目录

[1001. A+B Format (20) 6](#_Toc534654921)

[1002. A+B for Polynomials (25) 7](#_Toc534654922)

[1003. Emergency (25) 8](#_Toc534654923)

[1004. Counting Leaves (30) 9](#_Toc534654924)

[1005. Spell It Right (20) 10](#_Toc534654925)

[1006. Sign In and Sign Out (25) 11](#_Toc534654926)

[1007. Maximum Subsequence Sum (25) 12](#_Toc534654927)

[1008. Elevator (20) 13](#_Toc534654928)

[1009. Product of Polynomials (25) 14](#_Toc534654929)

[1010. Radix (25) 15](#_Toc534654930)

[1011. World Cup Betting (20) 16](#_Toc534654931)

[1012. The Best Rank (25) 17](#_Toc534654932)

[1013. Battle Over Cities (25) 19](#_Toc534654933)

[1014. Waiting in Line (30) 21](#_Toc534654934)

[1015. Reversible Primes (20) 22](#_Toc534654935)

[1016. Phone Bills (25) 24](#_Toc534654936)

[1017. Queueing at Bank (25) 26](#_Toc534654937)

[1018. Public Bike Management (30) 28](#_Toc534654938)

[1019. General Palindromic Number (20) 30](#_Toc534654939)

[1020. Tree Traversals (25) 32](#_Toc534654940)

[1021. Deepest Root (25) 33](#_Toc534654941)

[1022. Digital Library (30) 34](#_Toc534654942)

[1023. Have Fun with Numbers (20) 38](#_Toc534654943)

[1024. Palindromic Number (25) 38](#_Toc534654944)

[1025. PAT Ranking (25) 40](#_Toc534654945)

[1026. Table Tennis (30) 42](#_Toc534654946)

[1027. Colors in Mars (20) 45](#_Toc534654947)

[1028. List Sorting (25) 46](#_Toc534654948)

[1029. Median (25) 48](#_Toc534654949)

[1030. Travel Plan (30) 49](#_Toc534654950)

[1031. Hello World for U (20) 51](#_Toc534654951)

[1032. Sharing (25) 52](#_Toc534654952)

[1033. To Fill or Not to Fill (25) 54](#_Toc534654953)

[1034. Head of a Gang (30) 55](#_Toc534654954)

[1035. Password (20) 58](#_Toc534654955)

[1036. Boys vs Girls (25) 59](#_Toc534654956)

[1037. Magic Coupon (25) 61](#_Toc534654957)

[1038. Recover the Smallest Number (30) 62](#_Toc534654958)

[1039. Course List for Student (25) 63](#_Toc534654959)

[1040. Longest Symmetric String (25) 65](#_Toc534654960)

[1041. Be Unique (20) 65](#_Toc534654961)

[1042. Shuffling Machine (20) 66](#_Toc534654962)

[1043. Is It a Binary Search Tree (25) 68](#_Toc534654963)

[1044. Shopping in Mars (25) 69](#_Toc534654964)

[1045. Favorite Color Stripe (30) 71](#_Toc534654965)

[1046. Shortest Distance (20) 72](#_Toc534654966)

[1047. Student List for Course (25) 73](#_Toc534654967)

[1048. Find Coins (25) 76](#_Toc534654968)

[1049. Counting Ones (30) 77](#_Toc534654969)

[1050. String Subtraction (20) 78](#_Toc534654970)

[1051. Pop Sequence (25) 79](#_Toc534654971)

[1052. Linked List Sorting (25) 80](#_Toc534654972)

[1053. Path of Equal Weight (30) 82](#_Toc534654973)

[1054. The Dominant Color (20) 84](#_Toc534654974)

[1055. The World's Richest (25) 85](#_Toc534654975)

[1056. Mice and Rice (25) 88](#_Toc534654976)

[1057. Stack (30) 89](#_Toc534654977)

[解题思路 100](#_Toc534654978)

[1.树状数组+二分查找 100](#_Toc534654979)

[2.分桶法（分治，分层HASH，平方分割）本人快乐的原创 103](#_Toc534654980)

[3.线段树（分治）有种杀鸡用牛刀的感觉 105](#_Toc534654981)

[4.Prioriry Queue On Multiset(红黑树是支持插入与删除的堆)真正的牛刀 108](#_Toc534654982)

[5. 平衡搜索树——Treap（我觉得从前维护两个set求中间值是牛刀，那时我还小） 112](#_Toc534654983)

[1058. A+B in Hogwarts (20) 118](#_Toc534654984)

[1059. Prime Factors (25) 119](#_Toc534654985)

[1060. Are They Equal (25) 120](#_Toc534654986)

[1061. Dating (20) 121](#_Toc534654987)

[1062. Talent and Virtue (25) 122](#_Toc534654988)

[1063. Set Similarity (25) 125](#_Toc534654989)

[1064. Complete Binary Search Tree (30) 126](#_Toc534654990)

[1065. A+B and C (64bit) (20) 127](#_Toc534654991)

[1066. Root of AVL Tree (25) 128](#_Toc534654992)

[1067. Sort with Swap(0,\*) (25) 130](#_Toc534654993)

[1068. Find More Coins (30) 131](#_Toc534654994)

[1069. The Black Hole of Numbers (20) 132](#_Toc534654995)

[1070. Mooncake (25) 133](#_Toc534654996)

[1071. Speech Patterns (25) 135](#_Toc534654997)

[1072. Gas Station (30) 136](#_Toc534654998)

[1073. Scientific Notation (20) 138](#_Toc534654999)

[1074. Reversing Linked List (25) 139](#_Toc534655000)

[1075. PAT Judge (25) 140](#_Toc534655001)

[1076. Forwards on Weibo (30) 143](#_Toc534655002)

[1077. Kuchiguse (20) 144](#_Toc534655003)

[1078. Hashing (25) 146](#_Toc534655004)

[1079. Total Sales of Supply Chain (25) 147](#_Toc534655005)

[1080. Graduate Admission (30) 148](#_Toc534655006)

