiers on Parade

Problem code: SOPARADE

Protocol is really weird in Byteland. For instance, it is required that, when presenting arms before an officer, soldiers should stand in a single row (at positions numbered from 1 to n). Soldiers may have one of 4 possible ranks, distinguished by the number of squiggles on the epaulets (between 1 and 4). Soldiers standing beside each other must have a difference in rank of at least two squiggles. Moreover, there are additional sets of rules (different for every province). Each rule states that soldiers standing at some given positions of the row must differ in rank by at least a squiggle.

Starting from the new year onwards, some provinces are changing their set of protocol rules. As the Senior Military Secretary of Protocol, it is your task to approve the new rules. To your surprise, some of the provinces have put forward protocol rules which are quite impossible to fulfill, even if the soldiers were to be specially selected for the purpose of presenting arms. Detect all such offending provinces and on no account approve their laws.

Input

The first line of input contains a single positive integer $t \leq 10$ - the number of provinces which are proposing new laws. t sets of rules follow, separated by empty lines.

Each set of rule begins with a line containing two non-negative integers n p ($n \leq 100000$, $p \leq 100000$) - the number of soldiers arranged and the number of rules proposed in the province, respectively. Each of the next p lines contains a single rule: an integer b_i ($2 \leq b_i \leq n$), followed by b_i integers a_1, a_2, \dots, a_{b_i} ($1 \leq a_k \leq n$). Such a rule means that soldiers standing at positions a_1, a_2, \dots, a_{b_i} must all be of different rank.

Output

For every set of rules presented at input, output a single line containing the word *rejected* if no unit of soldiers can be arranged in accordance with protocol, or the word *approved* in the opposite case.

Example

Input :

2

2 1

2 1 2

5 2

3 1 3 2

4 2 3 4 5

Output :

approved

rejected

Added by: Adrian Kosowski
Date: 2004-10-08
Time limit [s]: 20
Source limit [B]:50000
Resource: DASM Programming League 2004 (problemset 1)

SPOJ Problem Set

218. Points on a Sphere

Problem code: PSPHERE

Imagine a number of identically charged weightless dimensionless particles placed on the surface of a ball. They will instantly reach a state of equilibrium (a stable state of minimum energy), becoming distributed fairly evenly all round the sphere.

You probably won't be surprised to hear that Byteland has a sadly distorted electrostatic field, and the energy of the system is not governed by ordinary laws. Instead, it is inversely proportional to the distance between the closest pair of charges on the sphere.

Please help the charges find positions in which they will feel as comfortable as possible. Charges should be regarded as points in 3D space, located on the surface of the unitary sphere (with center (0,0,0) and a radius of 1).

Input

An integer t denoting the number of test cases ($t \leq 10$), followed by t test cases, each consisting of a line with a single integer n - the number of points on the sphere ($2 \leq n \leq 1000$).

Output

For each test case, output n lines consisting of three floating point numbers, corresponding to the x y z coordinates of successive points.

Scoring

The score of your program is the total of scores awarded for individual test cases.

For each test case you will receive $n \cdot d$ points, where d denotes the minimum distance between the closest pair of points in the solution output by your program. If you place some of the points further than 10^{-5} from the surface of the sphere, your solution will be regarded as incorrect.

Example

For the sample input:

1

2

a program outputting:

0.0 0.0 1.0

0.0 1.0 0.0

will receive 2.828 points.

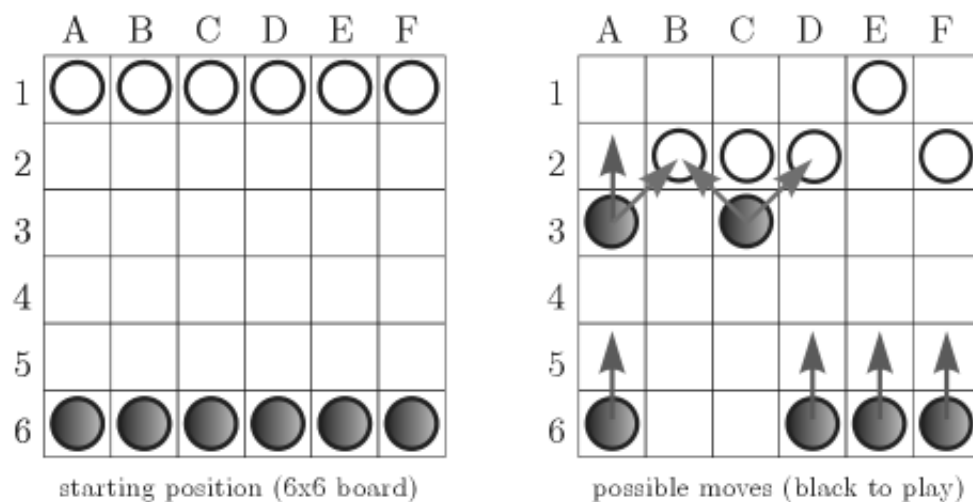
Added by: Adrian Kosowski
Date: 2004-10-09
Time limit [s]: 10
Source limit [B]:50000
Resource: DASM Programming League 2004 (problemset 1)

SPOJ Problem Set

219. Pawns Gone Wild

Problem code: PAWNS

Imagine a game played on an $n \times n$ chessboard by two players sitting at opposite ends, one having n white pawns, the other - n black pawns. Pawns are arranged in the row closest to the player. Moves are made in turn by both players and resemble those in chess: in a single move, a player can move exactly one pawn, either a square forward (if the square it is moving onto is free), or on the bias, one square forward and one square to the left or right (if the square it is moving onto is occupied by an enemy's pawn, which is considered beaten and removed from the game).



Pawns may never be moved backwards or off the board, and if a pawn reaches the final line it just has to stay there. The game ends if a player can't make a move. The winner is then the player who... oh, it doesn't matter really (possibly the players have a fight with beer bottles, and the one who isn't knocked out, wins). Your task is different - seeing snapshots of a game at two moments of time, try to reenact a sequence of moves that may have led from the first situation to the second.

Input

The first line of input contains a single positive integer $t \leq 25$, the number of test cases. t test cases (of successively increasing size) follow.

Each test case begins with an integer n ($2 \leq n \leq 26$) denoting the size of the board. Then, the snapshot of the earlier situation is given, followed by a snapshot of the later situation. Each snapshot is a sequence of n lines of n characters, corresponding to the squares of a chessboard oriented as in the figure above. Character '.' - denotes an empty square, 'W' - a square with a white pawn, 'B' - a square with a black pawn. Assume that it is black's turn to move after the earlier position (though black needn't have necessarily started the game as such).

Output

For each test case, output the number k of moves which could have led from the first to the second position (output 0 if you don't know a solution, even though such a solution exists for certain). In the next k lines, print the determined sequence of moves. Each move should be given in a separate line, using the format: `old_column old_row new_column` to describe the change of coordinates of a pawn (assume board orientation as in the exemplary figures).

Scoring

The score of your program is the total of scores awarded for individual test cases. For each test case for which you find a solution in k moves you will receive k points.

Example

For the sample input:

```
1
6

...W.
.WWW.W
B.B...
.....
.....
B..BBB

.....
.BWW..
.....W
.....
.....B
B..BB.
```

a program outputting:

```
5
A 3 B
F 2 F
C 3 D
E 1 D
F 6 F
```

will receive 5 points.

Added by: Adrian Kosowski
Date: 2004-10-10
Time limit [s]: 50
Source limit [B]: 50000
Resource: DASM Programming League 2004 (problemset 1)

SPOJ Problem Set

220. Relevant Phrases of Annihilation

Problem code: PHRASES

You are the King of Byteland. Your agents have just intercepted a batch of encrypted enemy messages concerning the date of the planned attack on your island. You immediately send for the Bytelandian Cryptographer, but he is currently busy eating popcorn and claims that he may only decrypt the most important part of the text (since the rest would be a waste of his time). You decide to select the fragment of the text which the enemy has strongly emphasised, evidently regarding it as the most important. So, you are looking for a fragment of text which appears in all the messages disjointly at least twice. Since you are not overfond of the cryptographer, try to make this fragment as long as possible.

Input

The first line of input contains a single positive integer $t \leq 10$, the number of test cases. t test cases follow. Each test case begins with integer n ($n \leq 10$), the number of messages. The next n lines contain the messages, consisting only of between 2 and 10000 characters 'a'-'z', possibly with some additional trailing white space which should be ignored.

Output

For each test case output the length of longest string which appears disjointly at least twice in all of the messages.

Example

Input :

```
1
4
abbabba
dabddkababa
bacaba
baba
```

Output :

```
2
```

(in the example above, the longest substring which fulfills the requirements is 'ba')

Added by: Adrian Kosowski
Date: 2004-10-11
Time limit [s]: 20
Source limit [B]: 50000
Resource: DASM Programming League 2004 (problemset 1)

SPOJ Problem Set

222. The Burning City

Problem code: BURNCITY

Terrorists from the BBFO have raised fires in the capital of Byteland! As is it is a hot summer day, most of the fire brigade have quite naturally taken a day off, and so the noble task of extinguishing all the fires falls to the only officer on duty. By now you will probably not be surprised to learn that he is in fact... Johnny. This enterprising youth remains undaunted by the challenge facing him, and, taking advantage of the absence of his superiors, he decides to use his favourite fire fighting technique. So, he loads the fire station's helicopter with as many dynamite charges as it can carry, and takes off on his errand of mercy.

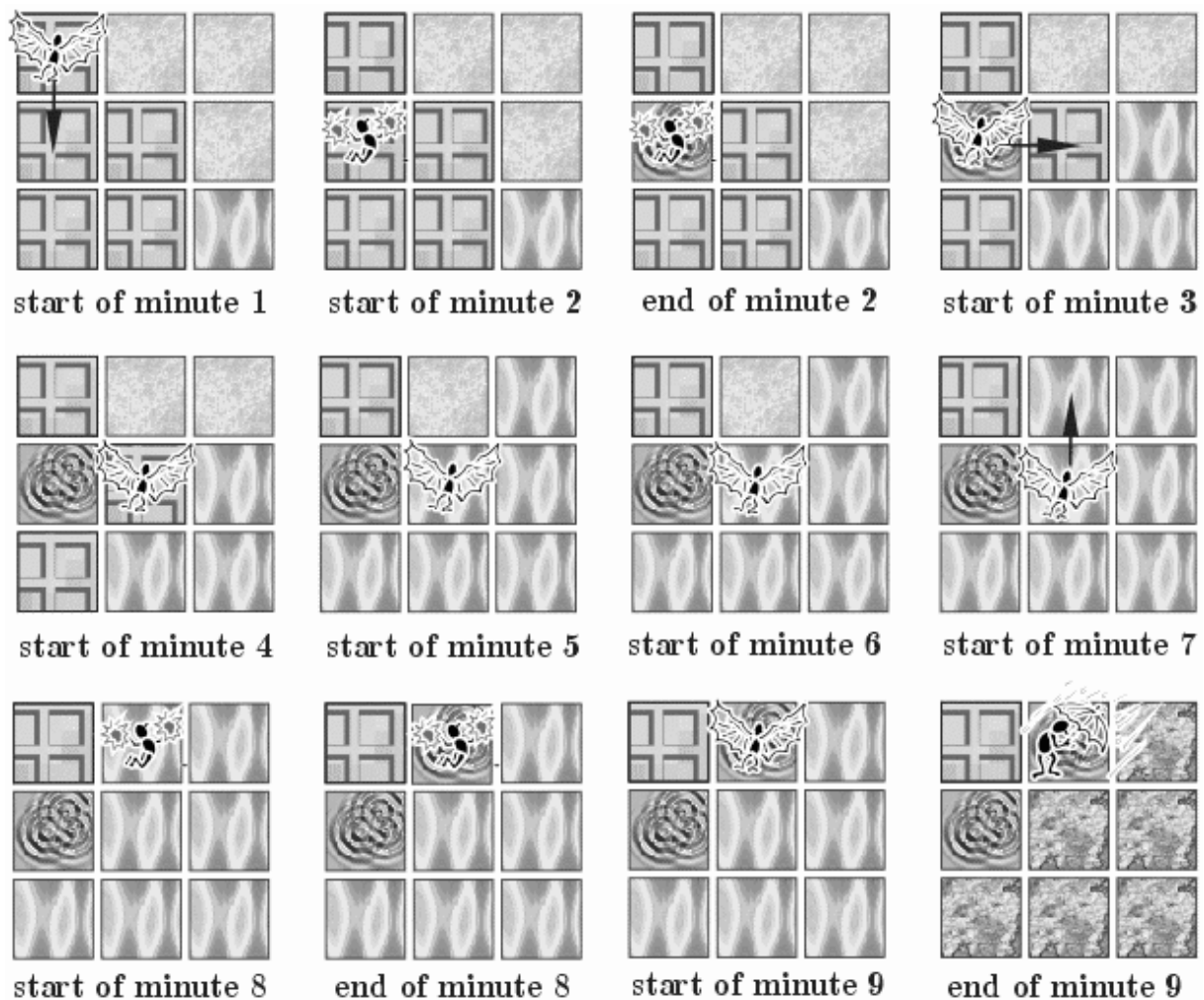
From up there in the sky Johnny can see the city as a square, sliced into smaller, identical squares by a regular grid of streets. Every square contains one of three kinds of terrain - buildings, grassland or water (perhaps most firemen would go into further detail when analysing terrain, but you really can't expect that from a firefighter whose preferred method of extinguishing fires is dynamite, can you?).

Johnny starts out in the centre of the square corresponding to the fire station. In the time from the start of a minute to the end of that minute he can move to the center of one of the four adjacent squares (but he is not allowed to leave the city). While over the center of a square he can choose to drop a single dynamite charge on it. He starts preparing the charge at the beginning of a minute, and it is dropped from the helicopter at the end of the same minute. Everything on the square on which the bomb was dropped is blown apart, and in its place a crater is formed and instantly flooded by subterranean waters.

The fire spreads in a most predictable way: if a square starts burning at the beginning of minute m , then all four adjacent squares will catch fire at the start of minute $(m+2)$. The only exception is a square filled with water (either naturally, or by Johnny's bombs) which never catches fire. If a square starts burning, all property on it is instantly destroyed. Once a square starts burning it will only stop burning if Johnny blows it up, or when the monsoon rain comes and floods the city, at the end of the h -th minute of firefighting.

Johnny's main objective is to save as many squares with buildings as possible (from fire and dynamite).

An example of the fire fighting process is presented below.



Input

The first line of input contains a single integer $t \leq 500$, the number of test cases.

The first line of every test case contains five integers $n \ c \ h \ s_x \ s_y$, respectively denoting: n - the length of one side of the city (measured in squares), c - the number of dynamite charges Johnny can use, h - the number of minutes after which the rain falls, s_x, s_y - the x and y coordinates of the square containing the fire-station from which Johnny starts, measured relative to the North-West corner of the city ($1 \leq s_x, s_y \leq n \leq 50$, $0 \leq c \leq h \leq 5 * n$; there are about 10 test cases for all possible values of n). Finally, the map of the city is given in the form of n lines of n characters each, each corresponding to the state of a square at the start of the fire fighting ('b' - building, 'g' - grassland, 'w' - water, 'f' - fire).

Output

For the i -th test case output a line containing the text 'city i Y' if you want to solve the test case or 'city i N' if you wish to leave it out.

If you chose to solve the test case, in the next line output a sequence of exactly h characters 'N', 'S', 'W', 'E', '+' or '-', corresponding to Johnny's actions in successive minutes (moving North, South, West and East on the map, dropping dynamite, not doing anything, respectively).

Score

The total score is the total number of rescued squares with buildings taken over all test cases.

Example

Input:

```
5
3 2 9 1 1
bgg
bbg
bbf
4 2 8 3 1
bbbb
bgwg
fwgg
gbbb
4 3 15 2 1
bbbb
bbbb
bbbb
fbbb
4 3 15 2 1
bbbf
bbbb
bbbb
fbbb
4 3 15 2 1
bbbf
bbbb
bbbb
fbbb
```

Output:

```
city 1 Y
S+E---N+-
city 2 Y
W+SSS+--
city 3 Y
ESE+SW+S+-----
city 4 Y
+EES+W+-----
city 5 Y
+ES+-E+-----
```

Score:

```
9
```

(The first test case is illustrated in the figure and Johnny can save one building. In testcases 2, 3, 4, 5 Johnny saves 4, 2, 2 and 0 buildings, respectively).

Bonus info: The three digit number after the decimal point of your score denotes the number of test cases you have solved correctly, rescuing at least one building.

Warning: large Input/Output data, be careful with certain languages

Added by: Michal Malafiejski

Date: 2004-10-11

Time limit [s]: 42

Source limit [B]:50000

Resource: DASM Programming League 2004 (problemset 3)
English version and pictures by Adrian Kosowski

SPOJ Problem Set

224. Vonny and her dominos

Problem code: VONNY

Vonny loves playing with dominos. And so she owns a standard set of dominos. A standard set of dominos consists of 28 pieces called bones, tiles or stones. Each bone is a rectangular tile with a line dividing its face into two square ends. Each square is labeled with a number between 0 and 6. The 28 stones are labeled (0,0),(0,1),(0,2),(0,3),(0,4),(0,5),(0,6), (1,1),(1,2),...,(5,5),(5,6),(6,6). Tommy - the brother of Vonny - build a box for Vonny's dominos. This box is sized 7 x 8 squares. Every square is labeled with a number between 0 and 6. You can see a example box here.

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

Now Vonny wants to arrange her 28 stones in such way that her stones cover all squares of the box. A stone can only be placed on two adjacent squares if the numbers of the squares and of the domino stone are equal. Tommy asks Vonny in how many different ways she can arrange the dominos. Tommy assumes that Vonny need a lot of time to answer the question. And so he can take some of Vonny's candies while she solves the task. But Vonny is a smart and clever girl. She asks you to solve the task and keeps an eye on her candies.

Input

The first line of the input contains the number of testcases. Each case consists of 56 numbers (7 rows and 8 cols) between 0 and 6 which represents Tommy's box.

Output

For each testcase output a single line with the number which answers Tommy's question.

Example

Input :

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

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

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

Output:

```
18
1
```

Added by: Simon Gog
Date: 2004-10-18
Time limit [s]: 10
Source limit [B]:50000

SPOJ Problem Set

225. Nightmare in the Towers of Hanoi

Problem code: HANOI

Consider the following variation of the well known problem Towers of Hanoi:

We are given n towers and m disks of sizes $1, 2, 3, \dots, m$ stacked on some towers. Your objective is to transfer all the disks to the k -th tower in as few moves as you can manage, but taking into account the following rules:

- moving only one disk at a time,
- never moving a larger disk one onto a smaller one,
- moving only between towers at distance at most d .

You can assume that all the problems can be solved in not more than 20000 moves.

Input

The first line of input contains a single positive integer $t \leq 1000$, the number of test cases.

Each test case begins with the number of towers $3 \leq n \leq 100$, the number of target tower $1 \leq k \leq n$, the number of disks $m \leq 100$ and the maximum distance $1 \leq d \leq n - 1$.

Then, the following m lines consist of pairs of numbers describing the initial situation, in the form: the tower and disk on it. Assume according to the rules that on every tower smaller disks are on larger disks.

Output

Process all test cases. The correct output for the i -th test case takes the following form:

i [the number of the test case (in input order)]

$a\ b$ [a sequence of lines of this form, where a is the tower with the moved disk on top of it and b is the target tower].

The test case is considered solved if after performing the sequence all disks are on the k -th tower. At the end of the series of moves you should always write a line consisting of two zeros ('0 0').

Scoring

The score awarded to your program is the sum of scores for individual test cases. For the i -th test case you will receive $T_i / (T_i + A_i)$ points, where $T_i \leq 20000$ and A_i is the number of moves in your solution. If you don't want to solve a test case, you may output the line '0 0' without a list of moves, for which you will not be awarded any points. Your program may not write more than 30000 kB to output (this will result in SIGXFSZ).

Example

Input

```
5
3 3 3 2
1 1
1 2
1 3
3 1 3 2
1 1
1 2
1 3
4 4 4 2
1 1
1 2
1 3
1 4
4 4 4 2
1 1
1 2
2 4
4 3
4 4 4 3
1 1
4 2
4 3
4 4
```

Output

```
1
1 3
1 2
3 2
1 3
2 1
2 3
1 3
0 0
2
0 0
3
0 0
4
4 3
2 4
3 4
1 2
1 3
3 4
2 4
0 0
5
1 2
0 0
```

Score

Assuming: $T = \{7,6,15,7,1\}$ the output will receive **2** points.

Bonus info: If score = *xxx.xxxaaa*, *aaa* means the number of test cases with non-zero score...

Added by: Michal Malafiejski

Date: 2004-10-27

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004 (problemset 2)

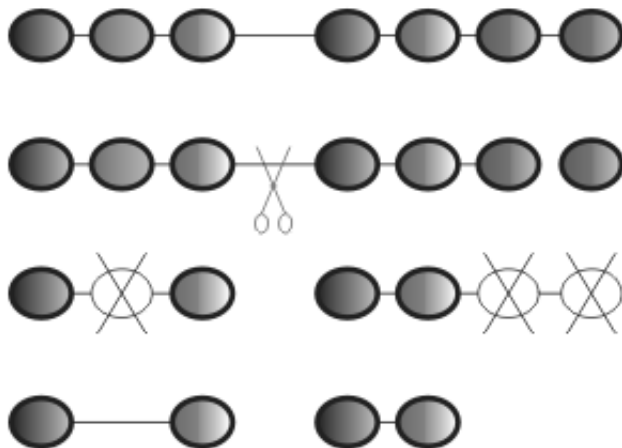
SPOJ Problem Set

226. Jewelry and Fashion

Problem code: JEWELS

You work for a small jewelers' company, renowned for the exquisite necklaces and multi-colored amber strings it produces. For the last three centuries, the sales of strings alone have been enough to keep business going without a hitch. Now however, the influence of fashion is greater than ever, and you face the prospect of imminent bankruptcy unless you adapt to the needs and fancies of the rather unusual part of society who constitute your main clientele. These elderly ladies have recently decided that fashion has changed: strings are out, and earrings are in. There is nothing to be done about it -- you have to comply and switch to the production of earrings.

One problem remains: what to do with the impressive heap of amber strings piled up in your shop? One of your assistants has a bright idea: he recommends cutting the strings into two parts, removing some stones to make both parts have an identical color pattern (either immediately, or after rotation by 180 degrees), and selling what remains as pairs of earrings. After a moment's thought, you decide to go ahead with the plan. But your careful managerial eye tells you that minimising the number of wasted (removed) stones may not be as easy as it sounds...



two earrings of length 2 are created
from one string of length 7

Input

The first line of input contains a single integer $t \leq 500$, the number of test cases. The next t lines contain one test case each, in the form of a string of at most 8000 characters 'a'-'z' (terminated by a new line, optionally preceded by whitespace which should be ignored). The i -th character of the line corresponds to the design on the i -th stone in the amber string it represents. The total length of the input file is not more than 100kB.

Output

For each test case output two numbers: the largest possible total length of the pair of earrings which can be produced from the string, and a positive integer denoting the number of the stone after which the string ought to be cut so as to achieve this. If more than one cutting position is possible, output the leftmost (smallest) one.

Example

Input :

```
3
abcacdd
acbddabedff
abcbca
```

Output :

```
4 3
6 4
4 2
```

(the first case is illustrated in the figure, in the second case we produce a pair of earrings of the form 'abd', in the third - a pair of earrings which look like 'ab' after rotating the second one by 180 degrees).

Added by: Adrian Kosowski

Date: 2004-10-29

Time limit [s]: 42

Source limit [B]:50000

Resource: DASM Programming League 2004, problemset 2

SPOJ Problem Set

227. Ordering the Soldiers

Problem code: ORDERS

As you are probably well aware, in Byteland it is always the military officer's main worry to order his soldiers on parade correctly. In Bitland ordering soldiers is not really such a problem. If a platoon consists of n men, all of them have different rank (from 1 - lowest to n - highest) and on parade they should be lined up from left to right in increasing order of rank.

Sounds simple, doesn't it? Well, Msgt Johnny thought the same, until one day he was faced with a new command. He soon discovered that his elite commandos preferred to do the fighting, and leave the thinking to their superiors. So, when at the first rollcall the soldiers lined up in fairly random order it was not because of their lack of discipline, but simply because they couldn't work out how to form a line in correct order of ranks. Msgt Johnny was not at all amused, particularly as he soon found that none of the soldiers even remembered his own rank. Over the years of service every soldier had only learned which of the other soldiers were his superiors. But Msgt Johnny was not a man to give up easily when faced with a true military challenge. After a moment's thought a solution of brilliant simplicity struck him and he issued the following order: "men, starting from the left, one by one, do: (step forward; go left until there is no superior to the left of you; get back in line)". This did indeed get the men sorted in a few minutes. The problem was solved... for the time being.

The next day, the soldiers came in exactly the same order as the day before, and had to be rearranged using the same method. History repeated. After some weeks, Msgt Johnny managed to force each of his soldiers to remember how many men he passed when going left, and thus make the sorting process even faster.

If you know how many positions each man has to walk to the left, can you try to find out what order of ranks the soldiers initially line up in?

Input

The first line of input contains an integer $t \leq 50$, the number of test cases. It is followed by t test cases, each consisting of 2 lines. The first line contains a single integer n ($1 \leq n \leq 200000$). The second line contains n space separated integers w_i , denoting how far the i -th soldier in line must walk to the left when applying Msgt Johnny's algorithm.

Output

For each test case, output a single line consisting of n space separated integers - the ranks of the soldiers, given from left to right in their initial arrangement.

Example

Input:

```
2
3
0 1 0
5
0 1 2 0 1
```

Output:

```
2 1 3
3 2 1 5 4
```

Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-10-30

Time limit [s]: 30

Source limit [B]: 50000

Resource: DASM Programming League 2004, problemset 2

SPOJ Problem Set

228. Shamans

Problem code: SHAMAN

In the far bare land there lives a mysterious tribe. They suffer from drought every year but they stick to their faith in god that they will never leave their home land. To counter the dry weather the shamans in the tribe must pray during the hard time and hope the blessed rain will aid their production of food.

There are 4 chief shamans in the tribe and each of them will choose a summit in the territory to proceed with his praying. The area in which the shamans' spells take effect will be the quadrangle they form, each of them being one of its vertices (which the god will see when he looks down from the high heavens). The land is quite full of pinch and punch and the tribe has selected quite a few peaks for the shamans to pray on. Of course the area of the quadrangle is expected to be as large as possible so before the shamans actually go out, they will have to choose the 4 peaks that best suit their purpose.

Input

One integer in the first line, stating the number of test cases, followed by a blank line. There will be not more than 80 tests.

For each test case, the first line is an integer n ($4 \leq n \leq 2000$) stating the number of peaks. Then n lines follow, each presenting the position of a peak, with two integers x, y ($-20000 \leq x, y \leq 20000$).

The test cases will be separated by a single blank line.

Output

A floating point number with exactly 1 digit precision: the maximum area the shamans can cover.

Example

Input :

2

4

0 0

1 0

1 1

0 1

4

0 0

0 1

1 1

1 0

Output :

1.0

1.0

Added by: Neal Zane
Date: 2004-11-02
Time limit [s]: 6
Source limit [B]:50000
Resource: Neal Zane

SPOJ Problem Set

229. Sorting is easy

Problem code: SORTING

Do you think sorting is easy?
try your luck
in brainfuck

For those who don't know that brainfuck is a programming language: Take a look at the converter to C. It will ignore every unknown command, therefore submitting a program in any other language won't necessarily lead to compile error, but certainly not to Accepted.

Input

The input consists of a line of up to 1000 uppercase letters, terminated with a '\n' character (ASCII value 10).

Output

The output should contain a line consisting of the same characters as the input line, but in non-descending order.

Example

Input :

BRAINFUCK

Output :

ABCFIKNRU

Added by: Adrian Kuegel

Date: 2004-11-04

Time limit [s]: 1

Source limit [B]:500

SPOJ Problem Set

232. Bytelandian Telecom

Problem code: BYTELE

King Johnny has a serious drink problem, which has recently become the focus of attention of all Bytelandian tabloids and colour magazines. In a desperate effort to divert the public's attention and ingratiate himself with his subjects, he decides to start giving out valuable gifts. This time he has chosen to harass the peaceful life of the CEO of Bytelandian Telecom, and requested him to create a Metropolitan Area Network for the citizens of the capital of Byteland, as part of an "our King is a good man" campaign.

The CEO has no choice but to obey the orders he receives. This rational and business-minded man would obviously like to perform the installation at the smallest possible cost, and he asks you for your help.

The King has stated the topology of the network plainly enough in the form of a graph (not necessarily connected), with vertices corresponding to nodes (computers), and edges to the cable connections between them. It is now your task to select the points of the city to place the nodes of the network at. The city is a regular mesh of streets (depicted as vertical and horizontal segments on a map), with crossroads located at points with integer coordinates. Nodes may only be located at crossroads of streets (no two nodes at the same crossroad). Cables may only run along streets and must connect nodes by the shortest possible route under this constraint. Moreover, a cable of precisely such length must be currently in stock (you are provided with a list of possible cable lengths). Try to layout the network in such a way as to minimise the total length of cable used.

Input

The input starts with a line containing integer $t \leq 1000$, the number of test cases. t test cases follow.

The first line of a test case begins with integer k , denoting the number of different available cable lengths, followed by k space separated integers p_j corresponding to the allowed lengths of cables ($1 \leq k \leq 100$, $1 \leq p_j \leq 100$). The next line contains two integers n m , denoting the required number of nodes and cables in the network, respectively ($1 \leq n \leq 100$, $1 \leq m \leq 1000$). The next m lines contain a pair of integers a_j b_j each, signifying that nodes a_j and b_j should be connected by a cable ($1 \leq a_j, b_j \leq n$).

Output

A valid solution to the i -th test case consists of a line with the text 'city i Y', followed by n_i lines each containing two integers, the x- and y-coordinates of successive nodes in the solution ($0 \leq x, y \leq 100$).

It is guaranteed that for every test case there exists at least one possible solution. You can however leave out a test case by outputting the line 'city i N' instead of a valid solution.

Score

For each correctly solved test case you are awarded $(m/sum) * ((p_1 + p_2 + \dots + p_k)/k)$ points, where sum is the total length of all cables used.

The score awarded to your program is the sum of scores for individual test cases.

Example

Input:

```
4
2 1 2
4 5
1 2
2 3
3 4
1 4
2 4
1 2
4 5
1 2
2 3
3 4
1 4
2 4
2 1 2
5 8
1 2
1 3
1 4
1 5
2 4
2 5
3 4
3 5
1 1
2 1
1 2
```

Output:

```
city 1 Y
0 0
0 1
1 1
1 0
city 2 Y
2 0
1 1
0 2
0 0
city 3 Y
0 1
0 2
1 1
1 2
0 0
city 4 N
```

Score:

```
score = 3.340003
```


Bonus info: If score = *xxx.xxxaaa*, *aaa* means the number of test cases with non-zero score...

Added by: Michal Malafiejski

Date: 2004-11-14

Time limit [s]: 42

Source limit [B]:50000

Resource: DASM Programming League 2004 (problemset 3)
English version by Adrian Kosowski

SPOJ Problem Set

234. Getting Rid of the Holidays (Act I)

Problem code: HOLIDAY1

King Johnny of Byteland has in his short period of sovereignty established quite a few national holidays (close on thirty, in fact) in honour of... more or less anything he could think of. Each of these holidays occurs every a fixed number of days (possibly different for every holiday), and is accompanied by feasts, cabaret shows, and general merrymaking. Sometimes more than one holiday occurs on a single day, and once in a while all holidays take place on the same day. If this happens, the celebrations are combined and even more festive. After one such party, king Johnny started behaving strangely and had to be temporarily isolated from society.

For the period of king Johnny's absence (about 48 hours) you have been appointed Regent of Byteland. As a true patriot, you know that holidays are not good for the people, and would like to remove some before king Johnny returns (he won't mind, he never remembers anything after a party anyway). The people however, very sadly, don't know what is good for them, and will revolt if you remove more than k holidays. Try to choose the holidays you remove in such a way as to guarantee that the number of days which elapse between two consecutive holiday parties is as long as possible.

Solve the problem in at most 4kB of source code.

Input

The first line of input contains a single integer $t \leq 200$ - the number of test cases. t test case descriptions follow.

For each test case, the first line contains two space separated integers n k ($1 \leq k < n \leq 30$), denoting the total number of holidays and the number of holidays to be removed. The next line contains n space separated integers, the i -th being t_i ($1 \leq t_i \leq 10^{18}$) - the number of days every which the i -th holiday occurs.

Output

For each test case, output one line containing an increasing sequence of exactly k integers - the numbers of the holidays to be removed (holidays are numbered in the input order from 1 to n).

Example

Input :

```
1
5 2
6 13 10 15 7
```

Output :

```
2 5
```

(The shortest period between two successive holiday parties is 2 days.)

Added by: Adrian Kosowski

Date: 2004-11-25

Time limit [s]: 42

Source limit [B]: 4096

Resource: DASM Programming League 2004, problemset 4

SPOJ Problem Set

235. Very Fast Multiplication

Problem code: VFMUL

Multiply the given numbers.

Input

n [the number of multiplications ≤ 101]

l1 l2 [numbers to multiply (at most 300000 decimal digits each)]

Text grouped in [] does not appear in the input file.

Output

The results of multiplications.

Example

Input:

```
5
4 2
123 43
324 342
0 12
9999 12345
```

Output:

```
8
5289
110808
0
123437655
```

Warning: large Input/Output data, be careful with certain languages

Added by: Darek Dereniowski

Date: 2004-11-27

Time limit [s]: 5

Source limit [B]: 50000

Resource: PAL

SPOJ Problem Set

236. Converting number formats

Problem code: ROMAN

Given the number n of test cases, convert n positive integers less than 2^{32} (given one per line) from one representation to another. For convenience, n is given in the same format as the other numbers.

Input

Input is given by spelling the number in english digits (all upper case letters). Thus the range of (32-bit) input values permissible extends from ZERO (or OH) through FOUR TWO NINE FOUR NINE SIX SEVEN TWO NINE FIVE.

Output

Output 2 lines for each test case. Output is in the form of "extended" Roman numerals (also called "butchered" Roman numerals), with an overline (see sample for details) indicating the value below is "times 1000", and lower-case letters indicating "times 1000000". Thus, the range of (32-bit) output values possible is from through ivccxcivCMLXVIICCXCV, where there is a line above iv and CMLXVII. Note: For values whose residues modulo 1000000 are less than 4000, M is used to represent 1000; for values whose residues are 4000 or greater, I is used. Thus 3999 would read out as MMMCMXCIX 2 while 4000 would readout as IV with an overline. Similar rules apply to the use of M and i for 1000000, and to that of m and i for 1000000000.

WARNING: This problem has a somewhat strict source limit

Example

Input :

```
THREE
FOUR OH
ONE NINE NINE NINE NINE NINE NINE NINE NINE NINE
ONE TWO THREE ZERO FOUR FIVE
```

Output :

```
XL
_____
mcmxcixCMXCIXCMXCIX
_____
CXXMMMMLV
```

Added by: Robin Nittka
Date: 2004-11-30
Time limit [s]: 20
Source limit [B]:2048

SPOJ Problem Set

237. Sums in a Triangle

Problem code: SUMITR

Let us consider a triangle of numbers in which a number appears in the first line, two numbers appear in the second line etc. Develop a program which will compute the largest of the sums of numbers that appear on the paths starting from the top towards the base, so that:

- on each path the next number is located on the row below, more precisely either directly below or below and one place to the right;
- the number of rows is strictly positive, but less than 100;
- all numbers are positive integers between 0 and 99.

Take care about your fingers, do not use more than **256** bytes of code.

Input

In the first line integer n - the number of test cases (equal to about 1000). Then n test cases follow. Each test case starts with the number of lines which is followed by their content.

Output

For each test case write the determined value in a separate line.

Example

Input :

```
2
3
1
2 1
1 2 3
4
1
1 2
4 1 2
2 3 1 1
```

Output :

```
5
9
```

Warning: large Input/Output data, be careful with certain languages

Added by: Lukasz Kuszner
Date: 2004-12-01
Time limit [s]: 4
Source limit 256
[B]:
Resource: 6-th International Olympiad In Informatics July 3-10. 1994. Stockholm - Sweden,
 Problem 1

SPOJ Problem Set

238. Getting Rid of the Holidays (Act II)

Problem code: HOLIDAY2

As King Johnny's temporary indisposition lengthens from days to weeks, and you still hold the office of Regent of Byteland, you begin to feel that acting king is not all that much fun. You encounter various absurdly weird problems. For instance, you find that contrary to your expectations the recent removal of holidays brought about a decrease in the efficiency of the kingdom's workforce.

There appears to be only one rational explanation for all this. It seems that although every holiday occurs every a fixed number of days, the periods between consecutive holidays are long and very irregular. And it is the lack of regularity that is the root of the problem.

So, you decide it is time to tackle the problem once again, and solve it properly this time. Your main purpose is to establish an r -day working rhythm (for some integer r). Workers will work for $(r-1)$ days, have a single day off, work for another $(r-1)$ days, and so on. The rhythm must be arranged in such a way that holidays only ever occur on the day off work. Choose exactly k of the n holidays to remove in such a way as to be able to establish a working rhythm of the maximum possible length r .

Solve the problem in at most 4kB of source code.

Input

The first line of input contains a single integer $t \leq 100$ - the number of test cases. t test case descriptions follow.

For each test case, the first line contains two space separated integers n k ($1 \leq k < n \leq 100$), denoting the total number of holidays and the number of holidays to be removed. The next line contains n space separated integers, the i -th being t_i ($1 \leq t_i \leq 10^{18}$) - the number of days every which the i -th holiday occurs.

Output

For each test case, output one line containing an increasing sequence of exactly k integers - the numbers of the holidays to be removed (holidays are numbered in the input order from 1 to n).

Example

Input:

```
2
6 4
1 3 4 5 6 1
8 4
200 125 200 999 380 500 200 500
```

Output:

```
1 3 4 6
2 4 5 6
```


(In the first test case r is equal to 3 days, in the second case it is equal to 100 days. For the second test case the output '1 2 4 5', '2 3 4 5', '2 4 5 6', '2 4 5 7' or '2 4 5 8' is also correct.)

Added by: Adrian Kosowski

Date: 2004-12-07

Time limit [s]: 42

Source limit [B]: 4096

Resource: DASM Programming League 2004, problemset 4

SPOJ Problem Set

239. Tour de Byteland

Problem code: BTOUR

As the mayor of Byteland's term of office draws to a close, he starts his preparations for reelection. For the first time in the 40 years of his political career his chances of victory seem somewhat uncertain. His main cause of worry are the disturbing results of an opinion poll which state that over 90% of the citizens regard the mayor as a portly, heavily smoking individual who sleeps in his armchair more or less all day.

After careful consultation with his public relations director, the mayor has decided to change his image. He is going to organise, sponsor and compete in... Byteland's first bicycle race! Quite naturally, the only relevant part of the race is the media coverage of the mayor; everything else is to be done at minimum cost. The street-map of Byteland consists of a not necessarily planar system of bi-directional street segments connecting intersections, in such a way that between 0 and 4 street segments meet at an intersection. The cyclists are to ride round and round a simple loop (a fixed, closed route consisting of several street segments, such that a cyclist goes along a street and through an intersection exactly once in each round). For innumerable reasons (not so difficult to guess at) the mayor would like to choose the shortest possible route for the race (in the sense of total street length). Help him determine the length of such a loop, and tell him how many different shortest loops he can choose from when organising the race.

Input

The input starts with a line containing a single integer $t \leq 200$, the number of test cases. t test cases follow.

Each test case begins with a line with two integers n m , denoting the number of intersections and the number of streets in Byteland, respectively ($1 \leq n \leq 1000$). m lines follow, each containing three integers u_i v_i d_i , denoting the end points and the length of the i -th street segment, respectively ($1 \leq u_i \leq v_i \leq n$, $1 \leq d_i \leq 10^6$).

Output

For each test case output a single line containing exactly two space separated non-negative integers d c - the length of the shortest possible race loop, and the number of routes of this length in the graph. Output 0 0 if the race cannot be held.

Example

Input :

```
2
3 2
1 2 1
1 3 2
4 6
1 2 5
1 4 5
```

2 3 4
2 4 5
3 4 5
3 1 5

Output:

0 0
14 2

Added by: Krzysztof Kluczek

Date: 2004-12-09

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004, problemset 4

SPOJ Problem Set

240. Santa Claus and the Presents

Problem code: SANTA

Every year Santa Claus faces a more and more difficult task. The number of children in the world is increasing rapidly, while Santa's old patched up sack can only accommodate a few presents at a time. And every child wants their own very special present... This means that, ever so often, once his sack is partially or completely empty, Santa has to fly back to his base in Lapland to replenish his supplies. So irksome has this become that Santa has decided to resort to modern operational research to aid him in his sack-packing and route planning. Please write a program which will guide Santa through his daily chores.

Input

The input starts with a line containing a single integer $t \leq 100$, the number of test cases. t test cases follow.

The first line of every test case consists of four integers $n \ x \ y \ S$, denoting the number of good children who will receive a present from Santa, the x and y coordinates of Santa's home base, and the amount of space in the sack, respectively ($1 \leq n \leq 10000$, $-10000 \leq x, y \leq 10000$, $1 \leq S \leq 100000$). n lines follow, each consisting of three integers $x_i \ y_i \ s_i$ - the x and y coordinates of the i -th child's home, and the amount of space taken up by this child's present when in Santa's sack, respectively ($-10000 \leq x_i, y_i \leq 10000$, $1 \leq s_i \leq S$).

Output

For each test case output a sequence of space separated integers, corresponding to successive actions that should be taken by Santa:

- $-i$ ($1 \leq i \leq n$) signifies that Santa should travel to his base and pack the present for the i -th child into his sack (he needs to have sufficient room in his sack to do this).
- i ($1 \leq i \leq n$) signifies that Santa should travel to the i -th child's home and leave a present for him/her.
- 0 signifies that Santa should travel back to his base and end his Christmas activity (and that you want to proceed to the next test case).

Assume that Santa starts in his base and always travels between points by the shortest possible route (along straight lines). All distances are measured using the Euclidean metric.

Score

The score awarded to your program is the total of all scores obtained for its individual test cases. The score for a test case is calculated as I/P , where P stands for the distance covered by Santa in your solution, while I is a test case specific constant ($I = n*d + D*(s_1 + \dots + s_n)/S$; d is the mean distance between two homes, while D is the mean distance between Santa's base and a child's home for the given test case).

No points are awarded for incompletely solved test cases (when not all the children receive presents).
Any other violation of transport rules results in Wrong Answer.

Example

Input:

```
1
3 0 0 3
1 0 1
1 0 2
1 0 3
```

Output:

```
-1 -2 1 2 -3 3 0
```

Score:

```
2/(1+1+1+1)= 0.5
```

Added by: Krzysztof Kluczek

Date: 2004-12-09

Time limit [s]: 42

Source limit [B]:50000

Resource: DASM Programming League 2004, problemset 4

SPOJ Problem Set

241. Arranging the Blocks

Problem code: BLOCKS

A group of n children are playing with a set of n^2 flat square blocks. Each block is painted from above with one colour, and there are no more than 2 blocks of each colour. The blocks are initially arranged in an $n \times n$ square forming some sort of picture.

The children have been provided with some other $n \times n$ picture and asked to rearrange the blocks to that form. Since this is not really what they enjoy doing most, they intend to solve the task together and spend as little time on it as possible. Thus, every minute each child chooses a single $1 \times n$ row or $n \times 1$ column of blocks to rearrange. This row/column may never intersect with rows/columns chosen by other children in the same minute. A child takes one minute to perform any rearrangement (permutation) of the blocks within its row/column it likes.

Determine whether the children can perform their task of converting one block image into the other, and if so -- find the minimum possible time in minutes required to achieve this.

Input

The input starts with a line containing a single integer $t \leq 200$, the number of test cases. t test cases follow. Each test case begins with a line containing integer n ($1 \leq n \leq 500$). The next n lines contain n integers P_{ij} each, forming a bitmap matrix representing the colours of the blocks in their initial configuration ($1 \leq P_{ij} \leq n^2$). The following n lines contain n integers Q_{ij} each, corresponding to the matrix for the final configuration ($1 \leq Q_{ij} \leq n^2$).

Output

For each test case output a line with a single non-negative integer corresponding to the number of minutes required to transform matrix P into matrix Q , or the word `no` if no such transformation is possible.

Example

Input :

```
3
3
1 3 4
2 1 3
2 5 5
3 1 3
2 1 2
4 5 5
3
1 2 3
4 5 6
7 8 9
1 5 6
4 2 9
```

```

7 8 3
2
1 2
1 2
1 3
1 2

```

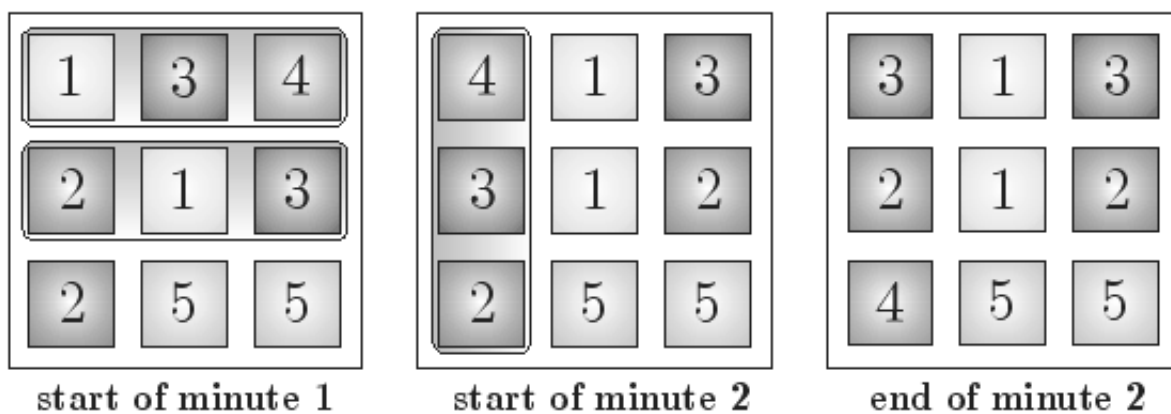
Output:

```

2
1
no

```

The actions taken in the first test case are illustrated below.



Warning: enormous Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2004-12-09

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004, problemset 4

SPOJ Problem Set

243. Stable Marriage Problem

Problem code: STABLEMP

There are given n men and n women. Each woman ranks all men in order of her preference (her first choice, her second choice, and so on). Similarly, each man sorts all women according to his preference. The goal is to arrange n marriages in such a way that if a man m prefers some woman w more than his wife, then w likes her husband more than m . In this way, no one leaves his partner to marry somebody else. This problem always has a solution and your task is to find one.

Input

The first line contains a positive integer $t \leq 100$ indicating the number of test cases. Each test case is an instance of the stable marriage problem defined above. The first line of each test case is a positive integer $n \leq 500$ (the number of marriages to find). The next n lines are the woman's preferences: i th line contains the number i (which means that this is the list given by the i th woman) and the numbers of men (the first choice of i th woman, the second choice,...). Then, the men's preferences follow in the same format.

Output

For each test case print n lines, where each line contains two numbers m and w , which means that the man number m and the woman number w should get married.

Example

Input :

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


7 1 7 4 2 6 5 3

Output:

1 3
2 2
3 1
4 4
1 4
2 5
3 1
4 3
5 7
6 6
7 2

Warning: large Input/Output data, be careful with certain languages

Added by: Darek Dereniowski

Date: 2004-12-13

Time limit [s]: 4

Source limit [B]:50000

Resource: problem known as the *Stable Marriage Problem*

SPOJ Problem Set

244. Internally Stable Sets

Problem code: INTSS

A weighted finite undirected graph is a triple $G = (V, E, w)$ consisting of vertex set V , edge set $E \subseteq \{\{u, v\} : u, v \in V, u \neq v\}$, and vertex weighting function w such that $w(u) \geq 0, \forall u \in V$ and $w(K) = \sum_{u \in K} w(u), K \subseteq V$. For $u \in V$ and $K \subseteq V$, $N(u)$ and $N(K)$ will denote the neighboring vertex sets of u and K respectively, formally defined as:

$$N(u) = \{v : \{u, v\} \in E\}, N(K) = \bigcup_{u \in K} N(u)$$

A vertex set $K \subseteq V$ satisfying $N(K) \cap K = \emptyset$ is called *internally stable* (also known as independent or anti-clique). In this problem you must find an internally stable set B such that $w(B) = \max\{w(S)\}$, where S belongs to the set of all internally stable sets of that graph.

Input

t – the number of test cases [$t \leq 100$]
 $n \ k$ – [n – number of vertices ($2 \leq n \leq 200$), k – number of edges ($1 \leq k \leq n*(n-1)/2$)]
then n numbers follows (w_i – the weight of i -th vertex) [$0 \leq w_i \leq 2^{31}-1$]
then k pairs of numbers follows denoting the edge between the vertices ($s_i \ s_j$ edge between i -th and j -th verices) [$1 \leq s_i, s_j \leq n$]

Output

For each test case output *MaxWeight* – the weight of a maximum internally stable set of the given graph. [$0 \leq \text{MaxWeight} \leq 2^{31}-1$]

Example

Input :

```
2
5 6
10 20 30 40 50
1 2
1 5
2 3
3 4
3 5
4 5

4 4
10 4 10 14
1 2
2 3
3 4
4 1
```

Output:

70

20

Added by: Roman Sol

Date: 2004-12-14

Time limit [s]: 50

Source limit [B]:50000

SPOJ Problem Set

245. Square Root

Problem code: SQRROOT

In this problem you have to find the Square Root for given number. You may assume that such a number exist and it will be always an integer.

Solutions to this problem can be submitted in C, C++, Pascal, Algol, Fortran, Ada, Ocaml, Prolog, Whitespace, Brainfk and Intercal only.**

Input

t - the number of test cases [$t \leq 50$]

then t positive numbers follow, each of them have up to 800 digits in decimal representation.

Output

Output must contain exactly t numbers equal to the square root for given numbers. See sample input/output for details.

Example

Input :

3
36
81
226576

Output :

6
9
476

Added by: Roman Sol
Date: 2004-12-15
Time limit [s]: 10
Source limit [B]:50000

SPOJ Problem Set

246. Plant a Christmas Tree

Problem code: CTQUINE

Evergreen trees are really wonderful. They were treasured by all civilisations of the Western world, from ancient Egyptian priests to Celtic druids in the British Isles. In the late Middle Ages a tradition of placing an evergreen tree at home for Christmas developed in Germany and spread throughout the Old and New World, reaching the Asia Pacific and America in the XIX-th century.

Surely, you must have noticed the sad fact that nowadays this global custom, however beautiful it may be, results in the death of millions of coniferous trees worldwide. Help us in our effort to restore the healthy balance. In the Christmas period, draw & plant your own tiny fir tree!

Since we are limited to text mode, there is little room for creative art, and solid, well built trees are definitely favoured. An *ideal tree* consists of several lines (at least 1) of the same length, consisting of ASCII characters -- both whitespace ("spaces"), and non-whitespace ("relevant characters"). Counting from the top, the number of characters between the first and last relevant characters in a line (inclusive) is equal to 1, 1, 3, 1, 3, 5, 1, 3, 5, 7, 1, 3,... for consecutive lines. The line for which this distance is the largest begins and ends with relevant characters. All other lines contain exactly the same number of spaces to the left of the leftmost relevant character and to the right of the rightmost relevant character (this gives the ideal tree a nice, vertical trunk).

Please write a program which outputs a tree as close to an ideal tree as you can get, and keeps it as small as possible (such a tree has the largest chance of sprouting roots when planted). *And it can hardly come as a surprise to you to learn that the source code of the program you submit has to be identical to the text it writes to output (character by character, there are no exceptions)!*

Score

Your program will be judged as follows: if the program is not a quine (i.e. if it contains no relevant characters or outputs text different than its own source code) it will be judged as a Wrong Answer. Any other program will receive some number of penalty points depending on its size and quality as a tree (the fewer points, the better). One penalty point is given for every line of code used. 10 penalty points are given for a line without any relevant characters (how can you expect a broken tree to grow?). For non-empty lines, the position of the leftmost and rightmost relevant characters in the analyzed tree are compared with respect to corresponding positions in an ideal tree with the same number of lines. The squared differences in position between these two pairs of characters are added to the penalty score.

Technical note: a single newline character (ASCII 10) should be used to terminate all lines. ASCII characters 32 (space) and 9 (tab) are treated as single spaces, all other characters are considered relevant. Notice that the problem description doesn't penalise for left out or excessive spaces after the last relevant character of a line (but doesn't allow any difference between the source code and output text in this respect).

Example

C source code:

```

    i
    i
    i i i
    i
    main(
) { i
    i
    i i
    /* */
;return
    0
    i i }
```

This code would be judged as Wrong Answer, since it isn't a valid quine. Were it a quine, it would receive 16 penalty points (12 for 12 lines, $4=2^2$ additional penalty points for a misplaced rightmost relevant character in line 5).

Solutions to this problem may only be submitted in the following languages: C, C++, Pascal, Java, C#, Python, Haskell, OCaml, Brainfk, Intercal.**

Added by:	Adrian Kosowski
Date:	2004-12-16
Time limit [s]:	42
Source limit [B]:	30000
Resource:	DASM Programming League 2004, problemset 4

SPOJ Problem Set

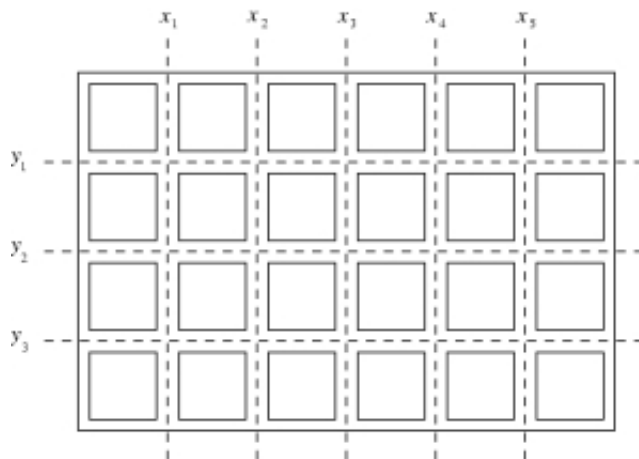
247. Chocolate

Problem code: CHOCOLA

We are given a bar of chocolate composed of $m*n$ square pieces. One should break the chocolate into single squares. Parts of the chocolate may be broken along the vertical and horizontal lines as indicated by the broken lines in the picture.

A single break of a part of the chocolate along a chosen vertical or horizontal line divides that part into two smaller ones. Each break of a part of the chocolate is charged a cost expressed by a positive integer. This cost does not depend on the size of the part that is being broken but only depends on the line the break goes along. Let us denote the costs of breaking along consecutive vertical lines with x_1, x_2, \dots, x_{m-1} and along horizontal lines with y_1, y_2, \dots, y_{n-1} .

The cost of breaking the whole bar into single squares is the sum of the successive breaks. One should compute the minimal cost of breaking the whole chocolate into single squares.



For example, if we break the chocolate presented in the picture first along the horizontal lines, and next each obtained part along vertical lines then the cost of that breaking will be $y_1 + y_2 + y_3 + 4*(x_1 + x_2 + x_3 + x_4 + x_5)$.

Task

Write a program that for each test case:

- Reads the numbers x_1, x_2, \dots, x_{m-1} and y_1, y_2, \dots, y_{n-1}
- Computes the minimal cost of breaking the whole chocolate into single squares, writes the result.

Input

One integer in the first line, stating the number of test cases, followed by a blank line. There will be not more than 20 tests.

For each test case, at the first line there are two positive integers m and n separated by a single space, $2 \leq m, n \leq 1000$. In the successive $m-1$ lines there are numbers x_1, x_2, \dots, x_{m-1} , one per line, $1 \leq x_i \leq 1000$. In the successive $n-1$ lines there are numbers y_1, y_2, \dots, y_{n-1} , one per line, $1 \leq y_i \leq 1000$.

The test cases will be separated by a single blank line.

Output

For each test case : write one integer - the minimal cost of breaking the whole chocolate into single squares.

Example

Input :

1

6 4

2

1

3

1

4

4

1

2

Output :

42

Added by: Thanh Vy Hua Le

Date: 2004-12-23

Time limit [s]: 5

Source limit [B]: 50000

Resource: 10th Polish Olympiad in Informatics, stage 1

SPOJ Problem Set

260. Containers

Problem code: CTAIN

We are given n containers, where $1 \leq n \leq 4$. At the beginning all of them are full of water. The liter capacity of the i -th container is a natural number o_i satisfying inequalities $1 \leq o_i \leq 49$.

Three kinds of moves can be made:

1. Pouring the whole content of one container into another. This move can be made unless there is too little room in the second container.
2. Filling up one container with part of the water from another one.
3. Pouring away the whole content of one container into a drain.

Task

Write a program that for each test case:

- Reads the number of containers n , the capacity of each container and the requested final amount of water in each container.
- Verifies, whether there exist a series of moves which leads to the requested final situation, and if there is one, the program computes the minimal number of moves leading to the requested situation,
- Writes the result. The result should be the minimal number of moves leading to the requested final situation, or one word "NO" if there is no such a sequence of moves.

Input

One integer in the first line, stating the number of test cases, followed by a blank line. There will be not more than 20 tests.

For each test case, at the first line, one positive integer n is written, $n \leq 4$, this is the number of containers. There are n positive integers written in the second line. These are the capacities of the containers (the i -th integer o_i denotes the capacity of the i -th container, $1 \leq o_i \leq 49$). In the third line there are written n numbers. These are the requested final volumes of water in the containers (the i -th integer w_i denotes the requested final volume of water in the i -th container, $0 \leq w_i \leq o_i$). All integers in the second and the third line are separated by single spaces.

The test cases will be separated by a single blank line.

Output

For each test case : write one integer - the minimal number of moves which lead to the requested final situation or write only one word "NO" if it is not possible to reach the requested final situation making only allowed moves.

Example

Input :

2

3

3 5 5

0 0 4

2

20 25

10 16

Output :

6

NO

Added by: Thanh Vy Hua Le

Date: 2004-12-24

Time limit [s]: 10

Source limit [B]:50000

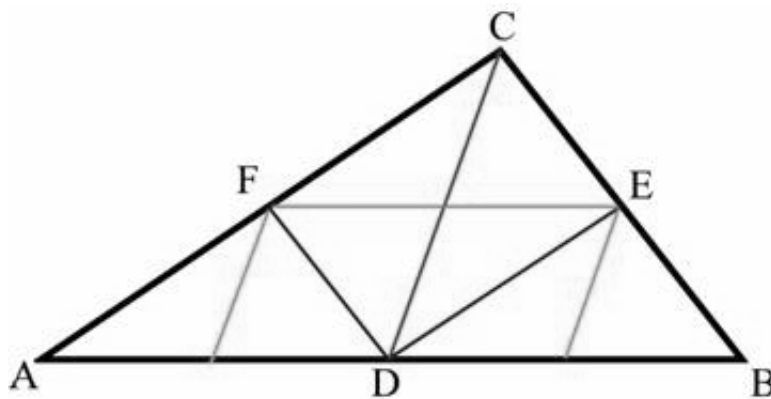
Resource: 3rd Polish Olympiad in Informatics, stage 1

SPOJ Problem Set

261. Triangle Partitioning

Problem code: TRIPART

A triangle can be divided into two equal triangles by drawing a median on its largest edge (in the figure below such a division is shown with the red line). Then the smaller two triangles can be divided in similar fashion into equal triangles (shown in the picture below with blue lines). This process can continue forever.



Some mathematicians have found that when we split a triangle into smaller ones using the method specified above we have only some "styles" of triangles that only differ in size. So now given the lengths of the sides of the triangle your job is to find out how many different styles of small triangles we have. (Two triangles are of same style if they are similar.)

Input

First line of the input file contains an integer N ($0 < N < 35$) that indicates how many lines of input there are.

Each line contains three integers a, b, c ($0 < a, b, c < 100$) which indicate the sides of a valid triangle. (A valid triangle means a real triangle with positive area.)

Output

For each line of input you should produce an integer T , which indicates the number of different styles of small triangles, formed for the triangle at input. Look at the example for details. You can safely assume that for any triangle T will be less than 100.

Example

Input :
2
3 4 5
12 84 90

Output:

3

41

Added by: Thanh Vy Hua Le

Date: 2004-12-24

Time limit [s]: 1

Source limit [B]:50000

Resource: Thanh Vy Hua Le, special thanks to my friends in EPS

SPOJ Problem Set

262. Connections

Problem code: CONNECT

Byteotian Ministry of Infrastructure has decided to create a computer program that helps to find quickly the lengths of routes between arbitrary towns. It would be small wonder if the inhabitants of Byteotia always wanted to find the shortest route. However, it happens that they want to know the k -th shortest route. Moreover, cycles in routes are possible, i.e. routes that have recurring towns.

For example, if there are 4 routes between two towns and their lengths are 2, 4, 4 and 5, then the length of the shortest connection is 2, the second shortest is 4, the third is 4, and the fourth is 5.

Task

Write a program that for each test case:

- Reads a description of Byteotian road network and queries concerning lengths of journey routes.
- Computes and writes answers to the queries read.

Input

One integer in the first line, stating the number of test cases, followed by a blank line. There will be not more than 15 tests.

For each test case, at the first line, there are two positive integers n and m , separated by a single space, $1 \leq n \leq 100$, $0 \leq m \leq n^2 - n$. They are the number of towns in Byteotia and the number of roads connecting the towns, respectively. The towns are numbered from 1 to n .

In each of m successive lines there are three integers separated by single spaces: a , b and l , $a < b$, $1 \leq l \leq 500$. Each triple describes one one-way road of length l enabling to move from the town a to b . For each two towns there exist at most one road that enables to move in the given direction.

In the following line there is one integer q , $1 \leq q \leq 10000$, denoting the number of queries. In the successive q lines there are queries written, one per line. Each query has a form of three integers separated by single spaces: c , d and k , $1 \leq k \leq 100$. Such a query refers to the length of the k -th shortest route from the town c to the town d .

The test cases will be separated by a single blank line.

Output

For each test case, your program should write the answers to the queries read, one answer per line. In the i -th line the answer to the i -th query should be written: one integer equal to the length of the route being sought or -1, when such a route does not exist.

Each test case should be separated by a single blank line.