[1081. Rational Sum (20) 150](#_Toc534655007)

[1082. Read Number in Chinese (25) 152](#_Toc534655008)

[1083. List Grades (25) 153](#_Toc534655009)

[1084. Broken Keyboard (20) 155](#_Toc534655010)

[1085. Perfect Sequence (25) 156](#_Toc534655011)

[1086. Tree Traversals Again (25) 156](#_Toc534655012)

[1087. All Roads Lead to Rome (30) 158](#_Toc534655013)

[1088. Rational Arithmetic (20) 160](#_Toc534655014)

[1089. Insert or Merge (25) 161](#_Toc534655015)

[1090. Highest Price in Supply Chain (25) 163](#_Toc534655016)

[1091. Acute Stroke (30) 164](#_Toc534655017)

[1092. To Buy or Not to Buy (20) 166](#_Toc534655018)

[1093. Count PAT's (25) 167](#_Toc534655019)

[1094. The Largest Generation (25) 168](#_Toc534655020)

[1095. Cars on Campus (30) 170](#_Toc534655021)

[1096. Consecutive Factors (20) 172](#_Toc534655022)

[1097. Deduplication on a Linked List (25) 173](#_Toc534655023)

[1098. Insertion or Heap Sort (25) 175](#_Toc534655024)

[1099. Build A Binary Search Tree (30) 176](#_Toc534655025)

[1100. Mars Numbers (20) 178](#_Toc534655026)

[1101. Quick Sort (25) 179](#_Toc534655027)

[1102. Invert a Binary Tree (25) 181](#_Toc534655028)

[1103. Integer Factorization (30) 182](#_Toc534655029)

[1104. Sum of Number Segments (20) 183](#_Toc534655030)

[1105. Spiral Matrix (25) 184](#_Toc534655031)

[1106. Lowest Price in Supply Chain (25) 185](#_Toc534655032)

[1107. Social Clusters (30) 187](#_Toc534655033)

[1108. Finding Average (20) 188](#_Toc534655034)

[1109. Group Photo (25) 190](#_Toc534655035)

[1110. Complete Binary Tree (25) 192](#_Toc534655036)

[1111. Online Map (30) 193](#_Toc534655037)

[1112. Stucked Keyboard (20) 196](#_Toc534655038)

[1113. Integer Set Partition (25) 197](#_Toc534655039)

[1114. Family Property (25) 198](#_Toc534655040)

[1116. Come on! Let's C (20) 200](#_Toc534655041)

[1117. Eddington Number(25) 202](#_Toc534655042)

[1118. Birds in Forest (25) 203](#_Toc534655043)

[1119. Pre- and Post-order Traversals (30) 204](#_Toc534655044)

[1120. Friend Numbers (20) 206](#_Toc534655045)

[1121. Damn Single (25) 207](#_Toc534655046)

[1122. Hamiltonian Cycle (25) 208](#_Toc534655047)

[1123. Is It a Complete AVL Tree (30) 210](#_Toc534655048)

[1124. Raffle for Weibo Followers (20) 212](#_Toc534655049)

[1125. Chain the Ropes (25) 214](#_Toc534655050)

[1126. Eulerian Path (25) 215](#_Toc534655051)

[1127. ZigZagging on a Tree (30) 218](#_Toc534655052)

[1128. N Queens Puzzle (20) 219](#_Toc534655053)

[1129. Recommendation System (25) 221](#_Toc534655054)

[1130. Infix Expression (25) 222](#_Toc534655055)

[1131. Subway Map (30) 225](#_Toc534655056)

[1132. Cut Integer (20) 228](#_Toc534655057)

[1133. Splitting A Linked List (25) 229](#_Toc534655058)

[1134. Vertex Cover (25) 231](#_Toc534655059)

[1135. Is It A Red-Black Tree (30) 233](#_Toc534655060)

[1136. A Delayed Palindrome (20) 234](#_Toc534655061)

[1137. Final Grading (25) 236](#_Toc534655062)

[1138. Postorder Traversal (25) 238](#_Toc534655063)

[1139. First Contact (30) 239](#_Toc534655064)

[1140 Look-and-say Sequence（20分） 242](#_Toc534655065)

[Input Specification: 243](#_Toc534655066)

[Output Specification: 243](#_Toc534655067)

[Sample Input: 243](#_Toc534655068)

[Sample Output: 243](#_Toc534655069)

[1141 PAT Ranking of Institutions（25 分） 243](#_Toc534655070)

[Input Specification: 243](#_Toc534655071)

[Output Specification: 244](#_Toc534655072)

[Sample Input: 244](#_Toc534655073)

[Sample Output: 244](#_Toc534655074)

[1142 Maximal Clique（25 分） 244](#_Toc534655075)

[Input Specification: 245](#_Toc534655076)

[Output Specification: 245](#_Toc534655077)

[Sample Input: 245](#_Toc534655078)

[Sample Output: 246](#_Toc534655079)

[1143 Lowest Common Ancestor（30 分） 246](#_Toc534655080)

[Input Specification: 246](#_Toc534655081)

[Output Specification: 246](#_Toc534655082)

[Sample Input: 246](#_Toc534655083)

[Sample Output: 247](#_Toc534655084)

[1144 The Missing Number（20 分） 247](#_Toc534655085)

[Input Specification: 247](#_Toc534655086)

[Output Specification: 247](#_Toc534655087)

[Sample Input: 247](#_Toc534655088)

[Sample Output: 248](#_Toc534655089)

[1145 Hashing - Average Search Time（25 分） 248](#_Toc534655090)

[Input Specification: 248](#_Toc534655091)

[Output Specification: 248](#_Toc534655092)

[Sample Input: 249](#_Toc534655093)

[Sample Output: 249](#_Toc534655094)

[1146 Topological Order（25 分） 249](#_Toc534655095)

[Input Specification: 249](#_Toc534655096)

[Output Specification: 250](#_Toc534655097)

[Sample Input: 250](#_Toc534655098)

[Sample Output: 250](#_Toc534655099)