Example

Input :

```
1

5 5
1 2 3
2 3 2
3 2 1
1 3 10
1 4 1
8
1 3 1
1 3 2
1 3 3
1 4 2
2 5 1
2 2 1
2 2 2
1 1 2
```

Output :

```
5
8
10
-1
-1
3
6
-1
```

Added by: Thanh Vy Hua Le

Date: 2004-12-25

Time limit [s]: 10

Source limit [B]:50000

Resource: 10th Polish Olympiad in Informatics, stage 2

SPOJ Problem Set

263. Period

Problem code: PERIOD

For each prefix of a given string S with N characters (each character has an ASCII code between 97 and 126, inclusive), we want to know whether the prefix is a periodic string. That is, for each i ($2 \leq i \leq N$) we want to know the largest $K > 1$ (if there is one) such that the prefix of S with length i can be written as A^K , that is A concatenated K times, for some string A . Of course, we also want to know the period K .

Input

The first line of the input file will contains only the number T ($1 \leq T \leq 10$) of the test cases.

Each test case consists of two lines. The first one contains N ($2 \leq N \leq 1\,000\,000$) – the size of the string S . The second line contains the string S .

Output

For each test case, output “Test case #” and the consecutive test case number on a single line; then, for each prefix with length i that has a period $K > 1$, output the prefix size i and the period K separated by a single space; the prefix sizes must be in increasing order. Print a blank line after each test case.

Example

Input:

```
2
3
aaa
12
aabaabaabaab
```

Output:

```
Test case #1
2 2
3 3

Test case #2
2 2
6 2
9 3
12 4
```

Added by: Thanh Vy Hua Le
Date: 2004-12-26
Time limit [s]: 5
Source limit [B]: 50000
Resource: ACM South Eastern European Region 2004

SPOJ Problem Set

264. Corporative Network

Problem code: CORNET

A very big corporation is developing its corporate network. At the beginning, each of the N enterprises of the corporation, numbered from 1 to N , organized its own computing and telecommunication center. Soon, for amelioration of the services, the corporation started to collect some enterprises in clusters, each of them served by a single computing and telecommunication center as follows. The corporation chose one of the existing centers I (serving the cluster A) and one of the enterprises J in some other cluster B (not necessarily the center) and linked them with a telecommunication line. The length of the line between the enterprises I and J is $|I - J|(\text{mod } 1000)$. In such a way two old clusters are joined to form a new cluster, served by the center of the old cluster B . Unfortunately after each join the sum of the lengths of the lines linking an enterprise to its serving center could be changed and the end users would like to know what is the new length.

Write a program to keep trace of the changes in the organization of the network that is able at each moment to answer the questions of the users.

Input

The first line of the input file will contains only the number T of the test cases ($1 \leq T \leq 5$). Each test will start with the number N of enterprises ($5 \leq N \leq 20000$). Then some number of lines (no more than 200000) will follow with one of the commands:

E I – asking the length of the path from the enterprise I to its serving center at the moment; **I I J** – informing that the serving center I is linked to the enterprise J . The test case finishes with a line containing the word **O**. There are fewer **I** commands than **N** commands.

Output

The output should contain as many lines as the number of **E** commands in all test cases. Each line must contain a single number – the requested sum of lengths of lines connecting the corresponding enterprise with its serving center.

Example

Input :

```
1
4
E 3
I 3 1
E 3
I 1 2
E 3
I 2 4
E 3
O
```

Output :

0
2
3
5

Added by: Thanh Vy Hua Le
Date: 2004-12-27
Time limit [s]: 2
Source limit [B]:50000
Resource: ACM South Eastern European Region 2004

SPOJ Problem Set

270. Digits of Pi

Problem code: PIVAL

In this problem you have to find as many digits of PI as possible.

Output

Output must contain as many digits of PI as possible (not more than 1000000).

Score

The score awarded to your program will be the first position of the digit where the first difference occurred.

Example

Output:

3.1415926535897932384626433832795

will be awarded with 33 points.

Added by: Roman Sol
Date: 2004-12-27
Time limit [s]: 60
Source limit [B]: 4096

SPOJ Problem Set

272. Cave Exploration

Problem code: CAVE

A long time ago one man said that he had explored the corridors of one cave. This means that he was in all corridors of the cave. Corridors are really horizontal or vertical segments. A corridor is treated as visited if he was in at least one point of the corridor.

Now you want to know if this is true. You have a map of the cave, and you know that the explorer used the following algorithm: he turns left if he can, if he can't he goes straight, if he can't he turns right, if he can't he turns back. Exploration ends when the man reaches the entry point for the second time. Your task is to count how many corridors weren't visited by explorer.

Input

In the first line there is an integer **T** ($T \leq 20$) - the number of different maps. For each map in the first line there is an integer **N** ($N \leq 1000$) - the number of corridors. It is known that no two vertical corridors have a common point and no two horizontal corridors have a common point. The next **N** lines contain the following information: the line starts with one of the characters 'V' or 'H' - vertical or horizontal corridor. Then one Y-coordinate and two X-coordinates are given for a horizontal corridor or one X-coordinate and two Y-coordinates for a vertical corridor. The last line for each map contains the X and Y coordinates of the entry point (start and end point of travel) and the direction ('W' - left, 'E' - right, 'N' - up and 'S' - down). You may assume that: the entry point is not located at the cross-point of two corridors, and the explorer can always move forward in the direction given in the input. All coordinates are integers and do not exceed 32767 by absolute value and there are no more than 500 vertical corridors and no more than 500 horizontal corridors.

Output

For each map the program has to print the number of unvisited corridors (in a separate line).

Example

Input :

```
2
6
H 0 6 0
H 2 1 6
V 1 0 4
V 5 3 0
V 3 0 2
H 1 2 4
6 0 W
1
V 0 -5 5
0 0 S
```

Output :

```
1
0
```



Added by: Thanh Vy Hua Le
Date: 2004-12-31
Time limit [s]: 5
Source limit [B]: 50000
Resource: ACM South Eastern European Region 2004

SPOJ Problem Set

273. The Modern Dress Code

Problem code: DCODE

Byteland is making preparations for the approaching period of *feasts, cabaret shows, and general merrymaking*. The women in Byteland are especially excited about the Grand Ball to be held on the last Saturday of the holiday period, and have already started making preparations. At present, one of the most important questions for every lady seems to be: *What should I put on?*. This problem is far more important than you might imagine possible, since the months after the period of feasts will offer no celebrations whatsoever, and during this time every lady likes to think back to how charming and original she looked at the ball. And of course, you have to remember that if two female friends come to the party dressed in the same dresses they will both be the subject of jokes and chuckled remarks for years.

As in most difficult problems among ladies, the gentlemen of Byteland are forcibly involved and asked to help out. So, every day each woman asks her man for advice on what to wear. Unfortunately very few men in Byteland know anything at all about fashion so they all come to you and ask for help.

You do not have a lot of to do with Bytelandian fashion either but, as a computer specialist and a generally clever person who never gives up, you know how to address any problem. You have modeled the situation as follows:

- there are $1 \leq k \leq 20$ days left till the grand feast,
- dresses in Byteland come in different colors, which you have numbered with integers starting from 0,
- the relationships between Bytelandian women are given by a graph (adjacent nodes denote two friends),
- every lady would like to have a dress of different color than all her friends, otherwise she will be most unhappy.

You have a plan to provide the men with a simple set of rules to help them determine the best, most unique color of clothes for their ladies. Since you only have a limited amount of time to spare, each man will have to content himself with the same algorithm from you. Because most of the men don't use a computer, you should use a very simple language to express your algorithm.

Task

Prepare an algorithm which, if applied properly by all the men, will guarantee that as few of the ladies as possible are unhappy due to their dresses.

A short description of the language you should use is given below:

- each program is a list of rules,
- each `< rule >` is of the form:

```

< rule > ::= if < comp_condition > then { < comp_command > };

< comp_command > ::= < command >;
< comp_command > ::= < command >; < comp_command >

< comp_condition > ::= < condition >
< comp_condition > ::= not < comp_condition >
< comp_condition > ::= (< comp_condition > or < comp_condition >)
< comp_condition > ::= (< comp_condition > and < comp_condition >)

```

Please note that after each < command > and < rule > you must put a semicolon.

- < command > is simply an assignment of the form:

```

< command > ::= < symbol > := < expr >

```

- < symbol > can be one of the following strings:
 - a state component: color, a, b,
 - an additional local variable: 10, 11, 12, 13, 14, 15, 16, 17, 18, 19.
- The expression < expr > should be constructed using:
 - writable symbols (as given above) and several special read-only variables which give you some information about the graph:

deg	degree of current vertex
delta	maximum vertex degree in a graph
n	number of vertices
m	number of edges
nr	number of possible colors
daysleft	number of days remaining till the ball

- operators (in order of precedence):

+ -	arithmetic plus and minus
* / %	arithmetic multiplication, division and remainder

- function rnd(< expr >), returning a random number between 0 and < expr > - 1 inclusive.
- functions mina, minb, minc returning the minimal value of variable a, b, color respectively, taken over all the neighbours of the vertex (or the smallest integer for a vertex without neighbours). Functions maxa, maxb, maxc act similarly for maximum values of variables.
- < condition > is a logical expression taking one of two forms:

```

< condition > ::= < expr > < operator > < expr >
< condition > ::= < exist-operator > ( < expr > )

```

The binary < operator > is one of the comparison operators ==, <, or >. The unary < exist-operator > is one of the following functions: Eaeq, Ebeq, Eceq, which return true iff for some neighbour of the given vertex the value of variable a, b, color, respectively, is equal to < expr >.

The algorithm has to be applied by every man every day. More formally speaking: at the start of the process all variables have random values. Every morning all the men assign 0 to their variables 10, 11, ..., 19. Exactly at noon, each lady comes to ask for advice. The man does all he can to help her: he processes all his rules from top to bottom, and repeats the process as long as at least one of the IF clauses is true. (However, if he has to do it more than one hundred times, he will give up in despair). Then he tells the lady the color he has chosen for her. Every evening, the men meet at the pub, and

when they have nothing left to talk about, they exchange information about how the women are boring them, and what values of `a`, `b` and `color` they have had to come up with today. (It is interesting to note that, as a side effect, functions `E_eq`, `min_` and `max_` actually make use of the previous day's values of the neighbours' variables.)

Finally, on the day of the ball the women put on clothing corresponding to the color last suggested by their men and it is time to judge how effective your algorithm has been. (This is one of these tasks which, if done badly, may result in getting the programmer lynched by an angry mob.)

Score

There are 50 test cases on which your program will be tested. Graphs used in tests will have no more than 25 vertices. Your score is the sum of scores of your program taken over all test cases. For each test case the score is equal to the ratio of the number of correctly colored vertices (i.e. such that $0 \leq \text{color} < \text{nr}$ and `Eeq(color)` is false) to the total number of vertices. If all vertices are correctly colored, a bonus of $1/(1 + \text{maximum color used})$ points is awarded.

A program will be assessed as Compilation Error if it cannot be interpreted due to syntactic errors. If on a given day the rules for a vertex cannot be processed within 100 iterations or the entire simulation takes more than about 60s, your program will be judged as Time Limit Exceeded.

Example

Submit code in the language TEXT, the judge will interpret it properly. Here is an example of a simple single-rule program.

Code:

```
if (l0==0 and not Eeq ((color-1)%n)) then {color:=(color-1)%n; l0:=1};
```

Score:

5.491

Added by: Lukasz Kuszner

Date: 2004-12-31

Time limit [s]: 60

Source limit [B]:50000

Resource: DASM Programming League 2004, problemset 5

SPOJ Problem Set

274. Johnny and the Watermelon Plantation

Problem code: WMELON

Shortly after his abdication from the Bytelandian throne Johnny decided to go into farming. Water melons were a natural choice as his first crop ever, since they seemed easy enough to grow and look after. So, he sold all his beer bottles and for the money he purchased a 1km x 1km square field. Here it was that he planted the water melon seeds. (The word 'planted' is really a bit of a euphemism for walking across a field gorging on a water melon and spitting out the pips but, for the sake of politeness, let us leave it this way).

To everyone's surprise a lot of the seeds sprouted stems, and soon enough many of the plants showed signs of fruit (and some had even more than one!). Then quite unexpectedly, when the water melons were still a little too unripe to eat, winter set in. Johnny knows that he has to construct a green house to protect the field but, with his rather limited budget, he cannot afford the glass to cover the whole area. He has decided that it is enough that k fruit survive the ordeal under a glazed roof. For reasons of architectural planning in Byteland it is necessary that the green house be a rectangle with sides parallel to the edges of the plot.

You have been requested to help Johnny minimise investment costs. Since glass is paid for by the square meter, design a green house with the smallest possible area fulfilling the imposed conditions.

Input

The first line of input contains the integer $t \leq 100$, the number of test cases. t test cases follow.

Every test case begins with a line containing two integers n k , denoting the total number of plants and the number of water melon fruit to be protected, respectively ($1 \leq n \leq 1000$, $1 \leq k \leq 10^6$, k doesn't exceed the total number of fruit in the plantation). Each of the next n lines describes a single plant, the i -th line containing three integers x_i y_i f_i - the X and Y coordinates of the plant, and the number of water melon fruit on it, respectively ($1 \leq x_i, y_i, f_i \leq 1000$).

Output

For each test case output a single integer, denoting the area of the smallest possible rectangular glass house with horizontal and vertical edges, sufficient to cover at least k fruit of the plantation.

Example

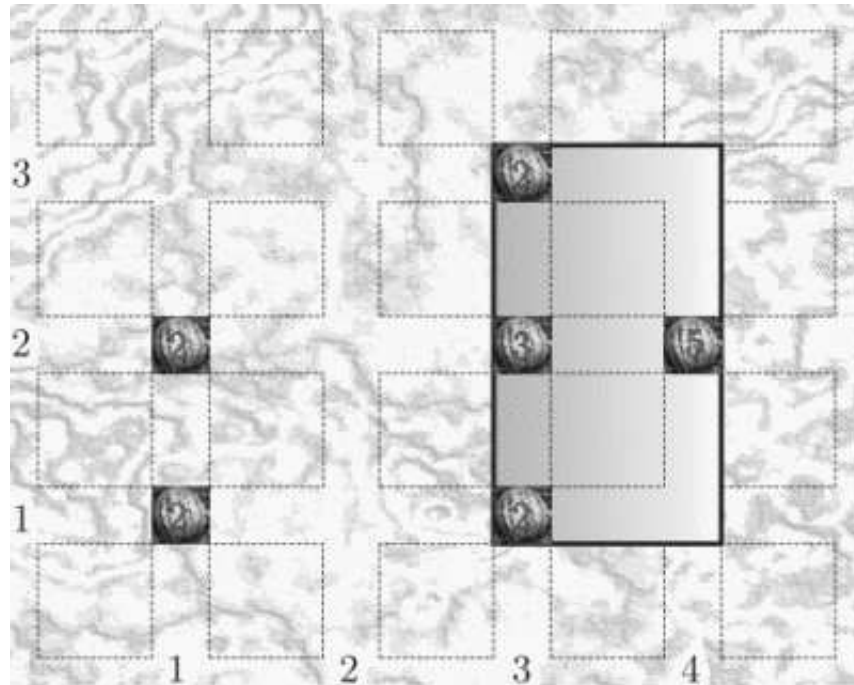
Input :

```
1
6 11
1 1 2
1 2 2
3 1 2
3 2 3
4 2 5
```


3 3 2

Output:

2



Added by: Adrian Kosowski

Date: 2005-01-03

Time limit [s]: 42

Source limit 50000
[B]:

Resource: DASM Programming League 2004, problemset 5 (acknowledgement to Thanh Vy Hua Le)

SPOJ Problem Set

275. The Water Ringroad

Problem code: WATERWAY

There is a land far, far away where the entire population dwells in walled cities at the peaks of mountains on the circumference of a plateau known as The Circle. The High Councillors of the cities developed an intricate system of communication: the cities were connected into a cycle by a perfectly round waterway. If need arose, a small paper boat with a message tied to its sail was released into the waterway and was guided by its solitary crew member (a small tin soldier) from one city to the next, and so on, until it reached its destination. Some segments of the waterway were only passable in one direction (due to waterfalls), and so there may have been pairs of cities for which communication was impossible.

As the centuries went by, the system slowly began to show its weaknesses. The waterway was so narrow that two boats going in opposite directions could never pass each other. To make matters worse, some of the more enterprising cities replaced the tin soldier by a plastic one to increase the speed of the boat, and the faster boats had to queue up behind the slower ones, and everyone got very angry indeed. The councillors gathered to address the problem and found that the best course of action would be to construct two separate channels between every pair of communicating cities A and B: one for carrying messages from A to B, the other from B to A (if communication was impossible in some direction in the old waterway, it needn't be enabled in the new one).

The High Priests of the Circle were the first to protest against the plan. They insisted that any waterway ever built should be circular and go round all the cities in the same manner as the original one, and the route of any boat must always be a perfect arc between any two adjacent cities. So the newly designed channels would in fact have to be composed of sets of adjacent fragments of circles, without any two channels sharing an arc.

The engineers have quite rightly pointed out that the new circles will be prone to the same problem of waterfalls on the same sections as the original waterway. Bearing this in mind, given a map of the old waterway, calculate the smallest possible number of circles the new waterway may consist of.

Input

Input begins with integer $t \leq 100$, the number of test cases. t test cases follow.

Each test case consists of two lines. The first contains a single integer n ($3 \leq n \leq 100000$), the number of cities around the Circle. The second line is a description of the old waterway - a sequence of exactly n characters 'A', 'B' or 'C', without separating spaces, terminated by a new line. These characters correspond to the state of the arcs between cities 1 and 2, 2 and 3, ..., $n-1$ and n , n and 1, respectively, and mean: 'A' - the arc is passable when going anticlockwise, 'B' - the arc is passable in both directions, 'C' - the arc is passable when going clockwise.

Output

For each test case output a line, containing a single integer - the number of circles required for the new waterway.

Example

Input :

2

3

AAA

4

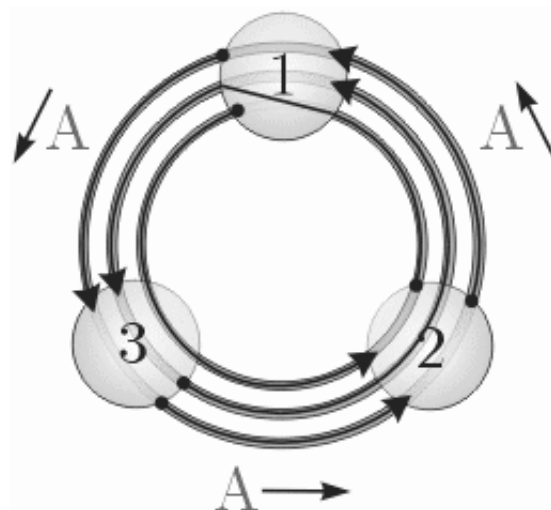
BACB

Output :

3

5

A solution to the first test case which requires 3 circles is presented below.



Added by: Adrian Kosowski

Date: 2005-01-03

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004, problemset 5

SPOJ Problem Set

276. Herdkeeper

Problem code: MFENCE

After the tragic end of the watermelon plantation, Johnny has switched to farming. More precisely, he is now a Certified Livestock Supervisor (i.e. shepherd) tending herds of antelope. It is his task to divide the animals into herds, and to build a fence around each herd, trying to keep the total length of all fences as small as possible. Each herd must consist of at least 2 antelope, and the antelope may stand arbitrarily close to the fence itself.

Input

t [the number of test cases ≤ 1000]
 n [$2 \leq$ the number of antelope ≤ 100]
 $x_1 \ y_1$ [coordinates of the antelope, between -1000 and 1000]
 $x_2 \ y_2$
.....
 $x_n \ y_n$
[next test cases]

Output

case i Y [N - if you want to skip the testcase]
 c [the number of herds]
 $a \ s_1 \ s_2 \ \dots \ s_a$ [$2 \leq a$ - the number of antelope in the first herd, and the numbers of the antelope in this herd in the order from the input sequence]
[next test cases]

Scoring

The score awarded to your program is the sum of scores for individual test cases. For the i -th test case you will receive $1 / (1 + \text{sum} / \text{conv})$ points, where sum is the sum of circumferences of all convex hulls of herds in your solution, and the conv is the circumference of the convex hull of the set of all antelope. If you don't want to solve the i -th test case, you may output the line 'case i N' and nothing else.

Example

Input :
6
2
0 0
5 0
3
4 0
-4 -5
2 3
5

```
20 10
10 10
40 50
-20 -40
-30 -20
4
2 4
2 -4
2 0
-5 -3
3
2 4
-4 -4
2 3
4
-1 -3
-1 5
3 -5
-1 5
```

Output:

```
case 1 Y
1
2 1 2
case 2 Y
1
3 1 2 3
case 3 Y
2
3 1 2 3
2 4 5
case 4 Y
2
2 1 4
2 2 3
case 5 Y
1
3 1 2 3
case 6 Y
1
4 1 2 3 4
```

Score:

```
3.079001
```

Bonus info: If score = *xxx.xxxaaa*, *aaa* means the number of test cases with score > 0.5

Added by: Michal Malafiejski

Date: 2005-01-07

Time limit [s]: 42

Source limit [B]:50000

Resource: DASM Programming League 2004, Problemset 6

SPOJ Problem Set

277. City Game

Problem code: CTGAME

Bob is a strategy game programming specialist. In his new city building game the gaming environment is as follows: a city is built up by areas, in which there are streets, trees, factories and buildings. There is still some space in the area that is unoccupied. The strategic task of his game is to win as much rent money from these free spaces. To win rent money you must erect buildings, that can only be rectangular, as long and wide as you can. Bob is trying to find a way to build the biggest possible building in each area. But he comes across some problems - he is not allowed to destroy already existing buildings, trees, factories and streets in the area he is building in.

Each area has its width and length. The area is divided into a grid of equal square units. The rent paid for each unit on which you're building stands is 3\$.

Your task is to help Bob solve this problem. The whole city is divided into **K** areas. Each one of the areas is rectangular and has a different grid size with its own length **M** and width **N**. The existing occupied units are marked with the symbol **R**. The unoccupied units are marked with the symbol **F**.

Input

The first line of the input contains an integer **K** - determining the number of datasets. Next lines contain the area descriptions. One description is defined in the following way: The first line contains two integers-area length **M** ≤ 1000 and width **N** ≤ 1000, separated by a blank space. The next **M** lines contain **N** symbols that mark the reserved or free grid units, separated by a blank space. The symbols used are:

R - reserved unit

F - free unit

In the end of each area description there is a separating line.

Output

For each data set in the input print on a separate line, on the standard output, the integer that represents the profit obtained by erecting the largest building in the area encoded by the data set.

Example

Input :

```
2
5 6
R F F F F F
F F F F F F
R R R F F F
F F F F F F
F F F F F F
```

5 5
R R R R R
R R R R R
R R R R R
R R R R R
R R R R R

Output:

45
0

Added by: Thanh Vy Hua Le
Date: 2005-01-08
Time limit [s]: 5
Source limit [B]: 50000
Resource: ACM South Eastern European Region 2004

SPOJ Problem Set

278. Bicycle

Problem code: BICYCLE

Peter likes to go to school by bicycle. But going by bicycle on sidewalks is forbidden and going along roads is dangerous. That's why Peter travels only along special bicycle lanes. Fortunately Peter's home and school are in the immediate proximity of such paths. In the city where Peter lives there are only two bicycle lanes. Both lanes have the form of a circle. At the points where they cross it is possible to move from one path to the other. Peter knows the point where he enters the road and the point at which it is necessary to leave to enter the school. Peter is interested in the question: "What is the minimal distance he needs to cover along the lanes to get to school?"

Input

t – the number of test cases [$t \leq 100$], then t test cases follow.

The first 2 lines of each test case contain the description of the bicycle lanes:

$x1\ y1\ r1$ - 3 integers ($x1, y1$ - coordinates of the center of the 1st circle, $r1$ - radius of 1st circle)

$x2\ y2\ r2$ - 3 integers ($x2, y2$ - coordinates of the center of the 2nd circle, $r2$ - radius of 2nd circle)

$-200 \leq x1, x2, y1, y2 \leq 200$

$0 \leq r1, r2 \leq 200$

Next 2 lines contain the coordinates of Peter's home and school:

$px1, py1$ - 2 real numbers

$px2, py2$ - 2 real numbers

You may assume that this points lie on the circle with high accuracy (10^{-8}). Both points may lie on the same circle.

Output

For each test case output the minimum distance that Peter needs to go from home to get to school. The precision of the answer must be under 0.0001. If it's impossible to get to school using the bicycle lanes output -1.

Example

Input :

3

0 0 5
4 0 3
3.0 4.0
1.878679656440357 -2.121320343559643

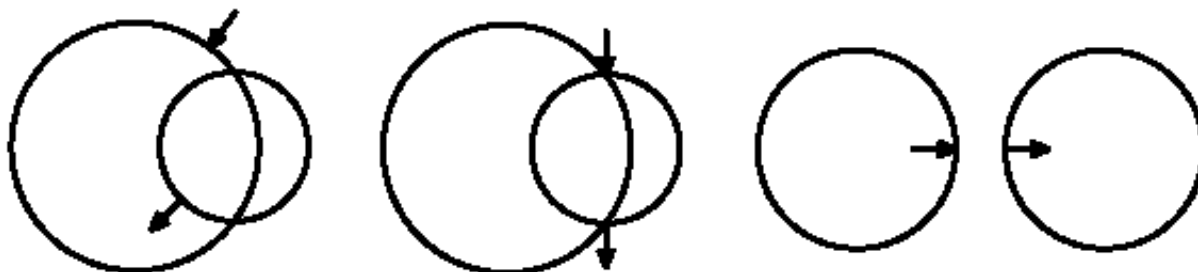
0 0 5
4 0 3
4.0 3.0
4.0 -3.0

0 0 4
10 0 4

4.0 0.0
6.0 0.0

Output:

8.4875540166
6.4350110879
-1



Added by: Roman Sol
Date: 2005-01-13

Time limit [s]: 1

Source limit [B]:50000

Resource: 5th Russian National Command Olympiad for schoolboys in programming

SPOJ Problem Set

279. Interesting number

Problem code: INUMBER

For the given number n find the minimal positive integer divisible by n , with the sum of digits equal to n .

Input

t – the number of test cases, then t test cases follow. ($t \leq 50$)

Test case description:

n - integer such that $0 < n \leq 1000$

Output

For each test case output the required number (without leading zeros).

Example

Input :

2
1
10

Output :

1
190

Added by: Roman Sol

Date: 2005-01-13

Time limit [s]: 15

Source limit [B]: 4096

Resource: XII team championship of St.-Petersburg in programming

SPOJ Problem Set

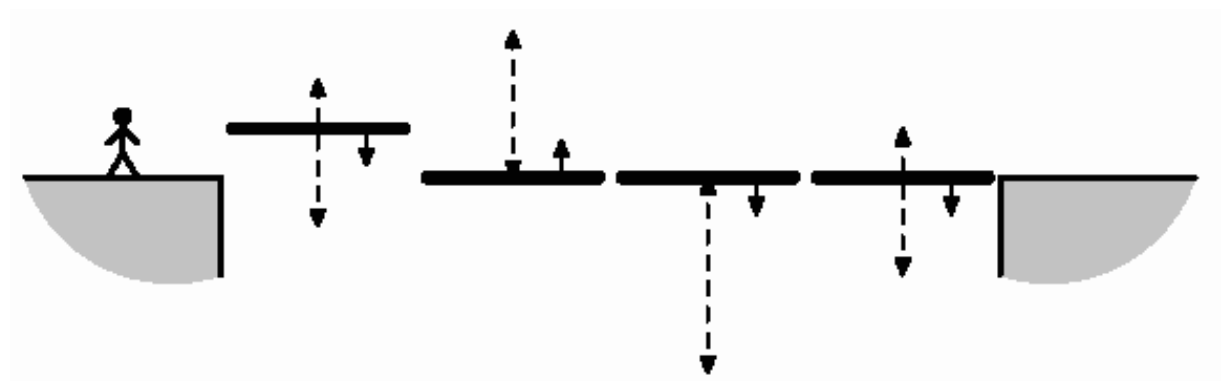
280. Lifts

Problem code: LIFTS

Serj likes old games very much. Recently he has found one arcade game in his computer. When controlling the hero it is necessary to move on a map and collect various items. At a certain stage of the game Serj has faced an unexpected problem. To continue his adventures the hero should get past over a chasm. For this purpose it is possible to use consistently located lifts which look like horizontal platforms. Each lift moves up-down vertically between some levels. The hero can pass between the next adjacent platform, however it can be done only at the moment when they are at the same level. Similarly, passing from the edge of a chasm onto the lift and vice versa is only possible at the moment when the lift appears on the level of the edge.

Each lift has a width equal to 4 meters. At the beginning the hero is in at a distance of two meters from the edge of a chasm. He should finish travel two meters after the opposite edge of the chasm. The hero moves at a speed of 2 meters a second. Thus, if the hero is in the initial position or in the center of the lift and wishes to pass to the next lift (or to descend from last lift onto the opposite edge of a chasm), he should begin movement exactly one second before they meet at one level. In two seconds the hero appears in the center of the next lift (or in the final position on the other side).

The edges of the chasm are at the same level. For each lift the range of heights between which it moves, its initial position and the direction of movement at the initial moment are given. All lifts move with a speed of one meter a second. Find out whether the hero can get over to the opposite edge of the chasm, and if so what the minimal time required for this purpose is.



Input

t – the number of test cases, then t test cases follows.

[empty line]

A test case begins with n - the number of lifts, a positive integer ($n \leq 100$), then n lines follow. The i -th line ($0 < i \leq n$) contains four integers $li\ ui\ si\ di$, where: li - lowest position of the lift, ui - highest position of the lift, si - initial position of the lift, di - initial direction of movement (1 means up, -1 means down); ($-100 \leq li \leq si \leq ui \leq 100, 11 < ui$).

Output

For each test case output the minimal time in seconds, required to get to the opposite edge of the chasm. If it is impossible output -1.

Example

Input:

1

4

-1 2 1 -1

0 3 0 1

-4 0 0 -1

-2 1 0 -1

Output:

29

Added by: Roman Sol

Date: 2005-01-17

Time limit [s]: 6

Source limit [B]: 10000

Resource: 5th Russian National Command Olympiad for schoolboys in programming

SPOJ Problem Set

282. Muddy Fields

Problem code: MUDDY

Rain has pummeled on the cows' field, a rectangular grid of R rows and C columns ($1 \leq R \leq 50$, $1 \leq C \leq 50$). While good for the grass, the rain makes some patches of bare earth quite muddy. The cows, being meticulous grazers, don't want to get their hooves dirty while they eat.

To prevent those muddy hooves, Farmer John will place a number of wooden boards over the muddy parts of the cows' field. Each of the boards is 1 unit wide, and can be any length long. Each board must be aligned parallel to one of the sides of the field.

Farmer John wishes to minimize the number of boards needed to cover the muddy spots, some of which might require more than one board to cover. The boards may not cover any grass and deprive the cows of grazing area but they can overlap each other.

Compute the minimum number of boards FJ requires to cover all the mud in the field.

Input

t – the number of test cases, then t test cases follows.

Each test case is of the following form:

Two space-separated integers: R and C , then R lines follows

Each line contains a string of C characters, with '*' representing a muddy patch, and '.' representing a grassy patch. No spaces are present.

Output

For each test case output a single integer representing the number of boards FJ needs.

Example

Input:

```
1
4 4
*.*.
.***
***.
..*.
```

Output:

```
4
```

Output details:

Boards 1, 2, 3 and 4 are placed as follows:

```
1.2.
.333
444.
..2.
```

Board 2 overlaps boards 3 and 4.

Added by: Roman Sol
Date: 2005-01-19
Time limit [s]: 10
Source limit [B]: 30000
Resource: USACO January 2005 Gold Division

SPOJ Problem Set

283. Naptime

Problem code: NAPTIME

Goneril is a very sleep-deprived cow. Her day is partitioned into N ($3 \leq N \leq 3,830$) equal time periods but she can spend only B ($2 \leq B < N$) not necessarily contiguous periods in bed. Due to her bovine hormone levels, each period has its own utility U_i ($0 \leq U_i \leq 200,000$), which is the amount of rest derived from sleeping during that period. These utility values are fixed and are independent of what Goneril chooses to do, including when she decides to be in bed.

With the help of her alarm clock, she can choose exactly which periods to spend in bed and which periods to spend doing more critical items such as writing papers or watching baseball. However, she can only get in or out of bed on the boundaries of a period.

She wants to choose her sleeping periods to maximize the sum of the utilities over the periods during which she is in bed. Unfortunately, every time she climbs in bed, she has to spend the first period falling asleep and gets no sleep utility from that period.

The periods wrap around in a circle; if Goneril spends both periods N and 1 in bed, then she does get sleep utility out of period 1 .

What is the maximum total sleep utility Goneril can achieve?

Input

t – the number of test cases, then t test cases follow.

Each test case takes the following form:

Two space-separated integers: N and B , then N lines follows

Each line contains a single integer, U_i , between 0 and 200,000 inclusive

Output

For each test case output a single integer, the maximum total sleep utility Goneril can achieve.

Example

Input:

```
1
5 3
2
0
3
1
4
```

Output:

```
6
```

Input/Output details:

The day is divided into 5 periods, with utilities 2, 0, 3, 1, 4 in that order. Goneril must pick 3 periods.

Goneril can get total utility 6 by being in bed during periods 4, 5, and 1, with utilities 0 [getting to sleep], 4, and 2 respectively.

Added by: Roman Sol
Date: 2005-01-19
Time limit [s]: 10
Source limit [B]:50000
Resource: USACO January 2005 Gold Division

SPOJ Problem Set

285. Atomic Shelters

Problem code: ATSHELT

The election campaign of the mayor of Byteland continues. His advisors firmly believe that a military touch might do good to his image. On the other hand, aggressive use of arms might arouse the insane anger of the pacifist part of the electorate. So, investing in national defence seems to be the best solution. And this is why the capital of Byteland will receive its first ever atomic shelters.

The Bytelandian capital consists of exactly n buildings and the mayor intends to build shelters underneath exactly k of them. Now it is your task to layout the shelters in the city in such a way as to minimise the maximum distance a citizen of Byteland may have to cover to reach the nearest atomic shelter. After all, there is nothing more important than a mayor who guarantees your safety by putting an atomic shelter not far from your house.

Input

t [the number of test cases ≤ 1000]
 $n\ k$ [$2 \leq$ the number of different buildings ≤ 100 , $1 \leq$ the number of shelters $\leq n-1$]
 $x_1\ y_1$ [$-1000 \leq$ coordinates ≤ 1000]
 $x_2\ y_2$
.....
 $x_n\ y_n$
[next test cases]

Output

case i Y [N if you wish to skip this test case]
 $b_1\ b_2 \dots b_k$ [numbers of buildings to put shelters in, in increasing input order]
[next test cases]

Scoring

The score awarded to your program is the sum of scores for individual test cases. For the i -th test case you will receive $diam / dist$ points, where $diam$ is the distance between the furthest two buildings of the city, while $dist$ is the longest distance from a building to the nearest shelter in your solution. If you don't want to solve the i -th test case, you may output the line 'case i N' and nothing else.

Example

Input :
5
5 2
-3 -4
-4 3
2 -3
-2 -3
-5 5

```
5 4
2 0
-5 -4
1 -1
-1 0
5 -5
5 2
-3 0
5 -2
-1 -5
2 4
4 5
5 3
5 0
-1 -5
3 2
-5 1
-1 3
5 4
-1 2
1 1
5 4
0 5
-2 2
```

Output:

```
case 1 Y
3 4
case 2 Y
1 3 4 5
case 3 Y
4 5
case 4 Y
1 2 3
case 5 N
```

Score:

5.592004

Bonus info: If score = *xxx.xxxxaaa*, *aaa* means the number of test cases with Y answer.

Added by: Michal Malafiejski

Date: 2005-01-21

Time limit [s]: 42

Source limit [B]:50000

Resource: DASM Programming League 2004, Problemset 6

SPOJ Problem Set

286. Selfish Cities

Problem code: SCITIES

Far, far away there is a world known as Selfishland because of the nature of its inhabitants. Hard times have forced the cities of Selfishland to exchange goods among each other. C1 cities are willing to sell some goods and the other C2 cities are willing to buy some goods (each city can either sell or buy goods, but not both). There would be no problem if not for the selfishness of the cities. Each selling city will sell its goods to one city only, and each buying city will buy goods from one city only. Your goal is to connect the selfish cities in such a way that the amount of exchanged goods is maximalized.

Input

The first line contains a positive integer $t \leq 1000$ indicating the number of test cases. Each test case is an instance of the problem defined above. The first line of each test case is a pair of positive integers C1 and C2 (the number of cities wanting to sell their goods $C1 \leq 100$ and the number of cities wanting to buy goods $C2 \leq 100$). The lines that follow contain a sequence of (c1,c2,g) trios ending with three zeros. (c1,c2,g) means that the city c1 can offer the city c2 the amount of $g \leq 100$ goods.

Output

For each test case print the maximal amount of goods exchanged.

Example

Input :

```
3
3 2
1 1 10
2 1 19
2 2 11
3 2 1
0 0 0
4 4
1 1 6
1 2 6
2 1 8
2 3 9
2 4 8
3 2 8
4 3 7
0 0 0
3 2
1 1 10
2 1 21
2 2 11
3 2 1
0 0 0
```

Output:

21
29
22

Added by: Tomasz Niedzwiecki
Date: 2005-01-22
Time limit [s]: 20
Source limit [B]:50000

SPOJ Problem Set

287. Smart Network Administrator

Problem code: NETADMIN

The citizens of a small village are tired of being the only inhabitants around without a connection to the Internet. After nominating the future network administrator, his house was connected to the global network. All users that want to have access to the Internet must be connected directly to the admin's house by a single cable (every cable may run underground along streets only, from the admin's house to the user's house). Since the newly appointed administrator wants to have everything under control, he demands that cables of different colors should be used. Moreover, to make troubleshooting easier, he requires that no two cables of the same color go along one stretch of street.

Your goal is to find the minimum number of cable colors that must be used in order to connect every willing person to the Internet.

Input

t [the number of test cases, $t \leq 500$]
 $n \ m \ k$ [$n \leq 500$ the number of houses (the index of the admin's house is 1)]
[m the number of streets, k the number of houses to connect]
 $h_1 \ h_2 \ \dots \ h_k$ [a list of k houses wanting to be connected to the network, $2 \leq h_i \leq n$]
[The next m lines contain pairs of house numbers describing street ends]
 $e_{11} \ e_{12}$
 $e_{21} \ e_{22}$
...
 $e_{m1} \ e_{m2}$
[next cases]

Output

For each test case print the minimal number of cable colors necessary to make all the required connections.

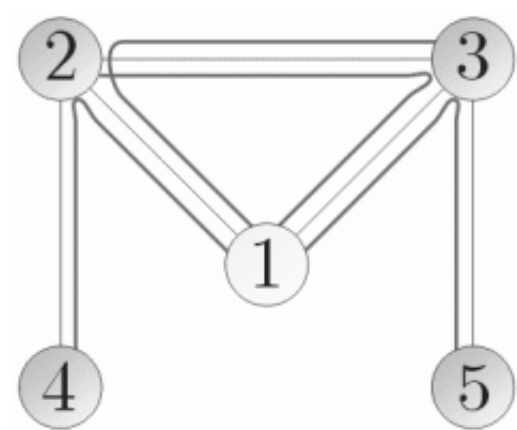
Example

Input :
2
5 5 4
2 3 4 5
1 2
1 3
2 3
2 4
3 5
8 8 3
4 5 7
1 2
1 8
8 7

```
1 3
3 6
3 2
2 4
2 5
```

Output:

```
2
1
```



Warning: large Input/Output data, be careful with certain languages

Added by: Tomasz Niedzwiecki
Date: 2005-01-23
Time limit [s]: 42
Source limit [B]: 50000
Resource: DASM Programming League 2004, problemset 6

SPOJ Problem Set

288. Prime or Not

Problem code: PON

Given the number, you are to answer the question: "Is it prime?"

Solutions to this problem can be submitted in C, C++, Pascal, Perl, Python, Ruby, Lisp, Hask, Ocaml, Prolog, Whitespace, Brainfk and Intercal only.**

Input

t – the number of test cases, then t test cases follows. [$t \leq 500$]

Each line contains one integer: N [$2 \leq N \leq 2^{63}-1$]

Output

For each test case output string "YES" if given number is prime and "NO" otherwise.

Example

Input :

```
5
2
3
4
5
6
```

Output :

```
YES
YES
NO
YES
NO
```

Added by: Roman Sol
Date: 2005-01-24
Time limit [s]: 50
Source limit [B]:5000

SPOJ Problem Set

289. The Turing Music Box

Problem code: TMBOX

If you've ever dealt with the theory of information, you are no doubt familiar with the theoretical notion of a *Turing Machine*. But have you ever wondered what you could do if you got a *real* Turing Machine -- one of those big metal things with all the cranks and levers and rolls of infinite tape that looks suspiciously like toilet paper?...

The sad answer is: there are few interesting things that can be done with such a machine. Even problems that have a little charm in the theoretical model (like the intractable Halting Problem) can be solved very efficiently with practical brute-force algorithms (see e.g. the figure at the end of the problem description). But there is one thing that you can do with a practical Turing Machine, and can't do with a theoretical one, and it is: to use it as a music box.

Our Turing Machine has exactly one state variable (an integer in the range 0 to 999) and is equipped with an infinite tape, consisting of cells with symbols from a given alphabet encoded on them. A movable read/write head is positioned over some cell of the tape, and is operated according to the list of rules encoded in the machine. The rules are of the form $S1 \ C1 \ S2 \ C2 \ M$, which means: if the machine is in state $S1$ and $C1$ is written in the current cell, change state to $S2$, write $C2$ in the current cell, and move the head as described by move M (one cell left, one cell right, or not at all). If no matching rule is found for the given state the machine should halt.

Now, here is the good bit. The head makes a creaking sound when performing each rule. It goes *da* when moved right, *di* when moved left, and *um* when left in place. Suppose that each cell of the tape can contain one of 16 possible symbols, formed as the concatenation of exactly two of the words: *da*, *di*, *um* and *sh* for silence. Initially, nearly all the cells of the tape are filled with the symbol *shsh*. Only a few (not more than 500) consecutive cells form a piece of music, each cell encoding a pair of sounds (one of 9 combinations of *da*, *di* or *um*, without any silences). The head of the machine is initially positioned over the leftmost of the cells containing sounds.

Now it is your task to use the Turing machine to play the piece of music written on its tape (as read from left to right, starting from the initial position of the head, as far as the first silence) as accurately as possible, using the head itself to produce the sounds required.

Output

The output of your program must contain a set of rules describing the behaviour of the Turing Machine designed for playing music. Each rule must be of the form $S1 \ C1 \ S2 \ C2 \ M$, where $S1$ and $S2$ are integers from the range 0..999, $C1$ and $C2$ belong to the 16 symbols of the alphabet, while M describes the move direction of the head by the sound it makes (*da*, *di* or *um*).

Score

Your program will be tested multiple times for different pieces of music written on the tape. The score of your program is equal to the total of non-negative scores, taken over all test cases.

For a test case with n notes ($n/2$ non-silent cells) your program will receive $n-d$ points, where d denotes the edit distance between the music played and the music required (i.e. the minimum total number of notes that have to be inserted into or changed in both the pieces to obtain the same piece of music).

Example

Consider the following set of rules output by a program:

```
000 dada 000 dada da
000 umda 000 dada da
000 shsh 000 shsh da
000 didi 001 didi di
001 dada 002 didi di
```

Then the results of exemplary testing could be as follows:

```
Music: da da|da da|da da|di di|um um
Plays: da da da di di
Score: 5
```

```
Music: um da|um da|um da|da um|di di
Plays: da da da
Score: 3
```

Total: $5 + 3 = 8$ points

Bonus info: There are no more than 100 tests. The score format is $s.xxyy$, where xx denotes the number of tests for which your machine played the music perfectly, yy - the number of tests for which it received a positive score.



Added by: Adrian Kosowski
Date: 2005-01-26
Time limit [s]: 42
Source limit [B]: 50000
Resource: DASM Programming League 2004, problemset 6

SPOJ Problem Set

290. Polynomial Equations

Problem code: POLYEQ

You are given the polynomial $F(x)$ as the sum of monomials. Each monomial has the form: $[coefficient]*x^{degree}$ or $[coefficient]$, where *coefficient* and *degree* are integers such that $-30000 \leq coefficient \leq 30000$, $0 \leq degree \leq 6$. The parameters given in $[]$ can be skipped. In this problem you have to find all solutions of the equation: $F(x)=0$.

Input

t – the number of test cases, then t test cases follow. [$t \leq 100$]
Each line contains one polynomial $F(x)$ given as string s in the form described above.
The length of string s is not more than 300 characters.

Output

For each test case output all solutions (including repeated) of the given equation in non-decreasing order. All solutions lie within the interval $[-100.0; 100.0]$. Each solution must be given with an error of not more than 0.01. It's guaranteed that all solutions are real, not complex.

Example

Input:

```
2
x^4-6*x^3+11*x^2-6*x
-x^2+2*x-1
```

Output:

```
0.00 1.00 2.00 3.00
1.00 1.00
```

Added by: Roman Sol
Date: 2005-01-27
Time limit [s]: 30
Source limit [B]:50000

SPOJ Problem Set

291. Cube Root

Problem code: CUBERT

Your task is to calculate the cube root of a given positive integer. We can not remember why exactly we need this, but it has something in common with a princess, a young peasant, kissing and half of a kingdom (a huge one, we can assure you).

Write a program to solve this crucial task.

Input

The input starts with a line containing a single integer $t \leq 20$, the number of test cases. t test cases follow.

The next lines consist of large positive integers of up to 150 decimal digits. Each number is on its own separate line of the input file. The input file may contain empty lines. Numbers can be preceded or followed by whitespaces but no line exceeds 255 characters.

Output

For each number in the input file your program should output a line consisting of two values separated by single space. The second value is the cube root of the given number, truncated (not rounded!) after the 10th decimal place. First value is a checksum of all printed digits of the cube root, calculated as the sum of the printed digits modulo 10.

Example

Input:

```
5
1

      8

    1000
2
```

33076161

Output:

```
1 1.0000000000
2 2.0000000000
1 10.0000000000
0 1.2599210498
6 321.0000000000
```

Added by: Thanh Vy Hua Le
Date: 2005-01-29
Time limit [s]: 10
Source limit [B]:50000
Resource: ACM South Eastern European Region 2004

SPOJ Problem Set

292. Alibaba

Problem code: ALIBB

Alibaba the famous character of our childhood stories would like to be immortal in order to keep bringing happiness to children. In order to reach this status he needs to prove that he is still able to do some unusual things. There are n treasures, ($n \leq 10000$) each in a different place located along a straight road. Each treasure has a time limit, after that it vanishes. Alibaba must take all the n treasures, and he must do it quickly. So he needs to figure out the order in which he should take the treasures before their deadlines starting from the most favorable position. Alibaba has the list of places and deadlines of the treasures. A place i is located at distance d_i from the leftmost end of the road. The time it takes to take a treasure is instantaneous.

Alibaba must find **the smallest time** by which he can take all the treasures.

Input

The first line of the input contains an integer $K \leq 10$ - determining the number of datasets

Each data set in the input stands for a particular set of treasures. For each set of treasures the input contains the number of treasures, and the list of pairs place - deadline in increasing order of the locations. White spaces can occur freely between the numbers in the input. The input data are correct.

Output

For each set of data the program prints the result to the standard output on a separate line. The solution is represented by the smallest time by which Alibaba can take all the treasures before they vanish. If this is not possible then the output is "No solution".

Example

Input :

```
2
5
1 3
3 1
5 8
8 19
10 15
5
1 5
2 1
3 4
4 2
5 3
```

Output :

```
11
No solution
```

Added by: Thanh Vy Hua Le
Date: 2005-01-29
Time limit [s]: 10
Source limit [B]:50000
Resource: ACM South Eastern European Region 2004

SPOJ Problem Set

293. Officers on the Beat

Problem code: OFBEAT

In the Middle Ages the capital of Byteland was surrounded by stout walls to protect the citizens from intruders. The gates of the city were well guarded and the drawbridge was lifted for the night, and everyone felt pretty happy and secure. At least, for a while.

With time the usual disadvantages of a walled city became apparent. As the population increased, crime flourished in the cramped living space. Eventually it all became so bad that the mayor decided to intervene. Some of the guards were reassigned from their usual occupation of reading newspapers in the guard posts near the gates, and told to start patrolling the city. Many of the officers were rather unhappy about all this, especially after the first men to go on the beat returned with bleeding noses and bumps on their heads. Sensing the low morale of the men, the Captain of the Guard, a bright young individual, decided to reinterpret the order he had received from the mayor. He decided that patrol officers would only go out in large groups and armed to the teeth, and would only move along a few carefully chosen streets from which they could see everything that was going on in the city without actually getting involved.

The city is laid out on a regular grid, with each street running North-South or East-West from one end of the city to the other (as far as the walls allow). Every point with integer coordinates is at an intersection of two streets, one leading North-South, the other East-West. The walls that surround the city form a simple polygon whose sides run directly alongside sections of some streets of the city.

Every street in the set of 'patrolled streets' chosen by the Captain intersects with at least one other patrolled street. Furthermore, if a point belongs to one of the streets of the city then it is visible from some point of one of the patrolled street (points see each other iff the line segment connecting them is a fragment of a street). Finally, the set of patrolled streets chosen by the Captain consists of the minimum possible number of streets.

Given a description of the capital of Byteland, find out how many of its streets were actually patrolled by guards after the Captain issued his order.

Input

The first line of input contains t - the number of test cases. t test cases follow.

For each test case, the first line contains a single integer n - the number of sections the city wall consists of ($4 \leq n \leq 2000$). The second line contains exactly n integers a_1, \dots, a_n describing successive sections of wall ($1 \leq |a_i| \leq 100000$). Any two successive sections of wall are perpendicular to each other. The length of the i -th section is the absolute value of a_i , while its direction is described by the sign of a_i (positive means northbound or eastbound, negative - southbound or westbound when traversing the walls clockwise).

Output

For each test case output a single integer k - the number of elements of the patrolled set of streets selected by the Captain.

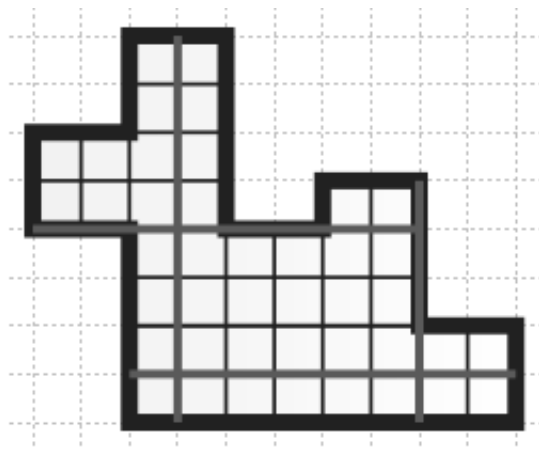
Example

Input:

```
1
14
+2 +2 +2 +2 -4 +2 +1 +2 -3 +2 -2 -8 +4 -2
```

Output:

```
4
```



Added by: Adrian Kosowski
Date: 2005-02-05
Time limit [s]: 42
Source limit [B]: 50000
Resource: DASM Programming League 2004, problemset 6

SPOJ Problem Set

294. Johnny and the Optimisation of Homework

Problem code: **HWORK**

One day when Johnny was still a schoolboy he got caught red-handed by his teacher while doing a Very Mischievous Thing (of the sort that you would expect of Johnny). As a punishment he was told off and assigned additional homework. The teacher underlined quite a few words in a dictionary and asked Johnny to rewrite all of them to his notebook.

Johnny wasn't at all pleased about this, since writing by hand is always a painful burden. Fortunately, Johnny's dad took pity on the crying boy and offered to help. He presented his son with a few sheets of carbon paper, thanks to which any text Johnny wrote was at once ready in exactly k copies. Some of the characters of particular copies could then be erased using a white correction pen, so as to obtain only the words required by the teacher. All the characters forming a single word have to be directly adjacent, but words can be written in any order on the sheets and different words can be separated by an arbitrary (possibly 0) amount of space.

Johnny has cheered up considerably by now, since the bit with the carbon paper and correction pen sounds rather fun. All that remains to be done is to write down an appropriate text, obviously keeping it as short as possible. Please advise Johnny what to write.

Input

Input begins with a single integer t ($t \leq 1000$). t test cases follow.

Each test case starts with a line containing two integers n k , respectively denoting the number of words the teacher has asked Johnny to write and the total number of carbon copies that Johnny creates, including the original ($1 \leq k \leq n \leq 512$). Each of the next n lines contains a word assigned by the teacher - a string of between 4 and 12 characters 'a', 'b', 'c' or 'd'. All words given at input are Bytelandian nouns in common use.

Output

For the i -th test case output a line with the text `case i Y` or `case i N`, specifying whether you wish to solve the given case. Then in the former case print a single line containing the text that should be written by Johnny. Exactly n lines with a single integer on each should follow, the i -th representing the position of the first letter of the i -th word (on the page on which this word eventually appears, before applying the correction pen).

Scoring

The score awarded to your program is the sum of scores taken over all test cases you chose to solve.

For each test case, the score is the difference between the total number of characters of the words provided by the teacher and the number of characters of the text eventually handwritten by Johnny. Programs which receive a negative score for some test case will be considered incorrect.

Example

Input:

```
1
4 2
aaaa
aaaa
aaaa
bbaaa
```

Output:

```
case 1 Y
aaaabbaaaa
1
1
7
5
```

Score:

```
17 - 10 = 7
```

When given in to the teacher, the 2 pages of homework may look as follows:

```
aaaa__aaaa
aaaabbaaa_
```

Added by: Michal Malafiejski

Date: 2005-02-11

Time limit [s]: 42

Source limit [B]:50000

Resource: DASM Programming League 2004, problemset 7

SPOJ Problem Set

295. Bytelandian Origami

Problem code: BRIGAMI

Many of Johnny's school friends have perfected the art of folding a square sheet of paper into beautiful shapes (known as *origami*). Johnny attempted to follow suit, but to his dismay he found that his fingers were a little too clumsy for the task in hand. After spending yet another day creating something especially disastrous (later named "From the series: *Crumpled Pieces of Paper Seen with an Artist's Eye*, No. 27"), Johnny decided he'd had enough. Therefore he proudly proclaimed to all his friends that origami was not fit for serious people, and that he intended to become the master of *kirigami*, the art of cutting paper. But after experimenting with kirigami for a few weeks, he sold the rather miserable results of his labour to the local confetti store, and announced that true beauty lay in convex polygons, and that they were the only shapes a true artist should ever cut. Still, if a person is as lazy and inapt as Johnny, even such a seemingly simple task may turn out a real challenge.

The method Johnny uses to create works of art consists of several steps. First, he takes a sheet of paper in the shape of a convex polygon and uses a ruler and pencil to draw a convex polygon (lying entirely within the sheet). Then, he proceeds to cut it out using a ruler and a razor-edged paper cutter. Every cut is thus a segment of a line, reaching from one edge of the sheet of paper to another, and adjacent to one side of the drawn polygon. Johnny then discards the cut off corner of the sheet and continues cutting until the shape outlined in pencil is completely cut out. Since he is extremely disinclined to perform hard work, please write a program to help him minimise the total length of the lines along which the paper is cut.

Input

Input begins with a single integer t ($t \leq 200$). t test cases follow.

Each test case starts with a line containing two integers m n , denoting the number of vertices of the sheet of paper and the shape drawn on it, respectively ($3 \leq m, n \leq 600$). The next m lines contain two integers a_i b_i each ($-20000 \leq a_i, b_i \leq 20000$), corresponding to the x and y coordinates of vertices of the sheet of paper (given in clockwise order). The description of the shape drawn on the sheet follows, given in the next n lines in a similar form.

Output

For the i -th test case output a line with the text `case i Y` or `case i N`, specifying whether you wish to solve the given case. In the former case, in the next line output a permutation of the numbers $1 \dots n$ denoting the order in which Johnny is supposed to cut out the respective sides of the shape drawn on the sheet (vertices are numbered from 1 to n in the input order, and side s connects vertices s and $1 + s \bmod n$).

Score

The score awarded to your program is the sum of scores taken over all test cases you chose to solve.

For each test case, the score awarded to your program is equal to the ratio of the perimeter of the sheet of paper and the total length of the lines along which the paper is cut.

Example

Input:

```
3
4 4
0 0
0 2
2 2
2 0
0 1
1 2
2 1
1 0
4 3
0 0
0 3
3 3
3 0
1 1
1 2
2 2
4 3
0 0
0 3
3 3
3 0
1 1
1 2
2 2
```

Output:

```
case 1 Y
1 2 3 4
case 2 Y
1 2 3
case 3 Y
3 2 1
```

Score:

```
4.94
```

Added by: Michal Malafiejski

Date: 2005-02-11

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004, problemset 8

SPOJ Problem Set

296. Teamwork is Crucial

Problem code: TWORK

In the late Middle Ages the University of Byteland was no different than any other university of the day. One of those gloomy places where philosophers brooded over the essence of life, theologians did likewise and quaralled with philosophers, while alchemists developed new caustic types of green shampoo in their futile search for gold. The thing that worried the Chancellor most was that none of the staff seemed to be in the least capable of making money in any form. When he complained about this to the Director of Human Resources, the Director came up with a brilliantly simple theory. He claimed that this lack of productivity was the direct consequence of the isolated model of work, and that wonders could be achieved by promoting teamwork.

The Director intends to assign every scientist to some 3-person workgroup. The members of the workgroup should then select which of them is to act as the group leader. And this of course is the root of the problem. Every scientist will tolerate either himself or one of his acquaintances as the leader of his group, but will never allow anyone else to have this privilege. So when creating workgroups it is necessary to bear in mind that every group should have at least one suitable candidate for the role of group leader, accepted by all its members.

Although everyone at the University knows *of* everyone else indirectly (as acquaintances of acquaintances of acquaintances of...), the number of direct acquaintances that every scientist has is relatively small - either equal to 2, or to 3. Even so, it ought to be possible to assign the vast majority of scientists to workgroups. Quite naturally, the dubious pleasure of performing this task has been left to you, the Acting University Algorithmist.

Input

Input starts with a single integer t , the number of test cases ($t \leq 100$). t test cases follow.

Each test case begins with a line containing two integers n m ($4 \leq n \leq m \leq 20000$, n is the number of scientists and is divisible by 4). Exactly m lines follow containing a pair of integers a_i b_i each which denote that scientists a_i and b_i are acquaintances ($1 \leq a_i, b_i \leq n$, each scientist has either 2 or 3 acquaintances). Acquaintanceship is mutual.

Output

For each test case, output a line containing a single integer k - the number of workgroups you have formed. In each of the next k lines output exactly 3 integers, representing the numbers of scientists belonging to respective workgroups.

Your solution will be regarded as incorrect if for some test case more than 25% of all scientists are left without a valid assignment to a workgroup.

Example

Input:

```
1
8 10
1 2
1 3
2 5
4 6
3 7
2 3
5 6
6 7
7 8
8 4
```

Output:

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

Added by: Adrian Kosowski

Date: 2005-02-14

Time limit [s]: 10

Source limit [B]:50000

Resource: DASM Programming League 2004, problemset 7

SPOJ Problem Set

297. Aggressive cows

Problem code: AGGRCOW

Farmer John has built a new long barn, with N ($2 \leq N \leq 100,000$) stalls. The stalls are located along a straight line at positions x_1, \dots, x_N ($0 \leq x_i \leq 1,000,000,000$).

His C ($2 \leq C \leq N$) cows don't like this barn layout and become aggressive towards each other once put into a stall. To prevent the cows from hurting each other, FJ want to assign the cows to the stalls, such that the minimum distance between any two of them is as large as possible. What is the largest minimum distance?

Input

t – the number of test cases, then t test cases follows.

* Line 1: Two space-separated integers: N and C

* Lines 2.. $N+1$: Line $i+1$ contains an integer stall location, x_i

Output

For each test case output one integer: the largest minimum distance.

Example

Input:

```
1
5 3
1
2
8
4
9
```

Output:

```
3
```

Output details:

FJ can put his 3 cows in the stalls at positions 1, 4 and 8, resulting in a minimum distance of 3.

Added by: Roman Sol
Date: 2005-02-16
Time limit [s]: 4
Source limit [B]: 10000
Resource: USACO February 2005 Gold Division

SPOJ Problem Set

298. Closing down Railway Lines

Problem code: DERAIL

Many cities in Byteland look back on the days when Johnny the First was king, and when nobody bothered about public spending. One of things that the citizens liked most about Johnny was that whenever he had a hangover, he would sign any public petition brought forth to him, just for the sake of peace and quiet. Rail travel was extremely popular, and lots of cities and villages requested railway lines connecting them directly, to which King Johnny always graciously agreed (even if he wasn't quite sure what he was agreeing to). Seeing that money was no object, the railway tracks were built in such a way as to connect pairs of cities directly, along straight lines. If two railroads intersected, a complex intersection involving bridges and tunnels was built and everyone seemed perfectly happy.

And then after Johnny's abdication, democracy returned, and the happy days of Byteland ended. One of the first things that had to be done was closing down most of the railway lines. The new government intends to disassemble a large part of the direct railway connections, preserving barely enough to make travel possible between any two cities (perhaps via other cities on the way). The total cost of maintenance of the lines which remain open, equal to k Bytelandian Dollars per kilometer of track open and l Bytelandian Dollars per intersection of 2 used tracks, is to be as low as possible. Please help the government decide which railroads should remain open.

Input

Input starts with a single integer t , the number of test cases ($t \leq 100$). t test cases follow.

Each test case begins with a line containing two integers n m k l , denoting the number of cities, the number of direct connections between cities, the cost of upkeep of a kilometer of track, and the cost of upkeep of a single railway line intersection, respectively ($3 \leq n \leq m \leq 10000$, $0 \leq k, l \leq 100000$). Each of the next n lines contains two integers x_i y_i , corresponding to the X and Y coordinates of the i -th city, measured in kilometers, respectively ($-40000 \leq x_i, y_i \leq 40000$). Then exactly m lines follow, containing a pair of integers a_i b_i each, which denote that cities a_i and b_i , numbered in input order, are connected by a direct railway track ($1 \leq a_i, b_i \leq n$). No three cities are collinear and no three tracks intersect at one point. All tracks are bidirectional.