[1147 Heaps（30 分） 250](#_Toc534655100)

[Input Specification: 251](#_Toc534655101)

[Output Specification: 251](#_Toc534655102)

[Sample Input: 251](#_Toc534655103)

[Sample Output: 251](#_Toc534655104)

[1148 Werewolf - Simple Version（20 分） 251](#_Toc534655105)

[Input Specification: 252](#_Toc534655106)

[Output Specification: 252](#_Toc534655107)

[Sample Input 1: 252](#_Toc534655108)

[Sample Output 1: 252](#_Toc534655109)

[Sample Input 2: 253](#_Toc534655110)

[Sample Output 2 (the solution is not unique): 253](#_Toc534655111)

[Sample Input 3: 253](#_Toc534655112)

[Sample Output 3: 253](#_Toc534655113)

[1149 Dangerous Goods Packaging（25 分） 253](#_Toc534655114)

[Input Specification: 254](#_Toc534655115)

[Output Specification: 254](#_Toc534655116)

[Sample Input: 254](#_Toc534655117)

[Sample Output: 254](#_Toc534655118)

[1150 Travelling Salesman Problem（25 分） 255](#_Toc534655119)

[Input Specification: 255](#_Toc534655120)

[Output Specification: 255](#_Toc534655121)

[Sample Input: 255](#_Toc534655122)

[Sample Output: 256](#_Toc534655123)

[1151 LCA in a Binary Tree（30 分） 256](#_Toc534655124)

[Input Specification: 256](#_Toc534655125)

[Output Specification: 257](#_Toc534655126)

[Sample Input: 257](#_Toc534655127)

[Sample Output: 257](#_Toc534655128)

[1152 Google Recruitment （20 分） 257](#_Toc534655129)

[Input Specification: 259](#_Toc534655130)

[Output Specification: 259](#_Toc534655131)

[Sample Input 1: 259](#_Toc534655132)

[Sample Output 1: 259](#_Toc534655133)

[Sample Input 2: 259](#_Toc534655134)

[Sample Output 2: 259](#_Toc534655135)

[1153 Decode Registration Card of PAT （25 分） 259](#_Toc534655136)

[Input Specification: 260](#_Toc534655137)

[Output Specification: 260](#_Toc534655138)

[Sample Input: 261](#_Toc534655139)

[Sample Output: 261](#_Toc534655140)

[1154 Vertex Coloring （25 分） 261](#_Toc534655141)

[Input Specification: 262](#_Toc534655142)

[Output Specification: 262](#_Toc534655143)

[Sample Input: 262](#_Toc534655144)

[Sample Output: 262](#_Toc534655145)

[1155 Heap Paths （30 分） 263](#_Toc534655146)

[Input Specification: 263](#_Toc534655147)

[Output Specification: 263](#_Toc534655148)

[Sample Input 1: 263](#_Toc534655149)

[Sample Output 1: 263](#_Toc534655150)

[Sample Input 2: 264](#_Toc534655151)

[Sample Output 2: 264](#_Toc534655152)

[Sample Input 3: 264](#_Toc534655153)

[Sample Output 3: 264](#_Toc534655154)

# 1001. A+B Format (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Calculate a + b and output the sum in standard format -- that is, the digits must be separated into groups of three by commas (unless there are less than four digits).

**Input**

Each input file contains one test case. Each case contains a pair of integers a and b where -1000000 <= a, b <= 1000000. The numbers are separated by a space.

**Output**

For each test case, you should output the sum of a and b in one line. The sum must be written in the standard format.

**Sample Input**

-1000000 9

**Sample Output**

-999,991

# 1002. A+B for Polynomials (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

This time, you are supposed to find A+B where A and B are two polynomials.

**Input**

Each input file contains one test case. Each case occupies 2 lines, and each line contains the information of a polynomial: K N1 aN1 N2 aN2 ... NK aNK, where K is the number of nonzero terms in the polynomial, Ni and aNi (i=1, 2, ..., K) are the exponents and coefficients, respectively. It is given that 1 <= K <= 10，0 <= NK < ... < N2 < N1 <=1000.

**Output**

For each test case you should output the sum of A and B in one line, with the same format as the input. Notice that there must be NO extra space at the end of each line. Please be accurate to 1 decimal place.

**Sample Input**

2 1 2.4 0 3.2

2 2 1.5 1 0.5

**Sample Output**

3 2 1.5 1 2.9 0 3.2

这道题不需要当没有项的时候输出0 0。

注意和多项式的项数不一定小于MAXK=10。

# 1003. Emergency (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

As an emergency rescue team leader of a city, you are given a special map of your country. The map shows several scattered cities connected by some roads. Amount of rescue teams in each city and the length of each road between any pair of cities are marked on the map. When there is an emergency call to you from some other city, your job is to lead your men to the place as quickly as possible, and at the mean time, call up as many hands on the way as possible.

**Input**

Each input file contains one test case. For each test case, the first line contains 4 positive integers: N (<= 500) - the number of cities (and the cities are numbered from 0 to N-1), M - the number of roads, C1 and C2 - the cities that you are currently in and that you must save, respectively. The next line contains N integers, where the i-th integer is the number of rescue teams in the i-th city. Then M lines follow, each describes a road with three integers c1, c2 and L, which are the pair of cities connected by a road and the length of that road, respectively. It is guaranteed that there exists at least one path from C1 to C2.

**Output**

For each test case, print in one line two numbers: the number of different shortest paths between C1 and C2, and the maximum amount of rescue teams you can possibly gather.  
All the numbers in a line must be separated by exactly one space, and there is no extra space allowed at the end of a line.

**Sample Input**

5 6 0 2

1 2 1 5 3

0 1 1

0 2 2

0 3 1

1 2 1

2 4 1

3 4 1

**Sample Output**

2 4

# 1004. Counting Leaves (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A family hierarchy is usually presented by a pedigree tree. Your job is to count those family members who have no child.

**Input**

Each input file contains one test case. Each case starts with a line containing 0 < N < 100, the number of nodes in a tree, and M (< N), the number of non-leaf nodes. Then M lines follow, each in the format:

ID K ID[1] ID[2] ... ID[K]

where ID is a two-digit number representing a given non-leaf node, K is the number of its children, followed by a sequence of two-digit ID's of its children. For the sake of simplicity, let us fix the root ID to be 01.

**Output**

For each test case, you are supposed to count those family members who have no child **for every seniority level** starting from the root. The numbers must be printed in a line, separated by a space, and there must be no extra space at the end of each line.

The sample case represents a tree with only 2 nodes, where 01 is the root and 02 is its only child. Hence on the root 01 level, there is 0 leaf node; and on the next level, there is 1 leaf node. Then we should output "0 1" in a line.

**Sample Input**

2 1

01 1 02

**Sample Output**

0 1

# 1005. Spell It Right (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a non-negative integer N, your task is to compute the sum of all the digits of N, and output every digit of the sum in English.

**Input Specification:**

Each input file contains one test case. Each case occupies one line which contains an N (<= 10100).

**Output Specification:**

For each test case, output in one line the digits of the sum in English words. There must be one space between two consecutive words, but no extra space at the end of a line.

**Sample Input:**

12345

**Sample Output:**

one five

# 1006. Sign In and Sign Out (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

At the beginning of every day, the first person who signs in the computer room will unlock the door, and the last one who signs out will lock the door. Given the records of signing in's and out's, you are supposed to find the ones who have unlocked and locked the door on that day.

**Input Specification:**

Each input file contains one test case. Each case contains the records for one day. The case starts with a positive integer M, which is the total number of records, followed by M lines, each in the format:

ID\_number Sign\_in\_time Sign\_out\_time

where times are given in the format HH:MM:SS, and ID number is a string with no more than 15 characters.

**Output Specification:**

For each test case, output in one line the ID numbers of the persons who have unlocked and locked the door on that day. The two ID numbers must be separated by one space.

Note: It is guaranteed that the records are consistent. That is, the sign in time must be earlier than the sign out time for each person, and there are no two persons sign in or out at the same moment.

**Sample Input:**

3

CS301111 15:30:28 17:00:10

SC3021234 08:00:00 11:25:25

CS301133 21:45:00 21:58:40

**Sample Output:**

SC3021234 CS301133

# 1007. Maximum Subsequence Sum (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a sequence of K integers { N1, N2, ..., NK }. A continuous subsequence is defined to be { Ni, Ni+1, ..., Nj } where 1 <= i <= j <= K. The *Maximum Subsequence* is the continuous subsequence which has the largest sum of its elements. For example, given sequence { -2, 11, -4, 13, -5, -2 }, its maximum subsequence is { 11, -4, 13 } with the largest sum being 20.

Now you are supposed to find the largest sum, together with the first and the last numbers of the maximum subsequence.

**Input Specification:**

Each input file contains one test case. Each case occupies two lines. The first line contains a positive integer K (<= 10000). The second line contains K numbers, separated by a space.

**Output Specification:**

For each test case, output in one line the largest sum, together with the first and the last numbers of the maximum subsequence. The numbers must be separated by one space, but there must be no extra space at the end of a line. In case that the maximum subsequence is not unique, output the one with the smallest indices i and j (as shown by the sample case). If all the K numbers are negative, then its maximum sum is defined to be 0, and you are supposed to output the first and the last numbers of the whole sequence.

**Sample Input:**

10

-10 1 2 3 4 -5 -23 3 7 -21

**Sample Output:**

10 1 4

此题要注意对于最大子列和为0的情况，就不是全是负数那种输出整个子列的第一个和最后一个值了

# 1008. Elevator (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The highest building in our city has only one elevator. A request list is made up with N positive numbers. The numbers denote at which floors the elevator will stop, in specified order. It costs 6 seconds to move the elevator up one floor, and 4 seconds to move down one floor. The elevator will stay for 5 seconds at each stop.

For a given request list, you are to compute the total time spent to fulfill the requests on the list. The elevator is on the 0th floor at the beginning and does not have to return to the ground floor when the requests are fulfilled.

**Input Specification:**

Each input file contains one test case. Each case contains a positive integer N, followed by N positive numbers. All the numbers in the input are less than 100.

**Output Specification:**

For each test case, print the total time on a single line.

**Sample Input:**

3 2 3 1

**Sample Output:**

41

# 1009. Product of Polynomials (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

This time, you are supposed to find A\*B where A and B are two polynomials.

**Input Specification:**

Each input file contains one test case. Each case occupies 2 lines, and each line contains the information of a polynomial: K N1 aN1 N2 aN2 ... NK aNK, where K is the number of nonzero terms in the polynomial, Ni and aNi (i=1, 2, ..., K) are the exponents and coefficients, respectively. It is given that 1 <= K <= 10, 0 <= NK < ... < N2 < N1 <=1000.

**Output Specification:**

For each test case you should output the product of A and B in one line, with the same format as the input. Notice that there must be NO extra space at the end of each line. Please be accurate up to 1 decimal place.

**Sample Input**

2 1 2.4 0 3.2

2 2 1.5 1 0.5

**Sample Output**

3 3 3.6 2 6.0 1 1.6

# 1010. Radix (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a pair of positive integers, for example, 6 and 110, can this equation 6 = 110 be true? The answer is "yes", if 6 is a decimal number and 110 is a binary number.

Now for any pair of positive integers N1 and N2, your task is to find the radix of one number while that of the other is given.

**Input Specification:**

Each input file contains one test case. Each case occupies a line which contains 4 positive integers:  
N1 N2 tag radix  
Here N1 and N2 each has no more than 10 digits. A digit is less than its radix and is chosen from the set {0-9, a-z} where 0-9 represent the decimal numbers 0-9, and a-z represent the decimal numbers 10-35. The last number "radix" is the radix of N1 if "tag" is 1, or of N2 if "tag" is 2.

**Output Specification:**

For each test case, print in one line the radix of the other number so that the equation N1 = N2 is true. If the equation is impossible, print "Impossible". If the solution is not unique, output the smallest possible radix.