Output

For the i -th test case output a line containing the text `case i Y` or `case i N`, specifying whether you wish to solve the given case. Then in the former case, write exactly $n-1$ lines containing one integer each -- the numbers of the railway connections that ought to be left open (numbered in input order). It is guaranteed that for the input data some solution always exists.

Score

The score awarded to your program is the sum of scores received for the test cases you chose to solve. For each such test case you will receive $(s/c)-1$ points, where s is the cost of maintenance of the original configuration, while c is the cost of maintenance of only those railway lines which you've

selected.

Example

Input:

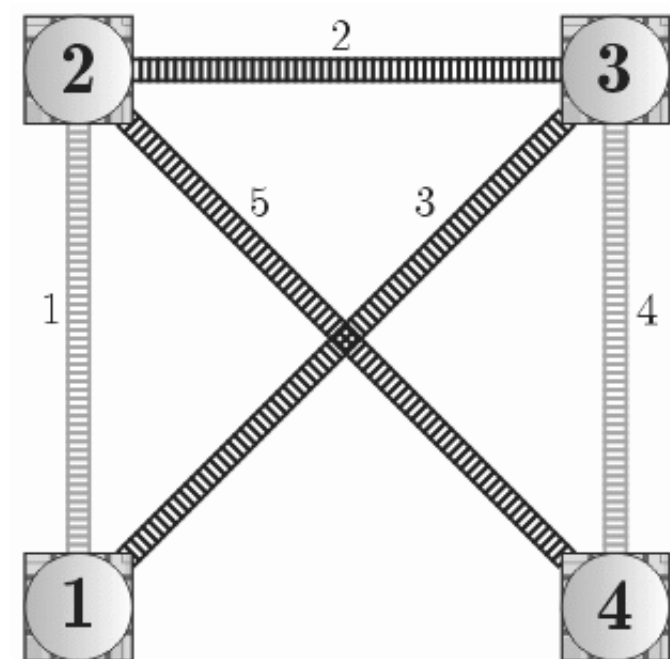
```
1
4 5 1 100
0 0
0 1
1 1
1 0
1 2
2 3
1 3
3 4
4 2
```

Output:

```
case 1 Y
3
2
5
```

Score:

$(100+1+1+1+1.414+1.414) / (100+1+1.414+1.414) - 1 = 0.019$



Added by: Adrian Kosowski

Date: 2005-02-19

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004, problemset 7

SPOJ Problem Set

300. Cable TV Network

Problem code: CABLETV

The interconnection of the relays in a cable TV network is bi-directional. The network is connected if there is at least one interconnection path between each pair of relays present in the network. Otherwise the network is disconnected. An empty network or a network with a single relay is considered connected. The safety factor **f** of a network with **n** relays is:

1. **n**, if the net remains connected regardless the number of relays removed from the net.
2. The minimal number of relays that disconnect the network when removed.

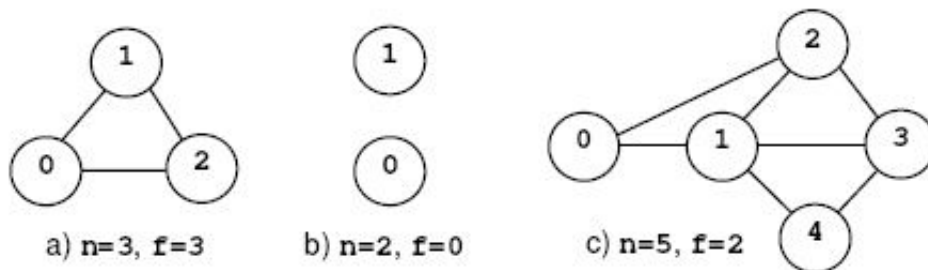


Figure 1. Cable TV networks

For example, consider the nets from figure 1, where the circles mark the relays and the solid lines correspond to interconnection cables. The network (a) is connected regardless the number of relays that are removed and, according to rule (1), $f=n=3$. The network (b) is disconnected when 0 relays are removed, hence $f=0$ by rule (2). The network (c) is disconnected when the relays 1 and 2 or 1 and 3 are removed. The safety factor is 2.

Input

The input starts with a line containing a single integer $t \leq 20$, the number of test cases. t test cases follow.

Write a program that computes the safety factor for the cable networks encoded by the data sets. Each data set starts with two integers: $0 \leq n \leq 50$, the number of relays in the net, and m , the number of cables in the net. Follow m data pairs (u,v) , $u < v$, where u and v are relay identifiers (integers in the range $0..n-1$). The pair (u,v) designates the cable that interconnects the relays u and v . The pairs may occur in any order. Except the (u,v) pairs, which do not contain white spaces, white spaces can occur freely in input. Input data terminate with an end of file and are correct.

Output

For each data set, prints from the beginning of a line, the safety factor of the encoded net.

Example

Input:

```
5
0 0
1 0
3 3 (0,1) (0,2) (1,2)
2 0
5 7 (0,1) (0,2) (1,3) (1,2) (1,4) (2,3) (3,4)
```

Output:

```
0
1
3
0
2
```

Added by: Thanh Vy Hua Le

Date: 2005-02-27

Time limit [s]: 10

Source limit [B]: 50000

Resource: ACM South Eastern European Region 2004

SPOJ Problem Set

301. Booklets

Problem code: BOOK

Bob has a difficult job. He must distribute advertising booklets for extra school activities in different schools. The booklets have different number of pages. Bob has a list with the number of pages of each booklet and the number of schools that he must visit. He has to distribute the booklets such that each school gets a number of booklets equal to either the lower integer part (LIP), or the upper integer part (UIP) of the number of booklets divided by the number of schools. Poor Bob must obey other rules too. He must distribute all the **UIP** number of booklets first and then the **LIP** number of booklets.

Any booklet **A** that is distributed to a school **S_i** must have fewer or at most an equal number of pages that any other booklet **B** that is distributed to a school **S_j**, if **S_i** gets the booklets before **S_j** (i.e if **i < j** then **pages(A) <= pages(B)**). When Bob distributes the booklets to a school he must distribute them in the same relative order in which they are on his list.

Moreover, he must distribute them very fast. When he comes back to the advertising company his boss verifies if he accomplished well his task, by asking him the number of pages of the first booklet distributed to a specific school, following the order in which Bob visited the schools (starting with 0). Difficult job, isn't it? Can you help him?

Input

The input starts with a line containing a single integer **t** ≤ 20 , the number of test cases. **t** test cases follow.

Each data set in the input stands for a particular set of booklets. For each set of booklets the input contains the number of schools, the school specified by Bob's boss, the number of booklets (**less than 3000**), the number of pages of each booklet (fits in integer). White spaces can occur freely between the numbers in the input. The input data are correct.

Output

For each set of data the program prints the result to the standard output on a separate line. The solution is represented by the number of pages of the first booklet distributed to the specified school.

Example

Input:

```
1
3
2
7
3 5 9 1 11 14 2
```

Output:

```
11
```

Added by: Thanh Vy Hua Le
Date: 2005-02-27
Time limit [s]: 10
Source limit [B]:50000
Resource: ACM South Eastern European Region 2004

SPOJ Problem Set

302. Count on Canton

Problem code: CANTON

One of the famous proofs of modern mathematics is Georg Cantor's demonstration that the set of rational numbers is enumerable. The proof works by using an explicit enumeration of rational numbers as shown in the diagram below.

```
1/1 1/2 1/3 1/4 1/5 ...
2/1 2/2 2/3 2/4
3/1 3/2 3/3
4/1 4/2
5/1
```

In the above diagram, the first term is $1/1$, the second term is $1/2$, the third term is $2/1$, the fourth term is $3/1$, the fifth term is $2/2$, and so on.

Input

The input starts with a line containing a single integer $t \leq 20$, the number of test cases. t test cases follow.

Then, it contains a single number per line.

Output

You are to write a program that will read a list of numbers in the range from 1 to 10^7 and will print for each number the corresponding term in Cantor's enumeration as given below.

Example

Input :

```
3
3
14
7
```

Output :

```
TERM 3 IS 2/1
TERM 14 IS 2/4
TERM 7 IS 1/4
```

Added by: Thanh Vy Hua Le
Date: 2005-02-27
Time limit [s]: 10
Source limit [B]: 50000
Resource: ACM South Eastern European Region 2004

SPOJ Problem Set

303. The Unstable Cube

Problem code: UCUBE

A large cube (of size $N \times N \times N$) is given. At the beginning it consists of small blocks ($1 \times 1 \times 1$) and each block is painted in some color (different blocks may have the same color). But in the process of exploitation some blocks have disappeared. Given 6 photos of the unstable cube you have to calculate the maximum possible number of blocks that still remain in the unstable cube. It is possible that the unstable cube consists of more than one part.

Input

t – the number of test cases, then t test cases follow.

N - size of the big cube [$1 \leq N \leq 10$]

In the next N lines views of the cube from 6 sides are described (in the following order: from the front, left, back, right, from above, from below). Each such view is represented by a table of size $N \times N$ in which different letters denote different colors, and the symbol "." (point) means that it is possible to see all the way through the cube at this point. Consecutive views are separated by exactly one space.

The bottom border of the top view corresponds to the top border of the front view, and the top border of the bottom view - to the bottom border of the front view. For the front, back, left and right views the top and bottom sides of a view correspond to the top and bottom of the cube.

The input file is correct, i.e. each test case describes a possible configuration.

Output

For each test case output one integer: the required maximum number of blocks remaining in the unstable cube.

Example

Input :

```
2
3
.R. YYR .Y. RYY .Y. .R.
GRB YGR BYG RBY GYB GRB
.R. YRR .Y. RRY .R. .Y.
2
ZZ ZZ ZZ ZZ ZZ ZZ
ZZ ZZ ZZ ZZ ZZ ZZ
```

Output :

```
11
8
```

Added by: Roman Sol
Date: 2005-03-01
Time limit [s]: 1
Source limit [B]:20000
Resource: The Moscow Olympiad on computer science 2004/05. Correspondence round.

SPOJ Problem Set

309. The Room Pattern

Problem code: RATTERN

It was decided to make a parquet floor in a room of size $N \times M$. The idea is to lay out some pattern on the floor. The parquet tiles with which the floor of the room looks best consist of squares 1×1 , each of which can be either white or black. The required color of each square of the room is specified on the map of the room.

There are four different forms of parquet tiles:

1		1	2		1	2	3		1	2
										3
Form 1		Form 2			Form 3				Form 4	

Squares of one parquet tile can be painted differently. Some types of tiles can be of identical shape, but painted differently. Tiles of different types can have different cost. The number of available tiles of each type is not limited. Tiles are allowed to be turned around somehow (by an angle which is a multiple of 90 degrees), but it is not permitted to break a tile or to put it face sheet downwards. Initially, any part of the floor can be already laid out by tiles. You are requested to calculate the minimal cost of the tiles necessary to pave the remaining part of the room.

Input

t – the number of test cases, then t test cases follow.

In the first line of each test case three numbers are written: N , M (the sizes of the room) and K (number of accessible types of tiles). $[1 \leq N, M \leq 8]$, $[1 \leq K \leq 10]$. Next there is a description of the desired painting of the floor. The description is given in the form of N lines of M numbers each, where 0 denotes the color white, 1 - the color black, 2 - a square which has already been covered by a tile. In the last K lines the descriptions of available types of tiles are given in the following format:

[Form] [cost] [painting] where:

[Form] is a number from 1 to 4, describing the form of a tile (see figure above)

[Cost] is an integer not larger than 10000, describing the cost of one tile of the type.

[Painting] is a sequence of between one and three numbers 0 or 1. Its length is the same as the number of squares of which the tile consists, and the respective numbers describe colors of square tiles in the order in which the squares are numbered in the figure.

Output

For each test case output one integer: the minimal cost of laying the remaining part of the parquet, or -1 if the task cannot be performed.

Example

Input:

```
1
4 3 3
2 2 2
2 0 0
2 1 2
2 2 2
2 10 0 0
1 5 1
4 6 0 0 1
```

Output:

```
15
```

Added by: Roman Sol

Date: 2005-03-05

Time limit [s]: 40

Source limit [B]: 20000

Resource: The Moscow Olympiad on computer science 2004/05. Correspondence round.

SPOJ Problem Set

313. The Game of Crosses & Crosses

Problem code: CROSSES

The game of gomoku (otherwise known as naughts & crosses), played on an $n \times n$ board has many interesting variations. One of them is the Game of Crosses & Crosses, with the following set of rules:

- Two players - red and black - take it in turns to place one cross of their respective color on an unoccupied square of the $n \times n$ gaming board. Red starts the game.
- After each player's move any rectangles with sides equal to at least 2, lying entirely within the gaming board and covered completely by crosses, are simultaneously removed (cut off) from the gaming board and the game continues.
- When all the squares remaining in the gaming board are covered by crosses, the game comes to an end. The score of each player is equal to the number of crosses of his color left standing on the gaming board, and the player with the higher score is considered the winner.

The game of crosses & crosses feels rather like playing a degenerated game of Go with an army of suicide bombers. For many years now it has been the favourite passtime of Bytelandian schoolchildren during their lessons. Little Johnny was no different, and among his friends he actually became a notable crossing champion.

But not many people knew about Johnny's crossing talent, and Johnny often used this to his advantage. So when a few years after Johnny's abdication from the throne of Byteland an unsuspecting publisher signed a million dolar contract with the ex-king for a series of memoirs entitled *The famous victories of Johnny the Great*, he was certainly not prepared for what he received -- a detailed account of Johnny's childhood games of crosses & crosses. To make matters worse, all accounts are written by Johnny in exciting prose, rich in action, e.g.: "Then I played yet another game on a 3×3 board. I placed my first cross at (1,1). Then I placed a cross at (2,3). The next cross I placed at (2,2). The cross after that I placed at (3,3). Finally, I placed a cross at (1,2) and I won the game 2:1."

In a desperate effort to save the day, the publisher employed you to create illustrations for the book. You are given a free hand in reinacting the games (and in particular the oponent's moves, which Johnny has modestly left out), provided your version of events is not an evident contradiction of Johnny's text.

Input

Input begins with a line containing a single integer t ($t=100$). t test cases follow.

Each test case starts with a line with three integers describing a single game: n sr sb , denoting the length of the side of the playing board, the number of points scored by the red player (Johnny) and the number of points scored by the black player (Johnny's oponent), respectively ($3 \leq n \leq 250$, $0 \leq sb < sr$). The next $\text{ceil}(n^2/2)$ lines contain 2 integers x_i y_i each - the coordinates of the squares where Johnny placed his crosses in successive moves ($1 \leq x_i, y_i \leq n$).

Output

For the i -th test case output a line with the text `case i Y` or `case i N`, specifying whether you wish to solve the given case. Then in the former case print exactly $\text{floor}(n^2/2)$ lines containing 2 integers each - the coordinates of the squares where Johnny's anonymous oponent placed his crosses in successive moves.

Scoring

The score awarded to your program is equal to the number of correctly solved test cases. For each case, the game defined by yours and Johnny's description must have the outcome (final score) defined at input.

Example

Input:

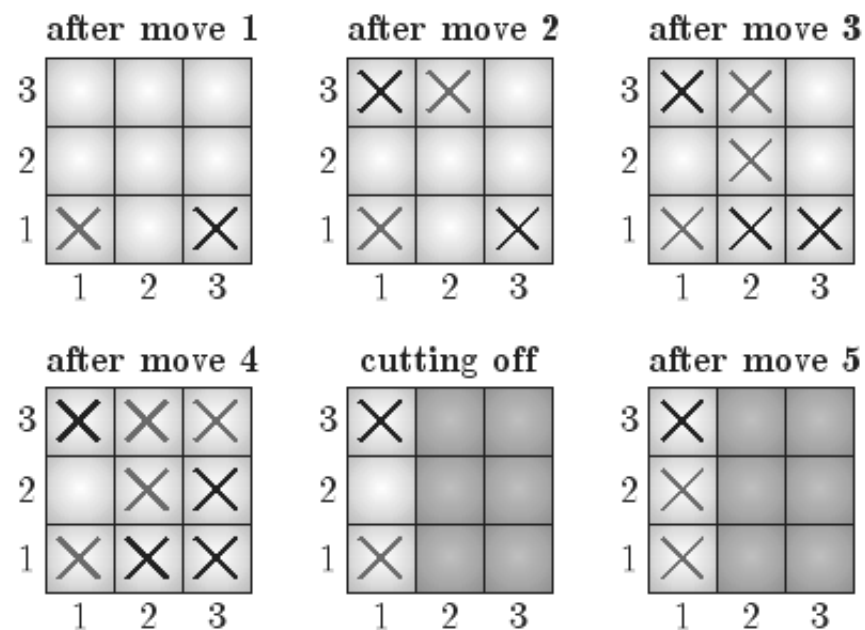
```
1
3 2 1
1 1
2 3
2 2
3 3
1 2
```

Output:

```
case 1 Y
3 1
1 3
2 1
3 2
```

Score:

```
1
```



Warning: large Input/Output data, be careful with certain languages

Added by: Adrian Kosowski

Date: 2005-03-09

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004, problemset 8

SPOJ Problem Set

314. Digits of e

Problem code: EVAL

In this problem you have to find as many digits of E as possible.

Input

There is no input for this problem

Output

Output must contain as many digits of E as possible (max = 1000000)

Score

The score awarded to your program will be the first position of digit where first difference occurred.

Example

Output:

2.7182

will be awarded with 6 points.

Added by: Roman Sol
Date: 2005-03-10
Time limit [s]: 60
Source limit [B]: 4096

SPOJ Problem Set

315. The Secret Fellowship of Byteland

Problem code: BFORG

The relationship between The University of Byteland and King Johnny was never a friendly one. The king was the easy-going, open-minded sort of person who is prepared to turn a blind eye to the embezzlement of public funds, but inwardly revolts at the thought of money going to waste, and supporting a university was to the king a perfect example of a waste of money. On the other hand, the chancellor of the university showed no tolerance whatsoever, and frequently stated in public that Byteland was being governed by a monarch who took terrible decisions when he was drunk and even worse ones when he was sober. After some time of bad-tempered coexistence, the king had had enough and decided to close down the university. However, the king's councillors advised against this move, suggesting it might cause social unrest. The king yielded to their advice, and instead established a law which banned all organisations, clubs and associations active at the university.

This action had a rather curious effect on the usually lazy students of the university. They had never before even thought of organising any sort of fellowship, but now they immediately decided they needed to set one up. And this is how the *Secret Fellowship* came to life.

The main problem that faced the management of the Fellowship was to organise members' meetings in such a way as to minimise the risk to the participants. It was decided that the n members of the fellowship should be split into k secret divisions, each consisting of at least 2 members. All members belonging to the same division would then meet regularly, and they would take it in turns to host the meetings of the division in their houses.

But one more important factor has to be taken into account -- the laziness of students. It is therefore your task to form the divisions in such a way that the furthest distance a student may ever be asked to walk is as short as possible.

Input

The first line of input contains a single integer t , the number of test cases ($t \leq 1000$). t test cases follow.

Each test case starts with a line containing two integers n k , denoting the number of students and the number of divisions to be formed, respectively ($2 \leq 2k \leq n \leq 200$). Each of the next n lines contains two integers x_i y_i each ($-1000 \leq x_i, y_i \leq 1000$), denoting the coordinates of the houses of successive students.

Output

For the i -th test case output a line with the text `case i Y` or `case i N`, specifying whether you wish to solve the given case. Then in the former case print exactly k lines. Each line should start with integer n_j ($n_j \geq 2$) and be followed by a space separated list of exactly n_j increasing integers s_{jl} , denoting the students belonging to the j -th division, numbered in input order ($1 \leq s_{jl} \leq n$). All divisions must be disjoint and the sum of all numbers n_j must equal n .

Score

The score awarded to your program is the total of scores for the test cases you chose to solve.

For each solved test case you will receive $diam / (d * k)$ points, where $diam$ denotes the distance between the two furthest houses of members of the fellowship, and d is the distance between the two furthest houses of members belonging to the same division.

Example

Input :

```
2
6 3
0 0
1 0
0 1
1 1
2 0
2 1
6 2
0 0
1 0
0 1
1 1
2 0
2 1
6 2
0 0
1 0
0 1
1 1
2 0
2 1
```

Output :

```
case 1 Y
3 1 2 4
3 3 5 6
case 2 Y
3 1 2 5
3 3 4 6
case 3 Y
2 1 3
4 2 4 5 6
```

Score :

1.849003

Bonus info: If score = *xxx.xxxaaa*, *aaa* means the number of test cases with Y answer.

Added by: Michal Malafiejski

Date: 2005-03-11

Time limit [s]: 42

Source limit [B]:50000

Resource: DASM Programming League 2004, problemset 9

SPOJ Problem Set

316. Japan Crossword

Problem code: JCROSS

Japan crossword is a very popular game. It represents encoded picture which consists of filled block of cells. At the start of game you see empty grid. Each row (column) has some numbers in beginning of the row (column). Each number means how many continuous cells are filled in a hidden picture (length of the filled blocks). Filled blocks of cells are arranged from left to right and from top to bottom. Between filled blocks must be at least one empty cell. For example, numbers are 4, 2, 7 mean that there are three groups with 4, 2, and 7 filled cells in it. Your task is decode hidden picture using hints.

						3
				1	5	1
		3	5	8	3	1
	3					
2	2					
	5					
	5					
	3					
	1					
	1					
	3					
	2					
	3					

						3
				1	5	1
		3	5	8	3	1
	3					
2	2					
	5					
	5					
	3					
	1					
	1					
	3					
	2					
	3					

Input

The first line of input contains a single positive integer $t \leq 300$ - the number of test cases. Then for every test case first line specifies integer numbers R and C (number of rows and columns) of the picture ($1 \leq R \leq 50$, $1 \leq C \leq 100$). Below R lines are follow. Each line consists of any integers for horizontal hints. The very last number for every line is 0. Then C lines are follow. Each line consists of any integers for vertical hints. And again every line ends with 0.

Output

For every test case you should write decoded picture in the form of rectangle with R rows and C characters in each line. Symbol '#'(sharp) means filled block and symbol '.'(point) means empty cell.

Score

The score awarded to your program is the total of all scores obtained for its individual test cases. The score for a test case is calculated so that for each 'right' row or column you get 1 points. The row(column) is counted as a 'right' if there is a group of filled cells for every number in beginning of the row(or column) and length of every cell is equal corresponded number. If All rows and columns are 'right' your score multiply by 1.5 for this test case.

Example

Input:

```
1
10 5
3 0
2 2 0
5 0
5 0
3 0
1 0
1 0
3 0
2 0
3 0
3 0
5 0
1 8 0
5 3 0
3 1 1 0
```

Output:

```
.###.
##.##
#####
#####
.###.
..#..
..#..
..###
..##.
..###
```

Score:

```
(10+5)*1.5 = 22.500
```

Bonus info: If score = *xxx.xxxaaa*, *aaa* means the number of entirely correct test cases

Added by: Maxim A. Sukhov

Date: 2005-02-07

Time limit [s]: 50

Source limit [B]:50000

Resource: ;)

SPOJ Problem Set

317. Simple Image Recognition

Problem code: IMGREC1

One of the hard problems that borrows human minds and can find the practical application in creating Artificial Intelligence is problem of Image Recognition. This problem in its simplest form can be applied in many spheres of manufactures. In given problem we interest in one elementary case of Image Recognition. You have to make choice form only two possible images that are represented on a bicoloured picture. This images is "dagger" or "zero". This images can be rotated, deformed, scaled, moved, have some noise or different width of lines on the picture. But human always can correctly define that is represented on a picture.

Input

t – number of test cases, than t test cases follows. [$t \leq 100$]

[empty line]

Each test case starts with integer N equals to number of pictures in this test, than N pictures follows.

[$4 \leq N \leq 10$]

[empty line]

Description of each picture starts from two integers H and W - height and width of picture accordingly. [$5 \leq H, W \leq 50$]

than follows exactly H lines each consists of W chars.

Description of picture consists of two symbols only: 'x' - painted square and '.' - empty square. You can be assured, that no other symbols are present at the description of a picture.

Output

For each test it is necessary to deduce on a separate line a string of chars with length equals to N . The string should consist of a set of two chars 'x' and '0'. Where 'x' corresponds to a dagger on a picture, and '0' corresponds to a zero. If answer will contains other chars or length of a string won't equals to N you will receive status "Wrong Answer".

Score

The score awarded to your program is the sum of scores for individual test cases. The score for individual correctly solved test equals to N (Number of pictures in this test).

Example

Input:

1

5

5 5

x . . . x

. x . x .

```

..x..
.x.x.
x...x
5 5
xxxxx
x...x
x...x
x...x
xxxxx
6 6
..x...
..x...
xxxxxxx
..x...
..x...
.....
5 5
.xxx.
x...x
x...x
x...x
.xxx.
5 5
.xxx.
.x.x
.xxx.
.....
.....

```

Output:

x0x00

Output:

You will receive 5 points for this solution

Added by: Roman Sol
Date: 2005-02-09
Time limit [s]: 50
Source limit [B]:50000
Resource: ;)

SPOJ Problem Set

318. Pythagorean Legacy

Problem code: PITPAIR

It is necessary to find a minimal integer value R which is equal to the length of the hypotenuse (the side opposite the right angle) of N non-identical rectangular triangles with integer lengths of sides.

Input

t - number of test cases [$t \leq 100$], then t lines follow, each line contains one integer - N , equal to the required number of different rectangular triangles. [$1 \leq N \leq 2000$]

Output

For each test case your program should output a number R in a separate line (R fits in a 64-bit integer), equal to the minimal integer value of a hypotenuse for which exactly N different rectangular triangles can be constructed; then in separate lines follow exactly N numbers equal to the shorter cathetus (side adjacent to the right angle) of each of the rectangular triangles, in ascending order.

Example

Input:

```
2
1
2
```

Output:

```
5
3
25
7
15
```

Added by: Roman Sol
Date: 2005-03-01
Time limit [s]: 20
Source limit [B]: 8192
Resource: Based on a problem from Bitwise 2005

SPOJ Problem Set

321. X-Words

Problem code: XWORDS

It is quite simple really: I'll give you a list of words and you use them to make a crossword puzzle in a 16x32 grid. You'll be able to use the words more than once in the grid and there is a special "flipper" square you can use as a wild card. The winner will be the program that can create the "best" fully connected crossword in one minute. The original problem appeared here: Programmer of the month contest (Feb. 2005).

The Starting Grid

- The grid will consist of 32 columns and 16 rows

The Word List

- There will be at least one word and fewer than 512 words in the wordlist
- Each word will be two letters long or more ($\text{WORDLENGTH} \geq 2$)
- Each word will be sixteen letters long or less ($\text{WORDLENGTH} \leq 16$)
- Words in the wordlist will contain only letters "A" through "Z" in upper case letters with no white space
- Words will appear in the wordlist with one word per line
- Words will not be repeated in the wordlist, but they may be used multiple times in your solution
- Do not assume anything (like sorting) about arrangement of the words in the list
- Do not assume anything about whether a "word" is contained in any dictionary: POTM, ABCDEFG, and XYZZY are all possible "words"
- Words in the list may be subsets of one another: SCAT, CAT, CATS and XCATS may all appear in the same wordlist ... there is no bonus for using words containing other words ... see scoring note below
- There may be words in the wordlist which are not possible to connect to any other words in the wordlist

Placement of the Words onto the Grid

- Any word from the word list may be used in your solution as many times as you wish
- You may use any subset of words in the wordlist, or all of them
- All words placed on the grid must read left-to-right or downwards
- All words placed on the grid must be connected to one another
- ONLY words on the wordlist may be used and empty squares or grid boundaries must be used immediately before and after all words
- Words may not "wrap-around" the grid boundaries in any sense
- Your solution does not need to be symmetric in any sense
- Output which is not connected, or contains words which do not appear in the wordlist, will receive a score of zero

The Flipper Square

- There is one (and only one) "flipper" square (denoted by an asterisk) permitted in your output
- You may place the flipper square within any word you place on the grid
- When used, it may represent a different letter in the horizontal and vertical words of which it is a part
- Any words formed using the "flipper" square must be part of the wordlist (if C*T is placed on the grid, then there must be a three letter word in the wordlist that begins with "C" and ends in "T")
- The "flipper" will likely be used at a word intersection, although this is not required (why would you use it elsewhere??)

Input

t – number of test cases [$t \leq 10$]

N - number of words for given test case, then N lines follows each line contain one word, in upper case. Word will contain no whitespace or characters other than [A-Z].

Output

For each testcase your output must contain exactly 16 lines with 32 characters followed by a line feed as in `printf("\n")` on each line. The letters in your output must be upper case [A-Z] as in the wordlist. The "Flipper" (if used) in your output should be an asterisk "*". Squares that do not contain a letter or a flipper should contain an underbar "_". There should be no white space in your output. Your output must be exactly $t \cdot 528$ bytes.

Score

Your "SCORE" will be the total number of letters in all the words used in your solution. If a word is contained within a longer word, only the longer word will contribute to the score ... for example, using POTM would not score for the word POT even if both are in the wordlist. The "flipper" square contributes to the word length as though it was a part of each word. The total score will be the sum of scores for individual test cases.

Example

Input :

```
1
28
NECESSARY
POLITICAL
CONNECTED
SEPARATE
OPINIONS
REQUIRES
SEPARATION
SELFEVIDENT
UNALIENABLE
HAPPINESS
GOVERNMENTS
INSTITUTED
DERIVING
GOVERNMENT
DESTRUCTIVE
INSTITUTE
FOUNDATION
PRINCIPLES
ORGANIZING
ESTABLISHED
TRANSIENT
ACCORDINGLY
EXPERIENCE
SUFFERABLE
THEMSELVES
ABOLISHING
ACCUSTOMED
USURPATIONS
```

Output:

CONNECTED__USURPATIONS_CONNECTED
O_E__R__R_U__R_P__O_E__R__
NECESSARY_E_FOUNDATION_NECESSARY
N_E__N__Q_F__N_N__N_E_U_N__
E_S_INSTITUTED_INSTITUTE_S_F_S__
C_S_N_I__I_R__I_O__C_S_F_I_E
TRANSIENT_R_A_GOVERNMENT_A_E_E_S
E_R_T_N_H_E_B__N_S_X_E_R_R_N_T
D_Y_I_THEMSELVES_T__P_D_Y_A_T_A
__T__M__E__E__E__B__B
HAPP*NESS__P__PRINCIPLES_L
__T__E_SEPARATION_I__E__I
SUFFERABLE__R__E__S
__D__V_SEPARATION_NECESSARY_H
__E__T__C__E
HAPPINESS_THEMSELVES_ESTABLISHED

Score:

341

Added by: Roman Sol
Date: 2005-04-08
Time limit [s]: 60
Source limit [B]:50000
Resource: Programmer of the Month 02.2005

SPOJ Problem Set

325. The Tall Windmills

Problem code: WINDMILL

In the later days of his career Johnny purchased a long and narrow strip of land on which he intended to erect a row of windmills, and live off the electrical energy produced by his little power plant. To his dismay, he soon discovered that he had been badly cheated - throughout most of the year the wind blew lengthwise through the strip, rather than in a perpendicular direction. As a result, the wind was certain to lose most of its force on the first windmill it encountered, leaving all the others idle. Johnny could only see one way of coping with this problem, namely - to vary the height of windmills situated relatively close to each other. More precisely, Johnny intends to build exactly n windmills along a straight line, with equal spacing (of one Bytelandian furlong) between adjacent windmills. It has been established by a team of experts that if two windmills are k Bytelandian furlongs apart from each other, their height must differ by at least $n-k$ Bytelandian yards. No windmill may ever be lower than 1 Bytelandian yard, and some, obviously, may need to be considerably higher. But tall windmills are far more expensive to construct, and thus you have been asked to choose the heights of Johnny's windmills in such a way as to guarantee that the tallest windmill has the minimum possible height.