**Sample Input 1:**

6 110 1 10

**Sample Output 1:**

2

**Sample Input 2:**

1 ab 1 2

**Sample Output 2:**

Impossible

# 1011. World Cup Betting (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

With the 2010 FIFA World Cup running, football fans the world over were becoming increasingly excited as the best players from the best teams doing battles for the World Cup trophy in South Africa. Similarly, football betting fans were putting their money where their mouths were, by laying all manner of World Cup bets.

Chinese Football Lottery provided a "Triple Winning" game. The rule of winning was simple: first select any three of the games. Then for each selected game, bet on one of the three possible results -- namely W for win, T for tie, and L for lose. There was an odd assigned to each result. The winner's odd would be the product of the three odds times 65%.

For example, 3 games' odds are given as the following:

W T L

1.1 2.5 1.7

1.2 3.0 1.6

4.1 1.2 1.1

To obtain the maximum profit, one must buy W for the 3rd game, T for the 2nd game, and T for the 1st game. If each bet takes 2 yuans, then the maximum profit would be (4.1\*3.0\*2.5\*65%-1)\*2 = 37.98 yuans (accurate up to 2 decimal places).

**Input**

Each input file contains one test case. Each case contains the betting information of 3 games. Each game occupies a line with three distinct odds corresponding to W, T and L.

**Output**

For each test case, print in one line the best bet of each game, and the maximum profit accurate up to 2 decimal places. The characters and the number must be separated by one space.

**Sample Input**

1.1 2.5 1.7

1.2 3.0 1.6

4.1 1.2 1.1

**Sample Output**

T T W 37.98

此题要求保留两位小数，指的是四舍五入，Python的.2f%是天然的四舍五入到第二位但是有可能浮点数的精度会不足，比如会以0.004999999998表示0.005这样四舍五入的结果就会有差别。做法是加上0.0001。C语言是直接舍掉，做法可以是加上0.0051

# 1012. The Best Rank (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

To evaluate the performance of our first year CS majored students, we consider their grades of three courses only: C - C Programming Language, M - Mathematics (Calculus or Linear Algebra), and E - English. At the mean time, we encourage students by emphasizing on their best ranks -- that is, among the four ranks with respect to the three courses and the average grade, we print the best rank for each student.

For example, The grades of C, M, E and A - Average of 4 students are given as the following:

StudentID C M E A

310101 98 85 88 90

310102 70 95 88 84

310103 82 87 94 88

310104 91 91 91 91

Then the best ranks for all the students are *No.1* since the 1st one has done the best in C Programming Language, while the 2nd one in Mathematics, the 3rd one in English, and the last one in average.

**Input**

Each input file contains one test case. Each case starts with a line containing 2 numbers N and M (<=2000), which are the total number of students, and the number of students who would check their ranks, respectively. Then N lines follow, each contains a student ID which is a string of 6 digits, followed by the three integer grades (in the range of [0, 100]) of that student in the order of C, M and E. Then there are M lines, each containing a student ID.

**Output**

For each of the M students, print in one line the best rank for him/her, and the symbol of the corresponding rank, separated by a space.

The priorities of the ranking methods are ordered as A > C > M > E. Hence if there are two or more ways for a student to obtain the same best rank, output the one with the highest priority.

If a student is not on the grading list, simply output "N/A".

**Sample Input**

5 6

310101 98 85 88

310102 70 95 88

310103 82 87 94

310104 91 91 91

310105 85 90 90

310101

310102

310103

310104

310105

999999

**Sample Output**

1 C

1 M

1 E

1 A

3 A

N/A

# 1013. Battle Over Cities (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

It is vitally important to have all the cities connected by highways in a war. If a city is occupied by the enemy, all the highways from/toward that city are closed. We must know immediately if we need to repair any other highways to keep the rest of the cities connected. Given the map of cities which have all the remaining highways marked, you are supposed to tell the number of highways need to be repaired, quickly.

For example, if we have 3 cities and 2 highways connecting city1-city2 and city1-city3. Then if city1 is occupied by the enemy, we must have 1 highway repaired, that is the highway city2-city3.

**Input**

Each input file contains one test case. Each case starts with a line containing 3 numbers N (<1000), M and K, which are the total number of cities, the number of remaining highways, and the number of cities to be checked, respectively. Then M lines follow, each describes a highway by 2 integers, which are the numbers of the cities the highway connects. The cities are numbered from 1 to N. Finally there is a line containing K numbers, which represent the cities we concern.

**Output**

For each of the K cities, output in a line the number of highways need to be repaired if that city is lost.

**Sample Input**

3 2 3

1 2

1 3

1 2 3

**Sample Output**

1

0

0

# 1014. Waiting in Line (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Suppose a bank has N windows open for service. There is a yellow line in front of the windows which devides the waiting area into two parts. The rules for the customers to wait in line are:

* The space inside the yellow line in front of each window is enough to contain a line with M customers. Hence when all the N lines are full, all the customers after (and including) the (NM+1)st one will have to wait in a line behind the yellow line.
* Each customer will choose the shortest line to wait in when crossing the yellow line. If there are two or more lines with the same length, the customer will always choose the window with the smallest number.
* Customer[i] will take T[i] minutes to have his/her transaction processed.
* The first N customers are assumed to be served at 8:00am.

Now given the processing time of each customer, you are supposed to tell the exact time at which a customer has his/her business done.

For example, suppose that a bank has 2 windows and each window may have 2 custmers waiting inside the yellow line. There are 5 customers waiting with transactions taking 1, 2, 6, 4 and 3 minutes, respectively. At 08:00 in the morning, customer1 is served at window1 while customer2 is served at window2. Customer3 will wait in front of window1 and customer4 will wait in front of window2. Customer5 will wait behind the yellow line.

At 08:01, customer1 is done and customer5 enters the line in front of window1 since that line seems shorter now. Customer2 will leave at 08:02, customer4 at 08:06, customer3 at 08:07, and finally customer5 at 08:10.

**Input**

Each input file contains one test case. Each case starts with a line containing 4 positive integers: N (<=20, number of windows), M (<=10, the maximum capacity of each line inside the yellow line), K (<=1000, number of customers), and Q (<=1000, number of customer queries).

The next line contains K positive integers, which are the processing time of the K customers.

The last line contains Q positive integers, which represent the customers who are asking about the time they can have their transactions done. The customers are numbered from 1 to K.

**Output**

For each of the Q customers, print in one line the time at which his/her transaction is finished, in the format HH:MM where HH is in [08, 17] and MM is in [00, 59]. Note that since the bank is closed everyday after 17:00, for those customers who cannot be served before 17:00, you must output "Sorry" instead.

**Sample Input**

2 2 7 5

1 2 6 4 3 534 2

3 4 5 6 7

**Sample Output**

08:07

08:06

08:10

17:00

Sorry

/\* 题目大意：程序模拟银行排队办业务。

共n个窗口 每个窗口外都有黄线 黄线内最多可以站m个人进行排队 共有k个人 剩下的k-m\*n个人就只能在黄线外等待。每当黄线内可以站人时，黄线外的人就进入需要排队的人最少的窗口进行排队（若存在多个这样的窗口，则选择窗口序号最小的）。17:00之前没能开始办理业务的 输出Sorry 否则输出办完业务的时间。

\*/

# 1015. Reversible Primes (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A *reversible prime* in any number system is a prime whose "reverse" in that number system is also a prime. For example in the decimal system 73 is a reversible prime because its reverse 37 is also a prime.

Now given any two positive integers N (< 105) and D (1 < D <= 10), you are supposed to tell if N is a reversible prime with radix D.

**Input Specification:**

The input file consists of several test cases. Each case occupies a line which contains two integers N and D. The input is finished by a negative N.

**Output Specification:**

For each test case, print in one line "Yes" if N is a reversible prime with radix D, or "No" if not.

**Sample Input:**

73 10

23 2

23 10

-2

**Sample Output:**

Yes

Yes

No

# 1016. Phone Bills (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A long-distance telephone company charges its customers by the following rules:

Making a long-distance call costs a certain amount per minute, depending on the time of day when the call is made. When a customer starts connecting a long-distance call, the time will be recorded, and so will be the time when the customer hangs up the phone. Every calendar month, a bill is sent to the customer for each minute called (at a rate determined by the time of day). Your job is to prepare the bills for each month, given a set of phone call records.

**Input Specification:**

Each input file contains one test case. Each case has two parts: the rate structure, and the phone call records.

The rate structure consists of a line with 24 non-negative integers denoting the toll (cents/minute) from 00:00 - 01:00, the toll from 01:00 - 02:00, and so on for each hour in the day.

The next line contains a positive number N (<= 1000), followed by N lines of records. Each phone call record consists of the name of the customer (string of up to 20 characters without space), the time and date (mm:dd:hh:mm), and the word "on-line" or "off-line".

For each test case, all dates will be within a single month. Each "on-line" record is paired with the chronologically next record for the same customer provided it is an "off-line" record. Any "on-line" records that are not paired with an "off-line" record are ignored, as are "off-line" records not paired with an "on-line" record. It is guaranteed that at least one call is well paired in the input. You may assume that no two records for the same customer have the same time. Times are recorded using a 24-hour clock.

**Output Specification:**

For each test case, you must print a phone bill for each customer.

Bills must be printed in alphabetical order of customers' names. For each customer, first print in a line the name of the customer and the month of the bill in the format shown by the sample. Then for each time period of a call, print in one line the beginning and ending time and date (dd:hh:mm), the lasting time (in minute) and the charge of the call. The calls must be listed in chronological order. Finally, print the total charge for the month in the format shown by the sample.

**Sample Input:**

10 10 10 10 10 10 20 20 20 15 15 15 15 15 15 15 20 30 20 15 15 10 10 10

10

CYLL 01:01:06:01 on-line

CYLL 01:28:16:05 off-line

CYJJ 01:01:07:00 off-line

CYLL 01:01:08:03 off-line

CYJJ 01:01:05:59 on-line

aaa 01:01:01:03 on-line

aaa 01:02:00:01 on-line

CYLL 01:28:15:41 on-line

aaa 01:05:02:24 on-line

aaa 01:04:23:59 off-line

**Sample Output:**

CYJJ 01

01:05:59 01:07:00 61 $12.10

Total amount: $12.10

CYLL 01

01:06:01 01:08:03 122 $24.40

28:15:41 28:16:05 24 $3.85

Total amount: $28.25

aaa 01

02:00:01 04:23:59 4318 $638.80

Total amount: $638.80

分析：

来自：http://www.th7.cn/Program/cp/201402/174424.shtml

　　模拟题，提交无数次WA，注意几点：

　　1.如果某人没有有效通话记录，则不输出该人的信息，在此WA15次，题目看了N遍也没出现啊。

　　2.通话时间钱的计算：假设我们计算time1到time2的账单；

　　　　　　　　　　　　(1)我们可以采用从起点（即00：00：00）开始计算，结果就是get\_money(time2) - get\_money(time1), 这样计算方便。

　　　　　　　　　　　　(2)我们也可以采用从time1开始递增直到time2， 这样比较烦。

　　3.有效的通话记录是指：如果某人的通话记录为1.on;2.on;3.off;，则其中1.on将被抛弃，匹配到2.on;3.off;。

# 1017. Queueing at Bank (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Suppose a bank has K windows open for service. There is a yellow line in front of the windows which devides the waiting area into two parts. All the customers have to wait in line behind the yellow line, until it is his/her turn to be served and there is a window available. It is assumed that no window can be occupied by a single customer for more than 1 hour.

Now given the arriving time T and the processing time P of each customer, you are supposed to tell the average waiting time of all the customers.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 numbers: N (<=10000) - the total number of customers, and K (<=100) - the number of windows. Then N lines follow, each contains 2 times: HH:MM:SS - the arriving time, and P - the processing time in minutes of a customer. Here HH is in the range [00, 23], MM and SS are both in [00, 59]. It is assumed that no two customers arrives at the same time.