Input

Input starts with a single integer t , the number of test cases ($t \leq 100$). t test cases follow.

Each test case consists of exactly one integer n ($1 \leq n \leq 100$) - the number of windmills Johnny intends to construct.

Output

For each test case output a line with exactly n numbers, denoting the heights of successive windmills given in the order in which they are arranged along the road.

Example

Input :

```
3
1
2
3
```

Output :

```
1
1 2
2 4 1
```

Added by: Adrian Kosowski

Date: 2005-04-13

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004, problemset 9

SPOJ Problem Set

326. Enjoying a Multiplayer Game

Problem code: MGAME

One of the most popular types of computer multiplayer games in existence is the simple deathmatch shooter, in which it is the player's task to eliminate all other players on the gaming board. Usually, at the start of the game the players are distributed fairly randomly over the board, and run around in order to find and shoot opponents.

But there is a fair percentage of players (especially the younger ones) who enjoy the shooting most and give up the running altogether. To achieve this, at the start of the match all players are arranged very close to each other, and everyone opens fire in the very first second of the game. The gunfire continues until everyone within sight of everyone else is dead, and then the game ends, since no one feels like moving from their selected camping point.

Parents are often helpless when their children get addicted to this sort of entertainment, and don't know how to make them stop playing without causing a major quarrel. But Johnny's dad has developed the perfect method. He always says to his son: *Sure, I'll let you play another round, but tell me please how long it'll take!* And no, the answer *only a minute or two* is just not good enough.

At the start of the game, the players are positioned on the board and each player has a list of other players he is capable of eliminating (from his location). At the start of every second, each living player fires a round towards one of the opponents on his list (provided the list is not yet empty). Players who have been hit are eliminated from the game directly after the shots were fired. The situation continues until the lists of all surviving players are empty.

We are not asking you to give an exact answer the question posed by Johnny's dad, but only for an honest estimate. Given an arrangement of players on the board, try to find scenarios of shooting leading to the longest possible and the shortest possible game.

Input

The first line of input contains a single integer t , the number of test cases ($t=100$). t test cases follow. Each test case starts with a line containing integer n , denoting the number of players on the board ($2 \leq n \leq 500$). Each of the next n lines contains a list of integers: first, d_i the length of the i -th player's list, followed by the considered list of exactly d_i other players (numbered in input order from the range 1 to n).

Output

For the i -th test case output a line with the text `case i Y` or `case i N`, specifying whether you wish to solve the given case. Then in the former case print a description of the longest known game scenario, followed by a description of the shortest known game scenario. Each scenario starts with an integer t , the duration of the game measured in seconds ($0 \leq t \leq n-1$). Each of the next t lines contains a list of integers, representing the identifiers of players eliminated by respective players in the given second (one integer for each player left alive and capable of hitting an enemy at the start of the second, ordered according to the input identifiers of the shooting players).

Score

The score awarded to your program is the total of scores for the test cases you chose to solve.

For each solved test case you will receive $t_{max} / t_{min} - 1$ points, where t_{max} is the length of the first presented scenario, while t_{min} - the length of the second one.

Example

Input:

```
1
4
2 2 3
2 1 3
3 1 2 4
1 3
```

Output:

```
case 1 Y
2
3 3 4 3
2 1
1
2 1 4 3
```

Score:

```
2/1 - 1 = 1.00
```

Added by: Adrian Kosowski

Date: 2005-04-14

Time limit [s]: 42

Source limit [B]: 50000

Resource: DASM Programming League 2004, problemset 9

SPOJ Problem Set

328. Bishops

Problem code: BISHOPS

Yesterday was Sam's birthday. The most interesting gift was definitely the chessboard. Sam quickly learned the rules of chess and defeated his father, all his friends, his little sister, and now no one wants to play with him any more.

So he decided to play with another birthday gift – a Book of Math Problems for Young Mathematicians. He opened the book somewhere in the middle and read the following problem: "How many knights can be placed on a chessboard without threatening each other?" After a while he realized that this was trivial and moved on to the next problem: "How many bishops can be placed on a chessboard without threatening each other?". Sam is in trouble here. He is not able to solve this problem and needs your help.

Sam's chessboard has size $N \times N$. A bishop can move to any distance in any of the four diagonal directions. A bishop threatens another bishop if it can move to the other bishop's position. Your task is to compute the maximum number of bishops that can be placed on a chessboard in such a way that no two bishops threaten each other.

Input

The input file consists of several lines. The line number i contains a single number N representing the size of the i -th chessboard. [$N \leq 10^{100}$]

Output

The output file should contain the same number of lines as the input file. The i -th line should contain one number – the maximum number of bishops that can be placed on i -th chessboard without threatening each other.

Example

Input :

2
3

Output :

2
4

Added by: Roman Sol
Date: 2005-04-17
Time limit [s]: 1
Source limit [B]: 10000
Resource: IPSC 2004

SPOJ Problem Set

329. Calls

Problem code: CALLS

A young archeologist Senoj Anaidni recently made a very important discovery which will make him famous (or at least he thinks so). He found several scraps of paper resembling advertisement flyers of an ancient phone company. His research showed that modern phone companies follow a few basic rules to compose their flyers (and there is no reason to assume that old companies were an exception).

Each company operates certain number of phone lines. Each phone line connects a pair of cities, and it can be used in both directions. The cost of using each line is a fixed positive number. A call from city A to city B may be routed through one or more other cities, in which case the cost of the call is the sum of the costs of all lines used. (In fact, sometimes it is cheaper to route the call through several other cities than to use the direct connection, even if there exists one.)

To make the information comprehensible to the customer, the phone company lists the cost of the cheapest possible call between every pair of cities serviced by the company. To impress the customer even more, the company also lists the number of lines it operates.

Indeed, each of Senoj's ancient flyers start like this: "Using our 47 telephone lines, we serve 10 most important cities of the world! A call from Sparta to Troja costs 12 dennario, Sparta to Athens is 15 dennario, ...". The list of all pairs of cities and the respective costs of the cheapest possible call between them follows.

This supports Senoj's hypothesis about the origin of the papers, but he is not sure whether they are really genuine. Other archeologists often play dirty jokes on him by making ridiculous forgeries in a hope, that he will make a fool of himself. Luckily, they are often not very meticulous, so we can safely assume, that a flyer is a forgery if and only if it could not have been published by any phone company.

Input

The first line of the input file gives the number t of flyers found by Senoj. [$t \leq 50$] Each flyer is described in a separate block starting with a line containing two integers - N and K - where N is the number of cities and K is the number of phone lines. [$N \leq 300$ $K \leq 1200$] The block continues with $N-1$ lines giving the costs of the cheapest calls between all pairs of cities. In particular, the i -th line contains $(N-i)$ numbers, where j -th number represents the cost of a call between the cities i and $(i+j)$.

Output

For every input block, output a line containing either "YES" or "NO". "YES" should be printed, if it is possible to assign costs to the phone lines operated by the company so that the cheapest calls are as advertised in the flyer. "NO" should be printed if this is not possible.

Example

Input :

```
2
3 3
1 2
2
3 2
1 2
```

Output:

YES

NO

Added by: Roman Sol
Date: 2005-04-17
Time limit [s]: 5
Source limit [B]: 10000
Resource: IPSC 2004

SPOJ Problem Set

332. Hard Question

Problem code: HARDQ

Students of computer science in Bratislava enjoy hiking and camping during their long summer breaks. They love walking silently in the groves, visiting sparkling waterfalls, exploring dark caves, climbing steep hills, or just sleeping in a tent. Some of them already visited all the national parks in Slovakia and nearby countries.

With no more new national parks to visit, frustrated students decided to set up a new national park (NP) by themselves. After long arguing, they finally agreed on the boundary of the NP. Now they want to purchase all the land needed for NP from present owners. Their funds are limited (after all, they are only students), therefore they do not want to buy any land outside the NP.

The NP can be described as a polygon with N vertices. There is a set P of M rectangular plots of land available for sale by their owners. The rectangles are mutually disjoint and axis-parallel. Your task is to decide whether it is possible to purchase subset of plots P exactly covering the proposed NP.

Input

Input file consists of several test cases separated by a blank line. Each test case starts with two integers N and M . Next N lines contain the coordinates of the vertices of the NP. Each of the following M lines describes one plot. For each plot, the coordinates of two opposite corners of the rectangle are given. The values $N=0$, $M=0$ end the input and should not be processed. [$N, M \leq 3000$]

Output

For each test case output either 'YES' or 'NO' depending on whether it is possible to set up the NP using P or not.

Example

Input :

```
4 3
0 0
0 2
2 2
2 0
1 0 0 2
1 0 2 2
```

```
3 1
0 0
2 2
2 0
0 0 1 1
```

```
0 0
```

Output :

```
YES
NO
```

Added by: Roman Sol
Date: 2005-04-17
Time limit [s]: 10
Source limit [B]: 30000
Resource: IPSC 2004

SPOJ Problem Set

334. The Philosophical Dispute

Problem code: PHDISP

One day, mathematician and philosopher were engaged in a heated dispute.

Philosopher said:

- Ideal line has only length and no width, therefore, no line can have an area.

Mathematician replied:

- That's as it may be, but still you can ll a square with a line in such a way that there will be no gaps.

And you can't deny that a square has an area, and he grinned.

But Philosopher still wasn't convinced:

- Show me this line, then.

- With pleasure... - responded Mathematician and scribbled some equations on a piece of paper:

$$\begin{cases} x = \sin(\sqrt{t}) \\ y = \cos(t) \end{cases}$$

- With t increasing, the point (x, y) will move around the square, forming a line.

- So what? - asked Philosopher. How is it going to ll the entire square?

- Indeed, it will, - said Mathematician, - Whichever point inside the square you draw, the line will eventually cross that point.

- No, - replied Philosopher indignantly, - Anyway, I don't believe. When will the line cross this point?

- and he put a thick dot inside the square.

Give Philosopher an answer.

Input

t – number of tests [$t \leq 150$], than t test cases follows.

The first line of each test case contains the coordinates (x_0 , y_0) of the dot center ($-1 \leq x_0$, $y_0 \leq 1$).

The second line contains $\text{eps} \leq 0.0001$ - the radius of the dot (the dot is essentially a small circle).

Output

For each test case output any value of t in the segment $[0, 10^{12}]$, which corresponds to the line crossing the dot, or "FAIL", if the line doesn't cross the dot.

Example

Sample input:

```
1
0.744 0.554
0.01
```

Sample output:

```
5.3
```

Added by: Roman Sol
Date: 2005-04-25
Time limit [s]: 5
Source limit [B]:20000
Resource: IX Ural Championship (Round II)

SPOJ Problem Set

336. Exchange Operations

Problem code: EOPERA

Given a sequence of 12 numbers consisting of 0 and the first 11 natural numbers. Suppose number 0 is in the i -th position of the sequence (positions are numbered from 0 to 11). You can swap it with the number in the j -th position if the following conditions hold:

- $|i - j| = d_k$, where $k=1..3$ and $(d_1, d_2, d_3, d_4) = (1; 3; 6; 12)$
- $\text{floor}(i/d_{k+1}) = \text{floor}(j/d_{k+1})$

Your task is to find the minimum number of exchange operations required to sort the sequence in increasing order.

Input

The first line of the input file contains an integer representing the number of test cases to follow. Each test case contains a sequence of twelve numbers consisting of 0,1,2,...,11, separated by single space. You can assume that the given sequence can always be sorted in increasing order by using the exchange operations

Output

For each test case, output the minimum number of exchange operations required to sort the given sequence in increasing order.

Example

Input :

2

1 10 2 3 0 5 7 4 8 6 9 11
6 4 1 0 3 5 9 7 2 10 11 8

Output :

8

9

Added by: Ngô Minh Đức
Date: 2005-04-28
Time limit [s]: 45
Source limit [B]:50000
Resource: Based on a problem from acm.uva.es

SPOJ Problem Set

338. Roads

Problem code: ROADS

N cities named with numbers 1 ... N are connected with one-way roads. Each road has two parameters associated with it: the road length and the toll that needs to be paid for the road (expressed in the number of coins). Bob and Alice used to live in the city 1. After noticing that Alice was cheating in the card game they liked to play, Bob broke up with her and decided to move away - to the city N. He wants to get there as quickly as possible, but he is short on cash. We want to help Bob to find the shortest path from the city 1 to the city N that he can afford with the amount of money he has.

Input

The input begins with the number t of test cases. Then t test cases follow. The first line of the each test case contains the integer K, $0 \leq K \leq 10000$, maximum number of coins that Bob can spend on his way. The second line contains the integer N, $2 \leq N \leq 100$, the total number of cities. The third line contains the integer R, $1 \leq R \leq 10000$, the total number of roads. Each of the following R lines describes one road by specifying integers S, D, L and T separated by single blank characters : S is the source city, $1 \leq S \leq N$ D is the destination city, $1 \leq D \leq N$ L is the road length, $1 \leq L \leq 100$. T is the toll (expressed in the number of coins), $0 \leq T \leq 100$ Notice that different roads may have the same source and destination cities.

Output

For each test case, output a single line contain the total length of the shortest path from the city 1 to the city N whose total toll is less than or equal K coins. If such path does not exist, output -1.

Example

Input :

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

Output:

11
-1

Added by: Ngô Minh Đức

Date: 2005-04-28

Time limit [s]: 15

Source limit [B]: 50000

Resource: Central European Olympiad in Informatics '98

SPOJ Problem Set

339. Recursive Sequence

Problem code: SEQ

Sequence (a_i) of natural numbers is defined as follows:

$$a_i = b_i \text{ (for } i \leq k)$$

$$a_i = c_1 a_{i-1} + c_2 a_{i-2} + \dots + c_k a_{i-k} \text{ (for } i > k)$$

where b_j and c_j are given natural numbers for $1 \leq j \leq k$. Your task is to compute a_n for given n and output it modulo 10^9 .

Input

On the first row there is the number C of test cases (equal to about 50).

Each test contains four lines:

k - number of elements of (c) and (b) ($1 \leq k \leq 10$)

b_1, \dots, b_k - k natural numbers where $0 \leq b_j \leq 10^9$ separated by spaces c_1, \dots, c_k - k natural numbers

where $0 \leq c_j \leq 10^9$ separated by spaces

n - natural number ($1 \leq n \leq 10^9$)

Output

Exactly C lines, one for each test case: a_n modulo 10^9

Example

Input :

```
3
3
5 8 2
32 54 6
2
3
1 2 3
4 5 6
6
3
24 354 6
56 57 465
98765432
```

Output :

```
8
714
257599514
```

Added by: Pawel Dobrzycki
Date: 2005-04-29
Time limit [s]: 4
Source limit [B]:8196
Resource: IV Podlasian Contest in Team Programming

SPOJ Problem Set

344. Poker

Problem code: POKER

In poker, you have 5 cards. There are 10 kinds of poker hands (from highest to lowest):

- royal flush - ace, king, queen, jack and ten, all in the same suit
- straight flush - five cards of the same suit in sequence, such as 10,9,8,7,6 of clubs; ace can be counted both as the highest card or as the lowest card - A,2,3,4,5 of hearts is a straight flush. But 4,3,2,A,K of hearts is not a straight flush - it's just a flush.
- four of a kind - four cards of the same rank, such as four kings.
- full house - three cards of one rank plus two cards of another rank
- flush - five cards of the same suit (but not a straight flush)
- straight - five cards in order - just like the straight flush, but mixed suits
- three of a kind - three cards of one rank and two other cards
- two pairs - two cards of one rank, two cards of another rank, and one more card
- pair - two cards of the same rank
- high card - none of the above

Write a program that will help you play poker by telling you what kind of hand you have.

Input

The first line of input contains the number of test cases (no more than 20). Each test case consists of one line - five space separated cards. Each card is represented by a two-letter (or digit) word. The first character is the rank (A,K,Q,J,T,9,8,7,6,5,4,3 or 2), the second character is the suit (S,H,D,C standing for spades, hearts, diamonds and clubs). The cards can be in any order (but they will not repeat).

Output

For each test case output one line describing the type of a hand, exactly like in the list above.

Example

Input:

```
3
AH KH QH TH JH
KH 5S 3C 5C 7D
QH QD 2S QC 2C
```

Output:

```
royal flush
pair
full house
```

Added by: Tomek Czajka
Date: 2005-05-03
Time limit [s]: 15
Source limit [B]:50000
Resource: Purdue Programming Contest Training

SPOJ Problem Set

345. Mixtures

Problem code: MIXTURES

Harry Potter has n mixtures in front of him, arranged in a row. Each mixture has one of 100 different colors (colors have numbers from 0 to 99).

He wants to mix all these mixtures together. At each step, he is going to take two mixtures that stand next to each other and mix them together, and put the resulting mixture in their place.

When mixing two mixtures of colors a and b , the resulting mixture will have the color $(a+b) \bmod 100$.

Also, there will be some smoke in the process. The amount of smoke generated when mixing two mixtures of colors a and b is $a*b$.

Find out what is the minimum amount of smoke that Harry can get when mixing all the mixtures together.

Input

There will be a number of test cases in the input.

The first line of each test case will contain n , the number of mixtures, $1 \leq n \leq 100$.

The second line will contain n integers between 0 and 99 - the initial colors of the mixtures.

Output

For each test case, output the minimum amount of smoke.

Example

Input :

```
2
18 19
3
40 60 20
```

Output :

```
342
2400
```

In the second test case, there are two possibilities:

- first mix 40 and 60 (smoke: 2400), getting 0, then mix 0 and 20 (smoke: 0); total amount of smoke is 2400
- first mix 60 and 20 (smoke: 1200), getting 80, then mix 40 and 80 (smoke: 3200); total amount of smoke is 4400

The first scenario is a much better way to proceed.

Added by: Tomek Czajka

Date: 2005-05-03

Time limit [s]: 20

Source limit [B]: 50000

Resource: Purdue Programming Contest Training

SPOJ Problem Set

346. Bytelandian gold coins

Problem code: COINS

In Byteland they have a very strange monetary system.

Each Bytelandian gold coin has an integer number written on it. A coin n can be exchanged in a bank into three coins: $n/2$, $n/3$ and $n/4$. But these numbers are all rounded down (the banks have to make a profit).

You can also sell Bytelandian coins for American dollars. The exchange rate is 1:1. But you can not buy Bytelandian coins.

You have one gold coin. What is the maximum amount of American dollars you can get for it?

Input

The input will contain several test cases (not more than 10). Each testcase is a single line with a number n , $0 \leq n \leq 1\,000\,000\,000$. It is the number written on your coin.

Output

For each test case output a single line, containing the maximum amount of American dollars you can make.

Example

Input:

12
2

Output:

13
2

You can change 12 into 6, 4 and 3, and then change these into $\$6 + \$4 + \$3 = \13 . If you try changing the coin 2 into 3 smaller coins, you will get 1, 0 and 0, and later you can get no more than \$1 out of them. It is better just to change the 2 coin directly into \$2.

Added by: Tomek Czajka
Date: 2005-05-03
Time limit [s]: 20
Source limit [B]: 50000
Resource: Purdue Programming Contest Training

347. Lazy Cows

Farmer John regrets having applied high-grade fertilizer to his pastures since the grass now grows so quickly that his cows no longer need to move around when they graze. As a result, the cows have grown quite large and lazy... and winter is approaching.

Farmer John wants to build a set of barns to provide shelter for his immobile cows and believes that he needs to build his barns around the cows based on their current locations since they won't walk to a barn, no matter how close or comfortable.

The cows' grazing pasture is represented by a 2 x B (1 ≤ B ≤ 15,000,000) array of cells, some of which contain a cow and some of which are empty. N (1 ≤ N ≤ 1000) cows occupy the cells in this pasture:

	COW				COW	COW	COW	COW	
	COW	COW	COW						

Ever the frugal agrarian, Farmer John would like to build a set of just K ($1 \leq K \leq N$) rectangular barns (oriented with walls parallel to the pasture's edges) whose total area covers the minimum possible number of cells. Each barn covers a rectangular group of cells in their entirety, and no two barns may overlap. Of course, the barns must cover all of the cells containing cows.

By way of example, in the picture above if $K=2$ then the optimal solution contains a 2×3 barn and a 1×4 barn and covers a total of 10 units of area.

The first line of the input contains integer t representing the number of test cases. Then t cases follow. Each case has the following form:

- Line 1: Three space-separated integers, N, K, and B.
- Lines 2..N+1: Two space-separated integers in the range (1,1) to (2,B) giving the coordinates of the cell containing each cow. No cell contains more than one cow.

For each test case, output the minimum area required by the K barns in order to cover all of the cows.

```
Input :
1
8 2 9
1 2
```

1 6
1 7
1 8
1 9
2 2
2 3
2 4

Output:

10

Input details:

As pictured above.

Output details:

As discussed above.

Added by: Ngô Minh Đức

Date: 2005-05-03

Time limit [s]: 20

Source limit [B]:50000

Resource: US Open International 2005 Gold Division

SPOJ Problem Set

348. Expedition

Problem code: EXPEDI

A group of cows grabbed a truck and ventured on an expedition deep into the jungle. Being rather poor drivers, the cows unfortunately managed to run over a rock and puncture the truck's fuel tank. The truck now leaks one unit of fuel every unit of distance it travels.

To repair the truck, the cows need to drive to the nearest town (no more than 1,000,000 units distant) down a long, winding road. On this road, between the town and the current location of the truck, there are N ($1 \leq N \leq 10,000$) fuel stops where the cows can stop to acquire additional fuel (1..100 units at each stop).

The jungle is a dangerous place for humans and is especially dangerous for cows. Therefore, the cows want to make the minimum possible number of stops for fuel on the way to the town. Fortunately, the capacity of the fuel tank on their truck is so large that there is effectively no limit to the amount of fuel it can hold. The truck is currently L units away from the town and has P units of fuel ($1 \leq P \leq 1,000,000$).

Determine the minimum number of stops needed to reach the town, or if the cows cannot reach the town at all.

Input

The first line of the input contains an integer t representing the number of test cases. Then t test cases follow. Each test case has the following form:

- Line 1: A single integer, N
- Lines 2.. $N+1$: Each line contains two space-separated integers describing a fuel stop: The first integer is the distance from the town to the stop; the second is the amount of fuel available at that stop.
- Line $N+2$: Two space-separated integers, L and P

Output

For each test case, output a single integer giving the minimum number of fuel stops necessary to reach the town. If it is not possible to reach the town, output -1.

Example

Input :

```
1
4
4 4
5 2
11 5
15 10
25 10
```

Output:

2

Input details

The truck is 25 units away from the town; the truck has 10 units of fuel. Along the road, there are 4 fuel stops at distances 4, 5, 11, and 15 from the town (so these are initially at distances 21, 20, 14, and 10 from the truck). These fuel stops can supply up to 4, 2, 5, and 10 units of fuel, respectively.

Output details:

Drive 10 units, stop to acquire 10 more units of fuel, drive 4 more units, stop to acquire 5 more units of fuel, then drive to the town.

Added by: Ngô Minh Đức

Date: 2005-05-03

Time limit [s]: 20

Source limit [B]: 50000

Resource: US Open International 2005 Gold Division

SPOJ Problem Set

349. Around the world

Problem code: AROUND

Over the years, FJ has made a huge number of farmer friends all around the world. Since he hasn't visited 'Farmer Ted' from England and 'Boer Harms' from Holland for a while, he'd like to visit them.

He knows the longitude of the farm where each of his worldwide friends resides. This longitude is an angle (an integer in the range 0..359) describing the farm's location on the Earth, which we will consider to be a circle instead of the more complex and traditional spherical representation. Except for the obvious discontinuity, longitudes increase when traveling clockwise on this circle.

FJ plans to travel by airplane to visit his N ($1 \leq N \leq 5,000$) friends (whose farms are uniquely numbered 1.. N). He knows the schedules for M ($1 \leq M \leq 25,000$) bidirectional flights connecting the different farms. Airplanes always travel shortest paths on the Earth's surface (i.e., on the shortest arc of a circle).

There will always be a unique shortest path between two farms that are directly connected. No pair of antipodal farms (exactly opposite each other on the circle) is ever directly connected.

Each airplane flight can be described as traveling in clockwise or counterclockwise direction around the Earth's surface. For example, a flight from longitude 30 to longitude 35 would be clockwise, as would be a flight from longitude 350 to longitude 10. However, a flight from longitude 350 to longitude 200 follows a shortest path counterclockwise around the circle.

FJ would find it very cool if he could make a trip around the world, visiting some of his friends along the way. He'd like to know if this is possible and if so, what is the minimum number of flights he can take to do so.

He wants to start and finish his journey at the location of his best friend (the one listed first in the input below). In order to make sure he actually circles the Earth, he wants to ensure that the clockwise distance he travels is different from the counterclockwise distance he travels.

Input

The first line of the input contains an integer t representing the number of test cases. Then t test cases follow. Each test case has the following form:

- Line 1: Two space-separated integers: N and M
- Lines 2.. $N+1$: Line $i+1$ contains one integer: the longitude of the i -th farm. Line 2 contains the location of the farm of his best friend.
- Lines $N+2$.. $N+M+1$: Line $i+N+1$ contains two integers giving the indices of two farms that are connected by a flight.

Output

For each test case, output a single integer specifying the minimum number of flights FJ needs to visit to make a trip around the world. Every time FJ moves from one farm to another counts as one flight. If it is impossible to make such a trip, output the integer -1.

Example

Input:

```
1
3 3
0
120
240
1 2
2 3
1 3
```

Output:

```
3
```

Input details

Farmer John has three friends at longitudes 0, 120, and 240. There are three flights: 0 \leftrightarrow 120, 120 \leftrightarrow 240, and 0 \leftrightarrow 240. The journey must start and finish at longitude 0.

Output details

FJ must visit all 3 friends to make a full trip around the world.

Added by: Ngô Minh Đức
Date: 2005-05-03
Time limit [s]: 20
Source limit [B]: 50000
Resource: US Open International 2005 Gold Division

SPOJ Problem Set

350. Landscaping

Problem code: LANDSCAP

Farmer John is making the difficult transition from raising mountain goats to raising cows. His farm, while ideal for mountain goats, is far too mountainous for cattle and thus needs to be flattened out a bit. Since flattening is an expensive operation, he wants to remove the smallest amount of earth possible.

The farm is long and narrow and is described in a sort of two-dimensional profile by a single array of N ($1 \leq N \leq 1000$) integer elevations (range 1..1,000,000) like this:

1 2 3 3 3 2 1 3 2 2 1 2,

which represents the farm's elevations in profile, depicted below with asterisks indicating the heights:

```
      * * *      *
    * * * * *   * * *   *
  * * * * * * * * * * *
1 2 3 3 3 2 1 3 2 2 1 2
```

A contiguous range of one or more equal elevations in this array is a "peak" if both the left and right hand sides of the range are either the boundary of the array or an element that is lower in elevation than the peak. The example above has three peaks.

Determine the minimum volume of earth (each unit elevation reduction counts as one unit of volume) that must be removed so that the resulting landscape has no more than K ($1 \leq K \leq 25$) peaks. Note well that elevations can be reduced but can never be increased.

If the example above is to be reduced to 1 peak, the optimal solution is to remove $2 + 1 + 1 + 1 = 5$ units of earth to obtain this set of elevations:

```
      * * *      -
    * * * * *   - - - -
  * * * * * * * * * *
1 2 3 3 3 2 1 1 1 1 1
```

where '-'s indicate removed earth.

Input

The first line of the input contains integer t representing the number of test cases. Then t test cases follow. Each test case has the following form:

- Line 1: Two space-separated integers: N and K
- Lines 2.. $N+1$: Each line contains a single integer elevation. Line $i+1$ contains the elevation for index i .

Output

For each test case, output the minimum volume of earth that must be removed to reduce the number of peaks to K.

Example

Input :

```
1
12 1
1
2
3
3
3
2
1
3
2
2
1
2
```

Output:

```
5
```

Input details

This is the example used above.

Added by: Ngo Minh Duc

Date: 2005-05-03

Time limit [s]: 20

Source limit [B]:50000

Resource: US Open International 2005 Gold Division

SPOJ Problem Set

351. Ha-noi!

Problem code: HAN01

Little Sabrina loves solving puzzles. Last week she got a new puzzle: The "Tower of Hanoi" puzzle. This puzzle is based on an old legend: "The temple priests of hanoi have to transfer a tower consisting of 64 fragile disks of gold from one part of the temple to another, one disk at a time. The disks are arranged in order, no two of them the same size, with the largest on the bottom and the smallest on top. Because of their fragility, a larger disk may never be placed on a smaller one, and there is only one intermediate location where disks can be temporarily placed. It is said that before the priests complete their task the temple will crumble into dust and the world will vanish in a clap of thunder." Sabrina reconstructed the problem with some coins of different size. She solved the puzzle for three coins in 7 steps, for four coins in 15 steps,... after solving the problem with 7 coins she had the hang of it. Yesterday she started to solve the puzzle with 31 coins and her optimal strategy. After hours of moving coins from one pile to the other she was very tired and went to bed. This was a bad idea! Her little brother Robin discovered the towers of coins and - whoops! - threw it on the floor. Then he noticed a sheet of paper: "Don't touch this towers! Steps: 16543". "Oh no!" Robin has to reconstruct the tower because his sister can get very, very angry... Your task is to help Robin to reconstruct the towers. Sabrina started the game with all disks on peg number one and her goal was to move the disks to peg number two. She used her optimal strategy and noted the number of steps she had done.

Input

The first line of input contains one integer t: The number of testcases. k lines follow. Each line contains two integers n ($2 < n < 61$) and k ($0 < k < 2^n$). n is the number of disks of the hanoi puzzle and k the number of steps Sabrina had done.

Output

Output the reconstructed configuration of the towers after k steps. For each testcase output three lines. One for each tower. Each line consists of the tower identifier (1,2,3) a colon, one space and the disk numbers (n,n-1,...,2,1) which are separated by a '|' character.

Example

Input :

```
3
3 6
32 889397450
31 16543
```

Output :

```
1: 1
2: 3|2
3:
1: 32|31|28|25|18|17|14|3
2: 30|29|26|13|12|11|10|9|6|5|2
```

3: 27|24|23|22|21|20|19|16|15|8|7|4|1
1: 31|30|29|28|27|26|25|24|23|22|21|20|19|18|17|16|7|6
2: 15|8|5|4|3|2|1
3: 14|13|12|11|10|9

Added by: Simon Gog
Date: 2005-05-03
Time limit [s]: 3
Source limit [B]:8082
Resource: Ulm Algorithm Course SoSe 2005

SPOJ Problem Set

353. Displace

Problem code: DISPLACE

You are given two strings S_1 , S_2 of not more than 250 characters each. S_1 does not contain characters '(' and ')'. You can swap two consecutive characters in S_1 . Your task is to do it in as small a number of swapping operations as possible to obtain a string which contains S_2 as a substring (you can assume that for the given input, this can always be done).