Notice that the bank opens from 08:00 to 17:00. Anyone arrives early will have to wait in line till 08:00, and anyone comes too late (at or after 17:00:01) will not be served nor counted into the average.

**Output Specification:**

For each test case, print in one line the average waiting time of all the customers, in minutes and accurate up to 1 decimal place.

**Sample Input:**

7 3

07:55:00 16

17:00:01 2

07:59:59 15

08:01:00 60

08:00:00 30

08:00:02 2

08:03:00 10

**Sample Output:**

8.2

# 1018. Public Bike Management (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

There is a public bike service in Hangzhou City which provides great convenience to the tourists from all over the world. One may rent a bike at any station and return it to any other stations in the city.

The Public Bike Management Center (PBMC) keeps monitoring the real-time capacity of all the stations. A station is said to be in *perfect* condition if it is exactly half-full. If a station is full or empty, PBMC will collect or send bikes to adjust the condition of that station to perfect. And more, all the stations on the way will be adjusted as well.

When a problem station is reported, PBMC will always choose the shortest path to reach that station. If there are more than one shortest path, the one that requires the least number of bikes sent from PBMC will be chosen.

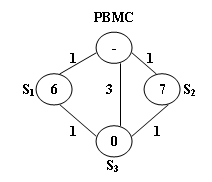
  
Figure 1

Figure 1 illustrates an example. The stations are represented by vertices and the roads correspond to the edges. The number on an edge is the time taken to reach one end station from another. The number written inside a vertex S is the current number of bikes stored at S. Given that the maximum capacity of each station is 10. To solve the problem at S3, we have 2 different shortest paths:

1. PBMC -> S1 -> S3. In this case, 4 bikes must be sent from PBMC, because we can collect 1 bike from S1 and then take 5 bikes to S3, so that both stations will be in perfect conditions.

2. PBMC -> S2 -> S3. This path requires the same time as path 1, but only 3 bikes sent from PBMC and hence is the one that will be chosen.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 4 numbers: Cmax (<= 100), always an even number, is the maximum capacity of each station; N (<= 500), the total number of stations; Sp, the index of the problem station (the stations are numbered from 1 to N, and PBMC is represented by the vertex 0); and M, the number of roads. The second line contains N non-negative numbers Ci (i=1,...N) where each Ci is the current number of bikes at Si respectively. Then M lines follow, each contains 3 numbers: Si, Sj, and Tij which describe the time Tij taken to move betwen stations Si and Sj. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print your results in one line. First output the number of bikes that PBMC must send. Then after one space, output the path in the format: 0->S1->...->Sp. Finally after another space, output the number of bikes that we must take back to PBMC after the condition of Sp is adjusted to perfect.

Note that if such a path is not unique, output the one that requires minimum number of bikes that we must take back to PBMC. The judge's data guarantee that such a path is unique.

**Sample Input:**

10 3 3 5

6 7 0

0 1 1

0 2 1

0 3 3

1 3 1

2 3 1

**Sample Output:**

3 0->2->3 0

注意自行车只能从前一个站点送到下一个或者直接送回PBMC，而不能往上一个站点输送。

不能采用贪心法求解，要用DFS来求解，比较出每一条路径的参数才行。

注意对于共同祖先的结点的搜索容易被忽视，因此每次搜索完后要初始化收集序列。

首先选择send最少的，send相同时选择take back最少的。

测试数据：

10 4 4 5

5 6 7 2

0 1 1

1 2 1

1 3 1

2 4 1

3 4 1

1 0->1->3->4 0

10 6 6 8

8 9 4 6 7 2

0 1 1

0 2 1

1 3 1

2 3 1

3 4 1

3 5 2

4 6 1

5 6 1

0 0->1->3->4->6 0

# 1019. General Palindromic Number (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A number that will be the same when it is written forwards or backwards is known as a Palindromic Number. For example, 1234321 is a palindromic number. All single digit numbers are palindromic numbers.

Although palindromic numbers are most often considered in the decimal system, the concept of palindromicity can be applied to the natural numbers in any numeral system. Consider a number N > 0 in base b >= 2, where it is written in standard notation with k+1 digits ai as the sum of (aibi) for i from 0 to k. Here, as usual, 0 <= ai < b for all i and ak is non-zero. Then N is palindromic if and only if ai = ak-i for all i. Zero is written 0 in any base and is also palindromic by definition.

Given any non-negative decimal integer N and a base b, you are supposed to tell if N is a palindromic number in base b.

**Input Specification:**

Each input file contains one test case. Each case consists of two non-negative numbers N and b, where 0 <= N <= 109 is the decimal number and 2 <= b <= 109 is the base. The numbers are separated by a space.

**Output Specification:**

For each test case, first print in one line "Yes" if N is a palindromic number in base b, or "No" if not. Then in the next line, print N as the number in base b in the form "ak ak-1 ... a0". Notice that there must be no extra space at the end of output.

**Sample Input 1:**

27 2

**Sample Output 1:**

Yes

1 1 0 1 1

**Sample Input 2:**

121 5

**Sample Output 2:**

No

4 4 1

注意N=0的情况，要输出：

No

0

# 1020. Tree Traversals (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Suppose that all the keys in a binary tree are distinct positive integers. Given the postorder and inorder traversal sequences, you are supposed to output the level order traversal sequence of the corresponding binary tree.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=30), the total number of nodes in the binary tree. The second line gives the postorder sequence and the third line gives the inorder sequence. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print in one line the level order traversal sequence of the corresponding binary tree. All the numbers in a line must be separated by exactly one space, and there must be no extra space at the end of the line.

**Sample Input:**

7

2 3 1 5 7 6 4

1 2 3 4 5 6 7

**Sample Output:**

4 1 6 3 5 7 2

# 1021. Deepest Root (25)

时间限制

1500 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A graph which is connected and acyclic can be considered a tree. The height of the tree depends on the selected root. Now you are supposed to find the root that results in a highest tree. Such a root is called *the deepest root*.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (<=10000) which is the number of nodes, and hence the nodes are numbered from 1 to N. Then N-1 lines follow, each describes an edge by given the two adjacent nodes' numbers.