Input

The first line of the input file contains an integer t representing the number of test cases ($t < 20$). Then t test cases follow. Each test case has the following form:

- The first line contains S_1
- The second line contains S_2

Output

For each test case, output 0 iff you do not want to solve this test case. Otherwise, output a line containing the number 1 and two more lines of the following form:

- The first line contains an integer k representing the number of swap operations
- The second line contains k integers $p_1 p_2, \dots, p_k$ separated by single spaces, p_i means that in the i -th operation, you swapped the i -th character and the $(i+1)$ -th character in S_1 .

Score

Your task is to minimise your score for this problem. If you choose to solve a test case and the number of swap of operations is smaller than 5000, your score is equal to the number of operations. Otherwise, your score is 5000. Your total score is equal to the sum of scores for individual tests.

Example

Input :

```
1
ABCDEFGH
FC
```

Output :

```
1
3
5 4 3
```

Score :

```
3
```

Added by: Ngô Minh Đức
Date: 2005-05-05
Time limit [s]: 20
Source limit [B]: 50000

SPOJ Problem Set

359. Alpha Centauri Tennis

Problem code: ACT

As you may know, planets of Alpha Centauri (if they indeed do exist) would provide excellent conditions for intelligent life forms.

It is indeed true that there is a small Earthlike planet near Alpha Centauri, inhabited by a population of no particular significance. These humanlike creatures have much in common with us. Living in similar communities and having similar body structure and behavioral patterns, they unsurprisingly appreciate (approximately) the same time-killing activities as we do. One of these, the second most popular after Alpha Centauri Croquet, is the Alpha Centauri Tennis.

Although its rules differ from Earth Tennis, the two player version of Alpha Centauri Tennis resembles it in many ways. Same as Earth Tennis, it is played on a rectangular court divided into two parts by a net. Two players, standing on opposite sides of it, use a stringed racket to hit a ball back and forth to each other. There are certain rules how to hit the ball. The player who forces his opponent to violate one of these rules wins the current ball. The aim of both players is to win enough balls to win a game, enough games to win a set and enough sets to win the whole match. In the N player version of the Alpha Centauri Tennis a ball can be won by any one of the N players. Although technical details of this can be difficult to imagine, Alpha Centaurians are extremely inventive.

In the general N-player version, players serve in turns, following order determined before the match. Moreover, they shift when starting individual games and sets. For example, the players are A, B and C. They are ordered alphabetically. Player A serves the first ball of the first game. When the ball is won by one of the players, its B's turn to serve. After the game is won by one of the players, player B starts the second game. Finally, when the first set is won by someone, player B starts the second set. This repeats, always shifted by one player, until the match ends.

For three players the serving order looks as follows:

Set 1:

Game 1: A,B,C,A,B,C...

Game 2: B,C,A,....

Game 3: C,A,B,....

Game 4: A,B,C,....

...

Set 2:

Game 1: B,C,A,B,....

Game 2: C,A,B,....

Game 3: A,B,C,A,...

...

There are exact rules for counting the number of balls/games/sets won by a player.

RULES FOR WINNING A GAME

The state of a game can be described by assigning a non-negative number of points to each of the players. At the beginning of a game, the score of each player is zero.

Note: In Earth terminology, 0 points is called "love", 1 point is a "fifteen", 2 points is a "thirty", 3 points is a "forty" and 4 points is an "advantage". Be glad that you don't have to learn the Centaurian terminology :)

When a player P just won a ball, the new score is determined by using the first rule from the list that applies to the situation.

If P currently has 3 points and no other player has more than 2 points, P wins the current game.

If P currently has 4 points, he wins the game.

If any other player currently has 4 points, that player loses one point. P gains a point.

RULES FOR WINNING A SET

The set is won by the first player that at the same time:

won at least 6 games in this set

won at least 2 games more than any other player

RULES FOR WINNING A MATCH

The winner is the first player to win at least three sets. A set in which no other player won a game counts as two won sets.

Problem specification

An observer from the Intergalactic Tennis Federation was watching a tournament in Alpha Centauri Tennis. Being unable to understand Alpha Centaurian language, he only managed to write down the winner of each ball. Now, for each match, knowing the sequence in which the players were winning the balls, he would like to somehow determine its winner.

Input

t - the number of test cases [$t \leq 150$] then t test cases follows, each corresponding to one match. Each line contains the number of players N [$N \leq 10$] and a string S consisting of uppercase letters [$2 \leq S \leq 50000$]. The players are represented by the first N letters of the English alphabet. If the i-th letter of S is X, it means the player X won the i-th ball from the beginning of the match. You may assume that the match transcripts are correct and complete.

The order in which the players serve is the same as the order of their letters in the English alphabet.

Output

For each line, output a single character, being the letter of the player who won the corresponding match.

Example

Input :

```
1
3 BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
```

Output :

```
B
```

(B has won two sets, each of them by winning 6 games, while A and C won none. Thus each of these sets counts as two and B has won the match.)

Added by: Roman Sol
Date: 2005-05-13
Time limit [s]: 1
Source limit [B]: 10000
Resource: IPSC 2005

SPOJ Problem Set

360. Bottom Coder (Easy)

Problem code: BCEASY

Some of you may be familiar with the TopCoder (TM) contest. Its exact rules are not important for this problem, but know that the most important part of it is writing a program according to the given specification. Many times the contestant ends up with a program which would work perfectly – if only he could change a couple of characters (like, replacing "=" by "==" in C, etc.). Unfortunately, even the best programmers sometimes aren't able to spot these tiny but necessary changes until it's too late... and that's why we developed a brand-new BottomCoder training for them!

The idea is very simple – you're given a problem specification, a source code, and a list of permitted modifications. Your task is to find a modification which would cause the program to behave according to the specification.

Specification: "Write a program which outputs EXACTLY 42 asterisks and NOTHING more (e.g. NO end-of-line markers, like "\n", ...)"

The code you are supposed to modify:

```
int i, n=42;
main() {
    for(i=0; i<n; i--) {
        printf("*");
    }
}
```

As this is a really, really simple problem, you are only permitted to make exactly ONE of these modifications to the source: 1) Add one character to the source. 2) Delete one character from the source. 3) Replace one character in the source by a different one.

Moreover, it would be definitely too easy if we asked you to find just one solution, so you'll need to find TWO DIFFERENT solutions in order to obtain credit for this problem. (There are exactly three different solutions, so don't worry, it can be done!)

Input

There is no input for given problem.

Output

Your submission should consist of two parts. The first part should contain the first of your solutions. A single line with the letter "Q" follows. (Note that the letter Q is used as a separator. You will have to do without inserting the letter Q in at least one of your solutions :) After this line you should add your second solution.

You don't need to worry much about the exact formatting of your submission. The exact judging procedure will look as follows:

The first occurrence of the letter Q is found, the input is split into two parts. Any whitespace in each of the parts is removed. It is checked whether the two submissions differ and whether each of them was obtained from the original program by an allowed change. Each of your two submissions is compared to each of the three correct solutions.

Example

Output:

```
int i, n=42; main(
){ for(i=0; i<n; i--)    { printf("?"); } }
Q
int i, n=41; main() { for(i=0; i<n;i--) { printf("*"); } }
```

(syntactically valid (but incorrect) submission)

Added by: Roman Sol
Date: 2005-05-13
Time limit [s]: 1
Source limit [B]:10000
Resource: IPSC 2005

SPOJ Problem Set

361. Bottom Coder (Hard)

Problem code: BCHARD

Some of you may be familiar with the TopCoder (TM) contest. Its exact rules are not important for this problem, but know that the most important part of it is writing a program according to the given specification. Many times the contestant ends up with a program which would work perfectly – if only he could change a couple of characters (like, replacing "=" by "==" in C, etc.). Unfortunately, even the best programmers sometimes aren't able to spot these tiny but necessary changes until it's too late... and that's why we developed a brand-new BottomCoder training for them!

The idea is very simple – you're given a problem specification, a source code, and a list of permitted modifications. Your task is to find a modification which would cause the program to behave according to the specification.

Specification: "Write a program which outputs a short English text mentioning our partner competition – IPSC. The text must consist of one or more English sentences and each sentence has to contain one or more English words (sequences of only upper-case characters) separated by spaces. Additionally, you may use certain punctuation characters – namely "!.?,'". Try to obfuscate the program as much as possible." The code you are supposed to modify:

```
#include <stdio.h>

int rex[5];

void f3(int *a) {
    int i;
    for (i=0; i<5; i++) a[i]=0;
}

int f2(int *a) {
    int i;
    for (i=0; i<5; i++) if (a[i]!=0) return 0;
    return 1;
}

void f1(int *a) {
    int i;
    for (i=0; i<5; i++) {
        a[i]++;
        if (a[i]<100) break;
        a[i]-=100;
    }
    for (i=4; i>=0 && a[i]>=rex[i]; i--)
        if (a[i]>rex[i])
            f3(a);
}

void f4(int *a) {
    int i;
    for (i=0; i<5; i++) {
        a[i]--;
        if (a[i]>=0) break;
        a[i]+=100;
    }
}
```

```

    if (i>=5) for (i=0; i<5; i++) a[i]=rex[i];
}

void f7(int *a, int *b) {
    int c[5];
    f3(c); f3(a);
    while(!f2(b)) { f1(a); f4(b); f1(c); }
    while(!f2(c)) { f1(b); f4(c); }
}

void f9(int *a, int *b) {
    f1(a);
    while(!f2(b)) { f4(b); f1(a); }
}

void f8(int *a, int *b) {
    int c[5], d[5];
    f7(d, a);
    f3(a); f1(a);
    while(!f2(b)) { f7(c, d); f9(a, c); f4(a); f4(b); }
}

void f5(int *a, int *b) {
    int c[5], d[5];
    f7(d, a);
    f3(a); f1(a);
    while(!f2(b)) { f7(c, d); f8(a, c); f4(a); f4(b); }
}

void f10(int x) {
    int rpl[] =
{80, 125, 111, 18, 59, 88, 88, 28, 65, 98, 119, 103, 101, 79, 107, 2, 16,
92, 102, 123, 103, 84, 112, 78, 68, 98, 65, 37, 105, 85, 107, 13, 45, 9,
104, 81, 21, 31, 55, 110, 78, 66, 66, 3, 77, 63, 16, 105, 15, 123, 16, 84,
31, 96, 4, 82, 82, 122, 68, 115, 35, 73, 3, 108, 115, 83, 15, 19, 31, 99, 5,
123, 24, 65, 36, 15, 75, 84, 4, 2, -1};

    int i;
    int a[5], b[5], c[5];

    if (x<100000000 || x>200000000) return;
    x--;
    f3(rex); rex[4]=1;
    for (i=0; rpl[i]!=-1; i++)
    {
        f3(a); a[0]=i+1;
        f3(b); f1(b); f3(c); f1(b); f1(b);
        f1(c); f1(b); f5(a, b);

        f1(c);
        while(!f2(a))
        {
            f3(b); b[0]=x%100; b[1]=x/100;
            f4(a); f8(c, b);
        }
        rpl[i]^=c[1];
        printf("%c", rpl[i]);
    }
    printf("\n");
}

```

```
int main()
{
    f10(47);
}
```

As you can see, the coder made almost everything according to the specification :) You're only allowed to alter one number in the source code – namely the number 47 on line 98 (the argument of function "f10" called from "main"). You can replace it by any integer between 100'000'000 and 200'000'000 inclusive.

Input

There is no input for given problem.

Output

Your submission should consist of two lines. The first line should contain the value of the constant – an integer between 100'000'000 and 200'000'000 inclusive. The second line should contain the output produced by the program if it were compiled and executed with the correct value of the constant.

Example

Output:

```
123456789
ARE YOU SOLVING IPSC PROBLEMS RIGHT NOW?
```

(syntactically valid (but incorrect) submission)

Added by: Roman Sol
Date: 2005-05-13
Time limit [s]: 1
Source limit [B]: 1000
Resource: IPSC 2005

SPOJ Problem Set

362. Ignore the Garbage

Problem code: IGARB

Fred works as an IT consultant in an insurance company. As they always had a large amount of customers waiting and arguing at the front desk, management decided to deploy a ticket machine. Each customer would get a ticket with a number and there will be fancy LCD display over each desk showing the number of the next person. Fred was appointed to get this new enhancement working.

Because Fred is lazy when it comes to manual labor and as an IT consultant he wouldn't lower himself to the level of some hardware technician (except when upgrading his own computer), he asked few technicians to install the displays and prepared himself just to plug in the ticket machine and try it out. Unfortunately (for Fred) the technicians, either inspired by Mr.Bean or because of their carelessness, installed the display upside-down.

Being a software guy, Fred decided that the hardware should not be tampered with after it is installed (except for the case if he would be able to get back the technicians to repair it, but they were already angry at him for his nagging). Then he noted that from time to time the display shows a correct number even when it is upside-down. And hey, the ticket machine is an embedded device and contains a small processor! It would be just a sin for an IT guy not to try to meddle with it and try running an own version of Linux. Now we just need to figure out which readable numbers will the display show.

Task specification

In the beginning the display shows the number 1 on its display. Each second the number shown is increased by 1. We see the display upside-down and thus not everything we see will make sense. Your task is to compute the K-th valid number we will see on the display. The digits the display uses are shown on the images below. An upside-down 1 still count as 1. The number we see may have leading zeroes – e.g. turning the number 600 upside down leads to a valid number.



Input

t - the number of test cases [$t \leq 2200$], then t test cases follow. Each test case consists of one integer K_i [$0 < K_i \leq 10^{200}$].

Output

For each K_i from the input file, output the K_i -th number shown on the display (including the leading zeroes, if there are some).

Example

Input :

8
1
2
3
4
5
6
8
98

Output :

1
2
5
9
8
6
11
002

Added by: Roman Sol
Date: 2005-05-15
Time limit [s]: 5
Source limit [B]:30000
Resource: IPSC 2005

SPOJ Problem Set

363. Crane Operator

Problem code: COPER

You are the operator of a crane. In the reach of the crane there are N locations numbered from 0 to $N-1$. Initially the location 0 is empty and each of the other locations contains a box with the respective number (i.e. the box in the location i has the number i).

You were assigned a task to rearrange the boxes to form a special top-secret configuration. Because of the secrecy the configuration had to be transferred in a special way that only you can understand.

The secret configuration is characterized by five numbers q, p, m, d and M . To find out the final box locations you should follow this procedure: Consider the boxes in the order of their numbers. (I.e. the box number 1 comes first, the block number 2 second, etc.) The final location for the i -th box will have the number $pi = (ci + d * x + y) \bmod N$, where:

- 1) Numbers ci are defined by the recurrence equation: $c0 = 0, ci + 1 = (ci * q + p) \bmod m$
- 2) Numbers x and y are nonnegative integers chosen in such a way that $pi < M$ and no previous box was assigned to the location pi
- 3) If there are more choices for x and y satisfying conditions 1 and 2, choose the one that minimizes y
- 4) If there are more choices for x and y satisfying conditions 1, 2 and 3, choose the one that minimizes x

Once you find the secret configuration, you are supposed to rearrange the boxes accordingly. Unfortunately there is almost no room to manipulate the boxes. The only operation you are able to do is to lift a box with the crane and drop it on the location that is currently empty. (Note that there will always be exactly one empty location.)

There is one more problem. The whole operation is extremely time-critical and it is imperative that you should achieve the final configuration as soon as possible – i.e. you must reach it using the minimal number of box moves.

Task specification

Given the numbers N, M, q, p, m and d , determine how many described operations (lifting a box and dropping it to the free location) are sufficient and necessary to reach the configuration determined using the procedure above.

Input

On the first line of input file there is one number T – the number of test cases. [$T \leq 7000$] T lines follow, each of them describing one test case. On each of these lines there are six numbers N, M, q, p, m and d separated by spaces. [$N, M, d \leq 1000000$] [$q, p, m \leq 10000$]

Output

For each test case output one line containing the number of operations sufficient and necessary to reach the final configuration.

Example

Input:

1
8 3 5 2 7 4

Output:

6

The recurrence gives the values **c1=2, c2=5, c3=6, c4=4, c5=1, c6=0** and **c7=2**. The secret configuration is: Box 6 on the location 0, box 5 on the location 1, box 1 on the location 2, location 3 empty, box 4 on the location 4, box 2 on the location 5, box 3 on the location 6 and box 7 on the location 7. We need 6 moves to reach this configuration.

Added by: Roman Sol
Date: 2005-05-16
Time limit [s]: 30
Source limit [B]: 30000
Resource: IPSC 2005

SPOJ Problem Set

364. Pocket Money

Problem code: LISA

Young people spend a lot of money on things like sweets, music CDs, mobile phones and so on. But most young girls/boys have one problem: Their pocket money is not enough for all these jolly things. Little Lisa Listig is one of these poor girls with a little pocket money budget. Last month her pocket money lasted only for one week. So she decided to enter into negotiations with her father. Her father Tomm - a mathematician - had a incredible ingenious idea: He wrote down some fancy digits with operators (+,*) inbetween them on a sheet of paper and allowed Lisa to insert brackets. Then he defined that the result of that arithmetic expression is the new pocket money of Lisa. Now it's Lisa task to maximize her pocket money. As her father was surprised what huge amount of money Lisa got for her result he decided to minimize the result of the expression for his son Manfred. Now it's your task to calculate the result of Lisa and her father.

Input

The first line of input consists the number of testcases k ($k < 5000$). Each of the following k lines consists of a arithmetic expression. This expression consists of two operators '*' and '+' and digits ('0'-'9'). There are no spaces between the characters.

Output

For each expressin output the result of Lisa and the result of her father separated by one space. The result of the calculations are smaller than 2^{64} .

Example

Input:

```
1
1+2*3+4*5
```

Output:

```
105 27
```

Two possible expressions for the first testcase:

```
105 = (1+2)*(3+4)*5
27  = 1+2*3+4*5
```

Added by: Simon Gog
Date: 2005-05-17
Time limit [s]: 32
Source limit [B]: 8082
Resource: Ulm Algorithm Course SoSe 2005

SPOJ Problem Set

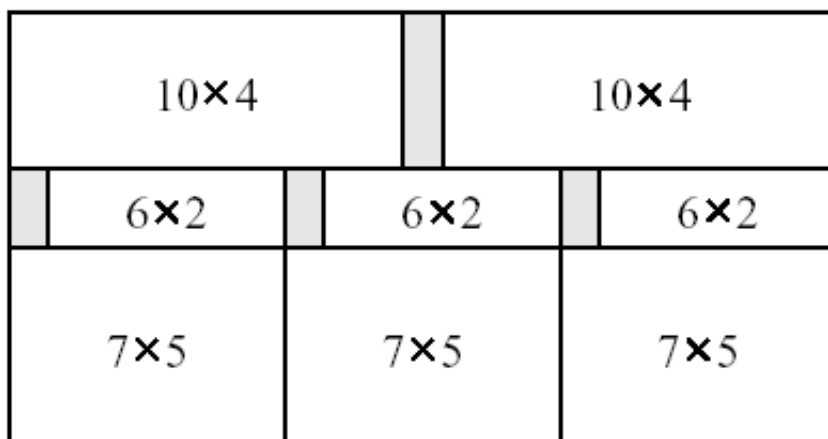
365. Phidias

Problem code: PHIDIAS

Famous ancient Greek sculptor Phidias is making preparations to build another marvelous monument. For this purpose he needs rectangular marble plates of sizes $W_1 \times H_1$, $W_2 \times H_2$, ..., $W_N \times H_N$.

Recently, Phidias has received a large rectangular marble slab. He wants to cut the slab to obtain plates of the desired sizes. Any piece of marble (the slab or the plates cut from it) can be cut either horizontally or vertically into two rectangular plates with integral widths and heights, cutting completely through that piece. This is the only way to cut pieces and pieces cannot be joined together. Since the marble has a pattern on it, the plates cannot be rotated: if Phidias cuts a plate of size $A \times B$ then it cannot be used as a plate of size $B \times A$ unless $A = B$. He can make zero or more plates of each desired size. A marble plate is wasted if it is not of any of the desired sizes after all cuts are completed. Phidias wonders how to cut the initial slab so that as little of it as possible will be wasted.

As an example, assume that in the figure below the width of the original slab is 21 and the height of the original slab is 11, and the desired plate sizes are 10×4 , 6×2 , 7×5 , and 15×10 . The minimum possible area wasted is 10, and the figure shows one sequence of cuts with total waste area of size 10.



Your task is to write a program that, given the size of the original slab and the desired plate sizes, calculates the minimum total area of the original slab that must be wasted.

Input

t - the number of test cases, then t numbers follows [$t \leq 20$] The first line of each test case contains two integers: first W , the width of the original slab, and then H , the height of the original slab. The second line contains one integer N : the number of desired plate sizes. The following N lines contain the desired plate sizes. Each of these lines contains two integers: first the width W_i and then the height H_i of that desired plate size ($1 \leq i \leq N$). [$1 \leq W \leq 600$, $1 \leq H \leq 600$, $0 < N \leq 200$, $1 \leq W_i \leq W$, and $1 \leq H_i \leq H$.]

Output

For each test case output one line with a single integer: the minimum total area of the original slab that must be wasted.

Example

Input:

```
21 11
4
10 4
6 2
7 5
15 10
```

Output:

```
10
```

Added by: Roman Sol
Date: 2005-05-20
Time limit [s]: 50
Source limit [B]: 30000
Resource: IOI 2004

SPOJ Problem Set

366. Farmer

Problem code: FARMER

A farmer has a set of fields, each of which is surrounded by cypress trees. Also, the farmer has a set of strips of land, each of which has a row of cypress trees. In both fields and strips, between every two consecutive cypress trees is a single olive tree. All of the farmer's cypress trees either surround a field or are in a strip and all of the farmer's olive trees are between two consecutive cypress trees in a field or in a strip.

One day the farmer became very ill and he felt that he was going to die. A few days before he passed away he called his eldest son and told him, "I give you any Q cypress trees of your choice and all the olive trees which are between any two consecutive cypress trees you have chosen." >From each field and from each strip the son can pick any combination of cypress trees. Since the eldest son loves olives he wants to pick the Q cypress trees which will allow him to inherit as many olive trees as possible.

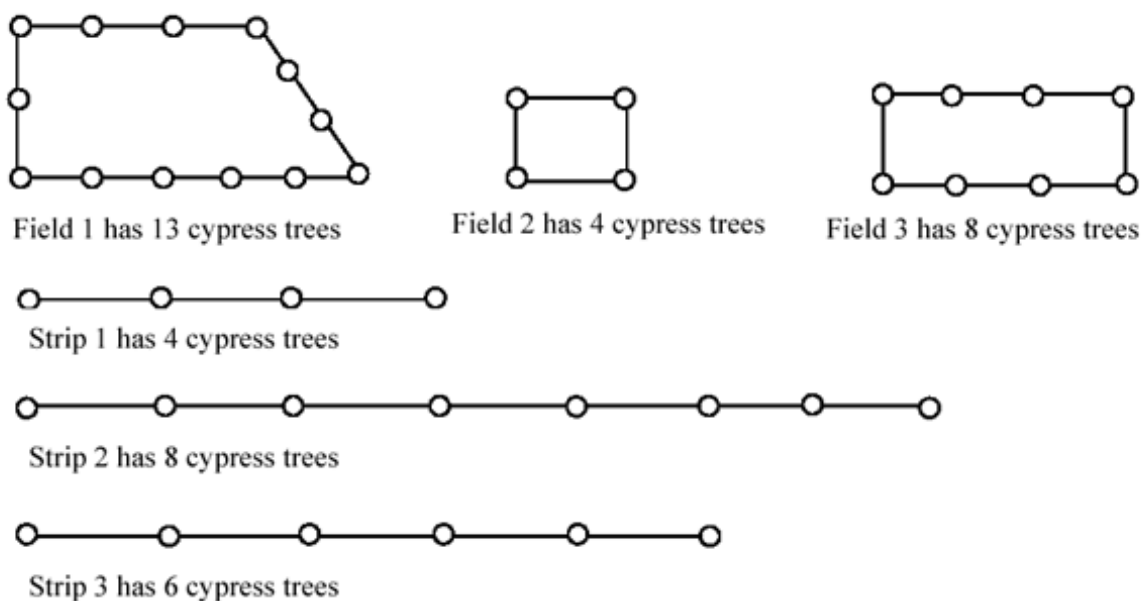


Figure 1. An example setting of cypress trees; olive trees are not shown.

In Figure 1, assume that the son is given $Q=17$ cypress trees. To maximize his olive inheritance he should choose all the cypress trees in Field 1 and Field 2, inheriting 17 olive trees.

You are to write a program which, given the information about the fields and the strips and the number of cypress trees the son can pick, determines the largest possible number of olive trees the son may inherit.

Input

t - the number of test cases [$t \leq 20$], then t test cases follow. The first line of each test case contains first the integer Q: the number of cypress trees the son is to select; then the integer M, the number of fields; and then the integer K, the number of strips. The second line contains M integers N_1, N_2, \dots, N_M , the numbers of cypress trees in fields. The third line contains K integers R_1, R_2, \dots, R_K : the numbers of cypress trees in strips.

In all test cases, $0 \leq Q \leq 150000$, $0 \leq M \leq 2000$, $0 \leq K \leq 2000$, $3 \leq N_1 \leq 150$, $3 \leq N_2 \leq 150, \dots, 3 \leq N_M \leq 150$, $2 \leq R_1 \leq 150$, $2 \leq R_2 \leq 150, \dots, 2 \leq R_K \leq 150$. The total number of cypress trees in the fields and strips is at least Q. Additionally, in 50% of the test cases, $Q \leq 1500$.

Output

For each test case output one integer: largest possible number of olive trees the son may inherit.

Example

Input :

```
1
17 3 3
13 4 8
4 8 6
```

Output :

```
17
```

Added by: Roman Sol
Date: 2005-05-22
Time limit [s]: 50
Source limit [B]: 30000
Resource: IOI 2004

SPOJ Problem Set

367. Empodia

Problem code: EMPODIA

The ancient mathematician and philosopher Pythagoras believed that reality is mathematical in nature. Present-day biologists study properties of biosequences. A biosequence is a sequence of M integers, which

- contains each of the numbers $0, 1, \dots, M-1$,
- starts with 0 and ends with $M-1$, and
- has no two elements $E, E+1$ in adjacent positions in this order.

A subsequence consisting of adjacent elements of a biosequence is called a segment..

A segment of a biosequence is called a framed interval if it includes all integers whose values are between the value of the first element, which must be the smallest element in the segment, and the last element, which must be the largest and different from the first. A framed interval is called an empodio if it does not contain any shorter framed intervals.

As an example, consider the biosequence $(0, 3, 5, 4, 6, 2, 1, 7)$. The whole biosequence is a framed interval. However, it contains another framed interval $(3, 5, 4, 6)$ and therefore it is not an empodio. The framed interval $(3, 5, 4, 6)$ does not contain a shorter framed interval, so it is an empodio. Furthermore, it is the only empodio in that biosequence.

You are to write a program that, given a biosequence, finds all empodia (plural for empodio) in that biosequence.

Input

t - the number of test cases [$t \leq 20$], then t test cases follow. The first line of each test case contains a single integer M : the number of integers in the input biosequence. The following M lines contain the integers of the biosequence in the order of the sequence. Each of these M lines contains a single integer. In one test case, $1000000 \leq M \leq 1100000$. In all other test cases, $1 \leq M \leq 60000$. Additionally, in 50% of the test cases, $M \leq 2600$.

Output

The first line for each test case is to contain one integer H : the number of empodia in the input biosequence. The following H lines describe all empodia of the input biosequence in the order of appearance of the starting point in the biosequence. Each of these lines is to contain two integers A and B (in that order) separated by a space, where the A th element of the input biosequence is the first element of the empodio and the B th element of the input biosequence is the last element of the empodio.

Example

Input :

```
1
8
0
3
5
4
```

6
2
1
7

Output:

1
2 5

Added by: Roman Sol
Date: 2005-05-22
Time limit [s]: 40
Source limit [B]: 50000
Resource: IOI 2004

SPOJ Problem Set

368. Cobbled streets

Problem code: CSTREET

The municipal chronicals of an unbelievable lordly major town in a land far, far away tell the following story:

> Once upon a time the new crowned king Günther decided to visit all towns in his kingdom. The people of the unbelievable lordly major town expected that king Günther would like to see some of the most famous buildings in their town. For the lordly citizens it seemed necessary that all streets in the town that the king would have to use had to be cobbled with stone. Unfortunately the unbelievable lordly major town had not much money at that time as they used most of their savings to erect the highest cathedral the world had ever seen.

> Rumours were afloat that the real reason for their thriftiness was not that the town treasury was empty but that many people believed that king Günther came to the throne by deceiving his father king Erwin and that in his youth he made a pact with the devil. But anyway, the citizens of the unbelievable lordly major town decided to pave only as much streets as were absolutely necessary to reach every major building.

> Can you help the citizens of the unbelievable lordly major town to find out which streets should be paved?

>It might be useful to know that all major buildings are either at the end of a street or at an intersection. In addition to that you can assume that all buildings are connected by the given streets.

Input

t [number of testcases ($1 \leq t \leq 100$)]

> p [price to pave one furlong of street (positive integer)]

> n [number of main buildings in the town ($1 \leq n \leq 1000$)]

> m [number of streets in the town ($1 \leq m \leq 300000$)]

> a b c [street from building a to building b with length c (lengths are given in furlong and the buildings are numbered from 1 to n)]

Output

For each testcase output the price of the cheapest possibility to reach all main buildings in the city on paved streets. You can assume that the result will be smaller than 2^{32} .

Example

Input :

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

3 4 3

Output:

12

Added by: Simon Gog

Date: 2005-05-24

Time limit [s]: 5

Source limit [B]: 32211

Resource: Ulm Algorithm Course SoSe 2005

SPOJ Problem Set

369. Math I

Problem code: MATH1

You are given n integers a_1, a_2, \dots, a_n ($0 \leq a_i \leq n$). The sum $a_1 + a_2 + \dots + a_n$ does not exceed n . Your task is to find n other integers x_1, x_2, \dots, x_n (note that x_i may be negative numbers) satisfying the following conditions:

- $(x_i - x_{i+1} + a_{i+1} = 0)$ or $(x_i - x_{i+1} + a_{i+1} = 1)$ for $i=1..n-1$
- $(x_n - x_1 + a_1 = 0)$ or $(x_n - x_1 + a_1 = 1)$
- $|x_1| + |x_2| + \dots + |x_n|$ is minimized

Input

The first line of the input file contains an integer t representing the number of test cases ($t \leq 20$). Then t test cases follow. Each test case has the following form:

- The first line contains n ($1 \leq n \leq 1000$)
- The second line contains n integers a_1, a_2, \dots, a_n separated by single spaces

Output

For each test case output a single value: the minimum value of $|x_1| + |x_2| + \dots + |x_n|$

Example

Input:

```
2
4
2 1 0 0
5
0 1 2 2 0
```

Output:

```
1
3
```

Output Details:

In the former case, the optimal solution is $(x_1=0, x_2=0, x_3=0, x_4=-1)$

In the latter case, the optimal solution is $(x_1=-1, x_2=-1, x_3=0, x_4=1, x_5=0)$

Added by: Ngo Minh Duc

Date: 2005-05-25

Time limit [s]: 20

Source limit [B]: 50000

SPOJ Problem Set

370. Ones and zeros

Problem code: ONEZERO

Certain positive integers have their decimal representation consisting only of ones and zeros, and having at least one digit one, e.g. 101. If a positive integer does not have such a property, one can try to multiply it by some positive integer to find out whether the product has this property.

Input

Number K of test cases
in each of the next K lines there is one integer n ($1 \leq n \leq 20000$)

Output

For each test case, your program should compute the smallest multiple of the number n consisting only of digits 1 and 0 (beginning with 1).

Example

Input:

```
3
17
11011
17
```

Output:

```
11101
11011
11101
```

Added by: Pawel Dobrzycki
Date: 2005-05-26
Time limit [s]: 8
Source limit [B]: 4096
Resource: II Polish Olympiad in Informatics, Ist Stage

SPOJ Problem Set

371. Boxes

Problem code: BOXES

There are n boxes on the circle. The boxes are numbered from 1 to n ($1 \leq n \leq 1000$) in clock wise order. There are balls in the boxes, and the number of all the balls in the boxes is not greater than n .

The balls should be displaced in such a way that in each box there remains no more than one ball. In one move we can shift a ball from one box to one of its neighboring boxes.

Write a program that: reads from the standard input the number of boxes n and the arrangement of balls in the boxes, computes the minimal number of moves necessary to displace the balls in such a way that in each box there remains no more than one ball, writes the result in the standard output.

Input

The first line of the input file contains an integer t representing the number of test cases ($t \leq 20$). Then t test cases follows. Each test case has the following form:

- The first line contains one positive integer n - the number of boxes
- The second line contains n nonnegative integer separated by single spaces. The i -th number is the number of balls in the i -th box.

Output

For each test case, output one nonnegative integer - the number of moves necessary to displace the balls in such a way that in each box there remains no more than one ball.

Example

Input:

```
1
12
0 0 2 4 3 1 0 0 0 0 0 1
```

Output:

```
19
```

Added by: Ngo Minh Duc
Date: 2005-05-31
Time limit [s]: 50
Source limit [B]: 50000
Resource: III Polish Olympiad in Informatics, stage 3

SPOJ Problem Set

10000. Simple Numbers Conversion

Problem code: TCONNUM

Every integer number n is represented in positional number system of base r by a sequence of digits $0 \leq d_i < r$, so the value is equal to:

$$n = d_0 + r * d_1 + r^2 * d_2 + r^3 * d_3 + \dots$$

Your task is to convert a given number in r -base representation into s -base representation, for example: decimal 231 into binary 11100111. Assume that $r \leq 36$ and the digits are 0,1,2,3,4,5,6,7,8,9, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z.

Input

N [the number of series ≤ 1000]
 $n \ r \ s$ [$n \leq 10^{1000}$, $r, s \leq 36$]

Output

n [s -base representation of number n]

Text grouped in [] does not appear in the input and output file.

Example

Input:

```
3
231 10 2
ABC 15 10
XYZ 36 2
```

Output:

```
11100111
2427
1010101111111011
```

Added by: Michal Malafiejski
Date: 2004-10-13
Time limit [s]: 30
Source limit [B]:5000

SPOJ Problem Set

10010. Not So Fast Multiplication

Problem code: TMUL

Multiply the given numbers.

Input

n [the number of multiplications ≤ 1000]
l1 l2 [numbers to multiply (at most 10000 decimal digits each)]

Text grouped in [] does not appear in the input file.

Output

The results of multiplications.

Example

Input:
5
4 2
123 43
324 342
0 12
9999 12345

Output:
8
5289
110808
0
123437655

Warning: large Input/Output data, be careful with certain languages

Added by: Michal Malafiejski
Date: 2004-10-19
Time limit [s]: 30
Source limit [B]: 50000
Resource: PAL (thanks to Darek Dereniowski)
Copy of MUL problem with 30s time limit

SPOJ Problem Set

10011. The Turtles Shortest Path

Problem code: TSHPATH

Given a list of cities. Each direct connection between two cities has its transportation cost (an integer bigger than 0). The goal is to find the paths of minimum cost between pairs of cities. Assume that the cost of each path (which is the sum of costs of all direct connections belonging to this path) is at most 200000. The name of a city is a string containing characters a,...,z and is at most 10 characters long.

Input

```
s [the number of tests <= 10]
n [the number of cities <= 10000]
NAME [city name]
p [the number of neighbours of city NAME]
nr cost [nr - index of a city connected to NAME (the index of the first city is 1)]
      [cost - the transportation cost]
r [the number of paths to find <= 100]
NAME1 NAME2 [NAME1 - source, NAME2 - destination]
[empty line separating the tests]
```

Output

```
cost [the minimum transportation cost from city NAME1 to city NAME2 (one per line)]
```

Example

```
Input:
1
4
gdansk
2
2 1
3 3
bydgoszcz
3
1 1
3 1
4 4
torun
3
1 3
2 1
4 1
warszawa
2
2 4
3 1
2
gdansk warszawa
bydgoszcz warszawa
```

Output:

3

2

Warning: large Input/Output data, be careful with certain languages

Added by: Michal Malafiejski

Date: 2004-10-21

Time limit [s]: 60

Source limit [B]:50000

Resource: DASM Programming League 2003 (thanks to Darek Dereniowski)
a copy of SHPATH problem with 60s time limit

SPOJ Problem Set

10013. Searching the Graph

Problem code: TDBFS

For a given list of adjacent vertices of a graph and a chosen vertex v write down in the Depth First Search (DFS) or Breadth First Search (BFS) order all the vertices from the connected component of a graph containing v . Assume that the number of vertices of a graph is at most 1000.

Input

t [the number of graphs ≤ 100]

Graph:

n [$1 \leq n \leq 1000$ the number of graph vertices]

$i \ m \ a \ b \ c \dots$ [the list of m adjacent vertices to vertex i]

Any query is as follows: [not more than n queries]

$v \ i$

where $1 \leq v \leq n$ is the beginning vertex and $i = 0$ for DFS order and $i = 1$ for BFS order.

0 0 [at the end of the serie]

The list for isolated vertex a is $a \ 0$.

Output

graph i [test case, word *graph* is necessary]

$a \ b \ c \dots$ [the DFS or BFS order of all vertices]

Example

Input :

```
3
6
1 2 3 4
2 2 3 6
3 2 1 2
4 1 1
5 0
6 1 2
5 1
1 0
1 0
0 0
10
1 6 3 5 6 7 8 9
2 1 9
3 2 1 5
4 5 6 7 8 9 10
5 4 1 3 7 8
6 3 1 4 7
7 5 1 4 5 6 8
8 5 1 4 5 7 10
9 3 1 2 4
10 2 4 8
```

```

7 1
1 0
2 1
4 1
7 1
0 0
2
1 0
2 0
1 1
0 0
Output:
graph 1
5
1 3 2 6 4
1 3 2 6 4
graph 2
7 1 4 5 6 8 3 9 10 2
1 3 5 7 4 6 8 10 9 2
2 9 1 4 3 5 6 7 8 10
4 6 7 8 9 10 1 5 2 3
7 1 4 5 6 8 3 9 10 2
graph 3
1

```

Added by: Michal Malafiejski
 Date: 2004-10-22
 Time limit [s]: 30
 Source limit [B]:5000

SPOJ Problem Set

10021. Simple Numbers with Fractions Conversion

Problem code: TCNUMFL

Every integer number n is represented in positional number system of base r by a sequence of digits $0 \leq d_i < r$, decimal point ',' and fractional part, so the value is equal to:

$$n = d_0 + r * d_{-1} + r^2 * d_{-2} + r^3 * d_{-3} + \dots + r^{-l} * d_{-l} + r^{-2} * d_{-2} + r^{-3} * d_{-3} + \dots$$

Your task is to convert a given number in r -base representation into s -base representation with l digits after decimal point (no rounding - use floor), for example: decimal 231,5 into binary 11100111,1 with one digit after decimal point. Assume that $r \leq 36$ and the digits are 0,1,2,3,4,5,6,7,8,9, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z.

Input

N [the number of series ≤ 1000]
 $n \ r \ s \ l$ [$n < 36^{1000} + 1$, $r, s \leq 36$, $l \leq 1000$]

Output

n [s -base representation of number n]

Text grouped in [] does not appear in the input and output file.

Example

Input:

```
10
500,1 6 31 3
3866,DJ 22 27 1
EH75,L3 24 4 3
A73C,10B 13 27 2
6C6J,E483 22 6 2
JA,L 30 5 4
6,5A 20 31 2
1,C5 14 7 1
HD,6K 26 9 2
1001,011 2 10 3
```

Output:

```
5P,555
1M8H,H
301223231,320
14MB,25
1255211,35
4310,3222
6,8G
1,6
555,23
9,375
```

Added by: Piotr Piotrowski
Date: 2004-11-08
Time limit [s]: 30
Source limit [B]:50000

SPOJ Problem Set

10022. Enormous Input Test

Problem code: INTTEST

The purpose of this problem is to verify whether the method you are using to read input data is sufficiently fast to handle problems branded with the **enormous Input/Output** warning. You are expected to be able to process at least 1MB of input data per second at runtime.

Input

The input begins with two positive integers n k ($n, k \leq 10^7$). The next n lines of input contain one positive integer t_i , not greater than 10^9 , each.

Output

Write a single integer to output, denoting how many integers t_i are divisible by k .

Example

Input:

```
7 3
1
51
966369
7
9
999996
11
```

Output:

```
4
```

Added by: Adrian Kosowski
Date: 2004-11-09
Time limit [s]: 20
Source limit [B]: 50000
Resource: Idea put forward by Michael Mendelsohn

SPOJ Problem Set

10024. Longest Common Subsequence

Problem code: TLCS

For a given two words $\mathbf{x} = x_1x_2...x_n$ and $\mathbf{y} = y_1y_2...y_m$ find the longest common subsequence, i.e. $\mathbf{z} = z_1z_2...z_k$ such that every two consecutive elements of \mathbf{z} are equal to some two elements of \mathbf{x} : x_a, x_b , and \mathbf{y} : y_c, y_d where $a < b$ and $c < d$. Assume, that elements of words are letters 'a' - 'z' and $m, n \leq 1000$.

Input

N [the number of series ≤ 1000]

n \mathbf{x}

m \mathbf{y}

...

Output

case 1 Y [or N when no answer to this case]

d [the length of the lcs]

z_j p q [position of z_j in \mathbf{x} and in \mathbf{y} , respectively]

...

Text grouped in [] does not appear in the input and output file.

Example

Input:

```
3
5 ddacc
3 cac
7 cbbccbc
4 aaca
4 cbeb
5 fdceb
```

Output:

```
case 1 Y
2
a 3 2
c 4 3
case 2 N
case 3 Y
3
c 1 3
e 3 4
b 4 5
```

Score

```
2
```

Added by: Michal Malafiejski
Date: 2004-11-10
Time limit [s]: 12
Source limit [B]:50000

SPOJ Problem Set

10025. Sums in a Triangle (tutorial)

Problem code: SUMTRIAN

This is problem SUMITR without strict source limit.

Let us consider a triangle of numbers in which a number appears in the first line, two numbers appear in the second line etc. Develop a program which will compute the largest of the sums of numbers that appear on the paths starting from the top towards the base, so that:

- on each path the next number is located on the row below, more precisely either directly below or below and one place to the right;
- the number of rows is strictly positive, but less than 100;
- all numbers are positive integers between 0 and 99.

Input

In the first line integer n - the number of test cases (equal to about 1000). Then n test cases follow. Each test case starts with the number of lines which is followed by their content.

Output

For each test case write the determined value in a separate line.

Example

Input :

```
2
3
1
2 1
1 2 3
4
1
1 2
4 1 2
2 3 1 1
```

Output :

```
5
9
```

Warning: large Input/Output data, be careful with certain languages

Added by: Lukasz Kuszner
Date: 2004-11-10
Time limit [s]: 4
Source limit
[B]: 5000
Resource: 6-th International Olympiad In Informatics July 3-10. 1994. Stockholm - Sweden,
 Problem 1

SPOJ Problem Set

10051. Permutation generator

Problem code: TPERML

For each index of n element permutation print m subsequent permutations (in separate lines) in lexicographical order starting from the one pointed by index. Between outputs of subsequent tests there should be an empty line. Next permutation to the last one is the first one.

Input

t [number of tests ≤ 1000]

n index m [$2 \leq n \leq 100$ - number of elements in permutation, $0 \leq \text{index} < n!$ - index of the first permutation, $1 \leq m \leq 100$ - how many permutations to print]

Output

$p_1 p_2 \dots p_{(n-1)} p_n$ [permutations]

$p_1 p_2 \dots p_n p_{(n-1)}$

$p_1 p_2 \dots p_{(n-1)} p_n$ [permutations]

$p_1 p_2 \dots p_n p_{(n-1)}$

Example

Input:

```
12
2 1 1
3 3 3
4 16 3
4 5 9
2 1 1
2 1 1
3 5 1
5 91 7
2 1 1
5 100 7
3 5 1
2 1 1
```

Output:

```
2 1

2 3 1
3 1 2
3 2 1

3 4 1 2
3 4 2 1
4 1 2 3

1 4 3 2
2 1 3 4
2 1 4 3
```

2 3 1 4
2 3 4 1
2 4 1 3
2 4 3 1
3 1 2 4
3 1 4 2

2 1

2 1

3 2 1

4 5 1 3 2
4 5 2 1 3
4 5 2 3 1
4 5 3 1 2
4 5 3 2 1
5 1 2 3 4
5 1 2 4 3

2 1

5 1 4 2 3
5 1 4 3 2
5 2 1 3 4
5 2 1 4 3
5 2 3 1 4
5 2 3 4 1
5 2 4 1 3

3 2 1

2 1

Added by: Piotr Piotrowski
Date: 2004-11-25
Time limit [s]: 30
Source limit [B]:50000

SPOJ Problem Set

10052. Jordan canonical form

Problem code: KMSL4

M is a 3x3 matrix. Find matrices P and J that $M = PJP^{-1}$ and J is in Jordan canonical form. You may assume that all eigenvalues of M are integer and within $[-10, 10]$.

Input

```
S - number of series
a11 a12 a13
a21 a22 a23
a31 a32 a33 - the first matrix M

...
(the following matrices are separated by empty lines)
```

Output

```
p11 p12 p13
p21 p22 p23
p31 p32 p33 - matrix P for the first M
(empty line)

j11 j12 j13
j21 j22 j23
j31 j32 j33 - matrix J for the first M

(empty line separates consecutive solutions)
```

Example

Input:

```
2
1 0 0
0 2 0
0 0 3

2 0 -2
1 2 -1
0 0 0
```

Output:

```
1 0 0
0 1 0
0 0 1

1 0 0
0 2 0
0 0 3

1 0 1
0 1 1
1 0 0
```

0 0 0
0 2 1
0 0 2

Added by: Adam Nadolski
Date: 2004-11-30
Time limit [s]: 5
Source limit [B]:50000

SPOJ Problem Set

10053. Roots of polynomial

Problem code: KMSL4B

$p(x) = p_k x^k + \dots + p_0 x^0$ is a given polynomial of degree at most 20. Check whether all roots of $p()$ belong to the open unit disc $|z| < 1$ on the complex plain.

Input

First the number of polynomials appears. Then the data for the following polynomials follows in the consecutive lines. For each of them first the degree is given, then in the following line the coefficients p_0, p_1, \dots appear, separated by spaces.

Output

Each line of the output is the solution for the following polynomials. It should be '1' if the roots of $p()$ belong to the open unit disc, or '0' otherwise.

Example

Input :

```
2
2
1 2 1
2
0.5 1 1
```

Output :

```
0
1
```

Added by: Adam Nadolski

Date: 2004-12-03

Time limit [s]: 10

Source limit [B]: 50000

SPOJ Problem Set

10055. Convex Hull

Problem code: TCONVEX

Given a collection of points in the plane, find the convex polygon with smallest area such that each point is contained within (or on the boundary of) the polygon.

Observe that the vertices of such a polygon will be points from the given collection.

Input

One integer in the first line, stating the number of test cases, followed by a blank line. There will be not more than 15 tests.

For each test case, the first line is an integer n ($3 \leq n \leq 50000$) stating the number of points. Then n lines follow, each presenting the coordinate of a point, with two integers x, y ($-10000 \leq x, y \leq 10000$). There are no two points which have the same pair of coordinates and no 3 points lie on one straight line.

The test cases will be separated by a single blank line.

Output

For each test case, write one integer - the number of vertices of the convex polygon you've found.

Example

Input :

1

4

0 0

1 0

1 1

0 1

Output :

4

Added by: Thanh Vy Hua Le

Date: 2004-12-25

Time limit [s]: 20

Source limit [B]: 50000

Resource: Thanh Vy Hua Le

SPOJ Problem Set

10056. Fossil in the Ice

Problem code: TFOSS

A small group of archaeologists is working in the Antarctic. Their sensors have detected a number of caves in which there are interesting fossils. However, a thick layer of ice blocks the entrance to each cave. The archaeologists possess the equipment needed to burn a tunnel in the layer of ice, but the fuel is extremely expensive. In order to determine the size of each fossil the group has launched a number of probes through small bore-holes. Each probe which hit the fossil emits a signal consisting of its x and y coordinates. Your task is to determine the smallest possible size of the tunnel, which is equal to the maximal distance between any two probes (so that the fossil won't be damaged during extraction). The drilling equipment needs to be provided with the squared value of this distance.

Given the list of coordinates of the points containing probes, find the square of the maximal distance between any two probes.

Input

```
t [the number of tests <= 20]
[empty line]
n [the number of active probes <= 100000]
x1 y1 [coordinates of the first probe]
...
xn xn
[integer coordinates from -500000000 to 500000000]
[empty line]
[input for the next test cases...]
```

Text grouped in [] does not appear in the input file.

Output

```
o1 [the square of the maximal distance in the first set]
[output for the next test cases...]
```

Example

Input:

```
5

1
2 -3

4
0 0
-2 2
2 2
1 0

6
-4 2
2 2
```

```
5 0
0 5
6 1
-1 -1
```

```
10
0 0
5 1
9 2
12 3
14 4
15 5
16 7
17 10
18 14
19 19
```

```
10
2 -3
-1 2
0 5
-5 -1
-4 2
4 0
1 3
4 3
-3 -4
0 -2
```

Output:

```
0
16
101
722
98
```

Added by: Lukasz Wrona
Date: 2004-12-29
Time limit [s]: 5
Source limit [B]:50000

SPOJ Problem Set

10062. Armies

Problem code: ARMIES

Two enemy countries - *Bajtocja* and *Megabajtolandia* - are preparing for crucial war with each other. Each country has built an army consisting of some number of divisions, and each division consists of some number of soldiers. The way of waging the war, given by strategists from each country, consists of sending the division with the most man power to fight, i.e. starting from the most numerous division to the least.

Thus, first each country will send its division with the most man power. If one of these divisions has more soldiers than the other, then the war is over and the winner is the owner of the larger division. If the man power of each of the divisions sent is the same then all the soldiers will kill each other and the next most numerous division is sent to fight. The man powers of the second divisions decide the war if and only if they are not the same. If not, the battle is carried on in aforementioned way. If, at some moment, one army runs out of divisions and the second one does not, then the war is over and the first army is the loser. If both armies run out of divisions then the war is over and there is a draw.

Give the result of the war, without any blood and murder.

Write a program, which:

- reads from standard input the description of *Bajtocja's* and *Megabajtolandia's* army, respectively,
- computes the result of the war,
- writes it to standard output.

Input

The first line of input contains one integer D ($1 \leq D \leq 30$) meaning the number of test cases. The description of each test case contains 4 lines. In the first, there is one integer B ($1 \leq B \leq 50\,000$) meaning the number of divisions in *Bajtocja's* army. The second line contains B integers b_i ($1 \leq b_i \leq 1\,000\,000\,000$) (separated by single space) meaning the man power (the number of soldiers) of consecutive divisions of *Bajtocja's* army. In the third line, there is one integer M ($1 \leq M \leq 50\,000$) meaning the number of divisions of *Megabajtolandia's* army. The fourth line contains M integers m_i ($1 \leq m_i \leq 1\,000\,000\,000$) (separated by single space) meaning the man power of consecutive divisions of *Megabajtolandia's* army.

Output

For each test case, your program should write, in separate lines, exactly one word:

- "Bajtocja" in case the winner is *Bajtocja*,
- "Megabajtolandia" in case the winner is *Megabajtolandia*,
- "Draw" in case of a draw.

Example

Sample input:

```
3
3
1 3 4
3
4 4 1
4
2 5 3 4
3
5 6 4
4
6 1 2 5
4
5 2 6 1
```

Sample output:

```
Megabajtolandia
Megabajtolandia
Draw
```

Added by: Rafal Nowak

Date: 2005-02-07

Time limit [s]: 5

Source limit [B]:5000

Resource: Winter sparing in Poznan, Poznan 2005 (22th January)

SPOJ Problem Set

281. The Cursed Room

Problem code: MMATCH

There is a school trip being organized for kids. The hotel the group is staying in can offer them one big room with enough beds to suit any group of visitors, and several smaller rooms with B beds altogether. The children have heard many strange and frightening stories about the big room. That's why not even one of them wants to sleep in the big room. Furthermore not every kid would like to sleep in any bed.

Your goal is to assign B beds from the smaller rooms in such a way that the maximal number of children are happy (a child is happy when it gets to sleep in one of the beds it has selected).

Input

The first line contains a positive integer $t \leq 1000$ indicating the number of test cases. Each test case is an instance of the problem defined above. The first line of each test case is a pair of positive integers L and B (the number of children $L \leq 100$ and beds $B \leq 100$). The next lines contain a sequence of (c,b) pairs ending with two zeros. (c,b) means that the child c will be happy if it gets to sleep in bed b.

Output

For each test case print the maximal number of happy children.

Example

Input :

```
3
3 3
1 1
2 1
2 2
3 2
3 3
0 0
4 3
1 1
1 3
2 1
3 1
3 2
4 2
0 0
4 2
1 1
1 2
2 1
2 2
3 1
3 2
4 1
4 2
0 0
```

Output:

3
3
2

Added by: Tomasz Niedzwiecki

Date: 2005-01-18

Time limit [s]: 10

Source limit [B]:50000

SPOJ Problem Set

10072. Turbo Sort

Problem code: TSORT

Given the list of numbers, you are to sort them in non decreasing order.

Input

t – the number of numbers in list, then t lines follow [$t \leq 10^6$].
Each line contains one integer: N [$0 \leq N \leq 10^6$]

Output

Output given numbers in non decreasing order.

Example

Input:

```
5
5
3
6
7
1
```

Output:

```
1
3
5
6
7
```

Added by: Roman Sol
Date: 2005-03-14
Time limit [s]: 10
Source limit [B]:50000
Resource: ;)

SPOJ Problem Set

10075. Prime Intervals

Problem code: PRINT

In this problem you have to print all primes from given interval.

Input

t - the number of test cases, then t lines follows. [$t \leq 150$]

On each line are written two integers L and U separated by a blank. L - lower bound of interval, U - upper bound of interval. [$2 \leq L < U \leq 2147483647$] [$U-L \leq 1000000$].

Output

For each test case output must contain all primes from interval $[L; U]$ in increasing order.

Example

Input:

```
2
2 10
3 7
```

Output:

```
2
3
5
7
3
5
7
```

Added by: Roman Sol
Date: 2005-03-28
Time limit [s]: 20
Source limit [B]: 15000
Resource: ;)

SPOJ Problem Set

10083. Easy Sorting

Problem code: LEXISORT

Given is a list of words and a lexicographical ordering according to the ascii alphabet. Your task is to sort the words in increasing order.

Input

The first line contains the numbers of testcases k ($k < 100$). Every testcase consists of $n+1$ ($1 < n < 50000$) lines. Each line contains a string of 10 characters. The first line of each testcase consists of n .

Output

Output the sorted list of words.

Example

Input :

```
2
2
helloworld
worldhello
2
aaaaaaaaaa
Aaaaaaaaaa
```

Output :

```
helloworld
worldhello
Aaaaaaaaaa
aaaaaaaaaa
```

Added by: Simon Gog
Date: 2005-04-13
Time limit [s]: 8
Source limit [B]: 8083

SPOJ Problem Set

10090. Zig-Zag Permutation

Problem code: ZZPERM

In the following we will deal with nonempty words consists only of lower case letters 'a','b',..., 'j' and we will use the natural 'a' < 'b' < ... < 'j' ordering. Your task is to write a program that generates almost all zig-zag words (zig-zag permutations) from a given collection of letters. We say that a word $W=W(1)W(2)...W(n)$ is zig-zag iff $n = 1$ or $W(i) > W(i+1)$ and $W(j) < W(j+1)$ for all odd $0 < i < n$ and for all even $0 < j < n$ or $W(i) > W(i+1)$ and $W(j) < W(j+1)$ for all even $0 < i < n$ and for all odd $0 < j < n$. For example: "aabcc" is not zig-zag, "acacb" is zig-zag, "cac" is zig-zag, "abababc" is not zig-zag. If you imagine all possible zig-zag permutations of a word in increasing lexicographic order, you can assign a serial number (rank) to each one. For example: the word "aabcc" generates the sequence: 1 \leftrightarrow "acacb", 2 \leftrightarrow "acbca", 3 \leftrightarrow "bacac", 4 \leftrightarrow "bcaca", 5 \leftrightarrow "cabac", 6 \leftrightarrow "cacab".

Input

The input file consists several test cases. Each case contains a word (W) not longer than 64 letters and one positive number (D). The letters of each word are in increasing order. Input terminated by EOF.

Output

For each case in the input file, the output file must contain all of the zig-zag permutations of W whose zig-zag serial is divisible by D, in increasing lexicographic order - one word per line. In the next line you have to print the total number of zig-zag permutations of W. There is no case that produces more than 365 lines of output. Print an empty line after each case.

Example

Input:

```
j 1
abc 2
aaabc 1
aaabb 2
aaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbccdd 123456
```

Output:

```
j
1

bac
cab
4

abaca
acaba
2
```

1

babacbcabacabadabababababababababadab
213216

Added by: Csaba Noszaly
Date: 2005-05-05
Time limit [s]: 18
Source limit [B]:12345
Resource: Folklore

SPOJ Problem Set

10098. Divisors

Problem code: DIV

Let N be a positive integer. In theory it is easy to decide if $d(N)$ (the number of positive divisors of N including 1 and N) is prime or not. Your task is just a little bit harder: compute all N in $[1, 10^6]$ for which $d(N) = p \cdot q$ where p and q distinct primes.

Input

There is no input for this problem.

Output

To make the problem less io related write out only every 9-th of them, one per line.

Output:

50
99
162
...
999524
999728
999927

Added by: Csaba Noszaly
Date: 2005-05-16
Time limit [s]: 9
Source limit [B]: 3333
Resource: Folklore.

SPOJ Problem Set

10099. Just for Fun (Easy)

Problem code: J4FUN

birds

Puzzle ID: birds

Ten birds sit on a clothes line. We shoot and kill one of them. How many birds remain on the clothes line?

The answer for this puzzle consists of two lines, containing respectively:

- the ID of this puzzle
- one number: the number of birds that remain on the clothes line

bus

Puzzle ID: bus

A bus was travelling with less than 100 passengers. At stop A, exactly three quarters of the passengers got off and 7 passengers got on the bus. The same thing happened at next two stops, B and C. How many people got off at the stop C?

The answer for this puzzle consists of two lines, containing respectively:

- the ID of this puzzle
- the number of people getting off at C

palindrome

Puzzle ID: palindrome

Suppose we write dates in the MMDDYYYY format. In this format, the 2nd of October 2001 is a palindrome (a string equal to its reverse): 10022001. Find the previous date that yields a palindrome in this format.

The answer for this puzzle consists of two lines, containing respectively:

- the ID of this puzzle
- the 8-digit string

cube

Puzzle ID: cube

You have a cube $N \times N \times N$. How many straight cuts are necessary to cut it into N^3 cubes of size $1 \times 1 \times 1$? You may arrange the pieces in any way you like before making each cut.

a) Solve for $N=3$

b) Solve for $N=4$

The answer for this puzzle consists of three lines, containing respectively:

- the ID of this puzzle
- the number of cuts from part a)
- the number of cuts from part b)

girl1

Puzzle ID: girl1

In a two-child family, one child is a boy.

What is the probability that the other child is a girl?

The answer for this puzzle consists of two lines, containing respectively:

- the ID of this puzzle
- the answer in the form a/b (where a, b are relatively prime)

girl2

Puzzle ID: girl2

In an unnamed overpopulated country the rulers agreed on a new law: Each woman may have as many

children as she wants to, until she gives birth to a girl. After that, she may have no more children. Assume that the law will never be broken. All families will have as many children as they are (physically and legally) able to. On each birth either one boy or one girl is born with equal chances. In the current population the ratio males:females is 1:1. What will happen in the next 100 years?

- A) The ratio of males to females will go up
- B) The ratio of males to females will stay the same
- C) The ratio of males to females will go down

The answer for this puzzle consists of two lines, containing respectively:

- the ID of this puzzle
- the uppercase letter corresponding to the correct answer

statements

Puzzle ID: statements

Given is a list with 2004 statements:

1. Exactly one statement on this list is false.
2. Exactly two statements on this list are false.
3. Exactly three statements on this list are false.

...

2004. Exactly 2004 statements on this list are false.

- a) Determine which statements are true.
- b) Replace "exactly" by "at least". Again, determine which statements are true.

The answer for this puzzle consists of three lines, containing respectively:

- the ID of this puzzle
- the encoded answer from part a)
- the encoded answer from part b)

How to encode the answer? If no statements are true, write the word 'NONE' (without the quotes). Otherwise take the set of true statements and write it as a set of ranges. E.g. the set {1,2,3,7,9,100,101} is encoded as 1-3,7,9,100-101

letters

Puzzle ID: letters

How many letters does the _shortest_ correct answer to this puzzle contain?

The answer for this puzzle consists of two lines, containing respectively:

- the ID of this puzzle
- your exact answer

century

Puzzle ID: century

The twentieth century ended on 31. 12. 2000, which was a Sunday. Looking into the future, on which days of the week won't any century ever end?

Remember that leap years are those divisible by 400 plus those divisible by 4 but not by 100. (1996 was a leap year, so was 2000, but 2100 won't be a leap year and neither will 2047.)

The answer for this puzzle consists of two lines, containing respectively:

- the ID of this puzzle
- the days of the week on which no century will ever end

The exact form of the answer is a comma-separated list of three-letter abbreviations of the days in the order in which they appear in a week. E.g. if the answer were Monday, Tuesday and Wednesday, write the string 'Mon,Tue,Wed' (without the quotes).

Input

There is no input for given problem.

Output

Output answers for each puzzle described below in the order they was described.

Example

Output:

```
birds
100
bus
10000
...
```

Added by: Roman Sol
Date: 2005-05-18
Time limit [s]: 1
Source limit [B]:10000
Resource: IPSC 2005

SPOJ Problem Set

10102. Divisors 2

Problem code: DIV2

Let N be a positive integer and $d(N)$ be the number of positive divisors of N including 1 and N . Your task is to compute all N in $[1, 10^6]$ for which $d(N) > 3$ and if M divides N then $d(M)$ divides $d(N)$ too.

Input

None.

Output

To make the problem less output related write out only every 108-th of them, one per line.

Example

Output:

```
267
511
753
...
999579
999781
999977
```

Added by: Csaba Noszaly
Date: 2005-05-24
Time limit [s]: 9
Source limit [B]: 3333
Resource: Folklore