**Output Specification:**

For each test case, print each of the deepest roots in a line. If such a root is not unique, print them in increasing order of their numbers. In case that the given graph is not a tree, print "Error: K components" where K is the number of connected components in the graph.

**Sample Input 1:**

5

1 2

1 3

1 4

2 5

**Sample Output 1:**

3

4

5

**Sample Input 2:**

5

1 3

1 4

2 5

3 4

**Sample Output 2:**

Error: 2 components

思路来自：https://www.2cto.com/px/201707/660235.html

题意：无环连通图也可以视为一棵树，选定图中任意一点作为根，如果这时候整个树的深度最大，则称其为 deepest root。

给定一个图，按升序输出所有 deepest root。如果给定的图有多个连通分量，则输出连通分量的数量

解题思路：先用并查集判断是不是一个联通分量，若是则后面的问题就是求树的直径，可以通过两次dfs求出树的直径。

首先从任意一点进行深搜，把其中距离最远的点记录下来，这些点是最长根，然后从这些最远点中任选一点再进行深度搜索，这个时候的所有最远点都是最长根。关键是第一遍dfs保存的最远点也是最长根。

首先，若第一遍dfs后的最远根不在从1开始的同一条支路上（他们的最近公共祖先是1），这是这些点必定都会是将来的最长根这很明显。

若第一遍dfs后的最远根在从1开始的同一条支路上，那么第二次dfs时就会漏掉同一支路上的最远根。

所以第一次与第二次dfs的合集才是最终结果

# 1022. Digital Library (30)

时间限制

1000 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A Digital Library contains millions of books, stored according to their titles, authors, key words of their abstracts, publishers, and published years. Each book is assigned an unique 7-digit number as its ID. Given any query from a reader, you are supposed to output the resulting books, sorted in increasing order of their ID's.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (<=10000) which is the total number of books. Then N blocks follow, each contains the information of a book in 6 lines:

* Line #1: the 7-digit ID number;
* Line #2: the book title -- a string of no more than 80 characters;
* Line #3: the author -- a string of no more than 80 characters;
* Line #4: the key words -- each word is a string of no more than 10 characters without any white space, and the keywords are separated by exactly one space;
* Line #5: the publisher -- a string of no more than 80 characters;
* Line #6: the published year -- a 4-digit number which is in the range [1000, 3000].

It is assumed that each book belongs to one author only, and contains no more than 5 key words; there are no more than 1000 distinct key words in total; and there are no more than 1000 distinct publishers.

After the book information, there is a line containing a positive integer M (<=1000) which is the number of user's search queries. Then M lines follow, each in one of the formats shown below:

* 1: a book title
* 2: name of an author
* 3: a key word
* 4: name of a publisher
* 5: a 4-digit number representing the year

**Output Specification:**

For each query, first print the original query in a line, then output the resulting book ID's in increasing order, each occupying a line. If no book is found, print "Not Found" instead.

**Sample Input:**

3

1111111

The Testing Book

Yue Chen

test code debug sort keywords

ZUCS Print

2011

3333333

Another Testing Book

Yue Chen

test code sort keywords

ZUCS Print2

2012

2222222

The Testing Book

CYLL

keywords debug book

ZUCS Print2

2011

6

1: The Testing Book

2: Yue Chen

3: keywords

4: ZUCS Print

5: 2011

3: blablabla

**Sample Output:**

1: The Testing Book

1111111

2222222

2: Yue Chen

1111111

3333333

3: keywords

1111111

2222222

3333333

4: ZUCS Print

1111111

5: 2011

1111111

2222222

3: blablabla

Not Found

# 1023. Have Fun with Numbers (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Notice that the number 123456789 is a 9-digit number consisting exactly the numbers from 1 to 9, with no duplication. Double it we will obtain 246913578, which happens to be another 9-digit number consisting exactly the numbers from 1 to 9, only in a different permutation. Check to see the result if we double it again!

Now you are suppose to check if there are more numbers with this property. That is, double a given number with k digits, you are to tell if the resulting number consists of only a permutation of the digits in the original number.

**Input Specification:**

Each input file contains one test case. Each case contains one positive integer with no more than 20 digits.

**Output Specification:**

For each test case, first print in a line "Yes" if doubling the input number gives a number that consists of only a permutation of the digits in the original number, or "No" if not. Then in the next line, print the doubled number.

**Sample Input:**

1234567899

**Sample Output:**

Yes

2469135798

# 1024. Palindromic Number (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A number that will be the same when it is written forwards or backwards is known as a Palindromic Number. For example, 1234321 is a palindromic number. All single digit numbers are palindromic numbers.

Non-palindromic numbers can be paired with palindromic ones via a series of operations. First, the non-palindromic number is reversed and the result is added to the original number. If the result is not a palindromic number, this is repeated until it gives a palindromic number. For example, if we start from 67, we can obtain a palindromic number in 2 steps: 67 + 76 = 143, and 143 + 341 = 484.

Given any positive integer N, you are supposed to find its paired palindromic number and the number of steps taken to find it.

**Input Specification:**

Each input file contains one test case. Each case consists of two positive numbers N and K, where N (<= 1010) is the initial numer and K (<= 100) is the maximum number of steps. The numbers are separated by a space.

**Output Specification:**

For each test case, output two numbers, one in each line. The first number is the paired palindromic number of N, and the second number is the number of steps taken to find the palindromic number. If the palindromic number is not found after K steps, just output the number obtained at the Kth step and K instead.

**Sample Input 1:**

67 3

**Sample Output 1:**

484

2

**Sample Input 2:**

69 3

**Sample Output 2:**

1353

3

# 1025. PAT Ranking (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Programming Ability Test (PAT) is organized by the College of Computer Science and Technology of Zhejiang University. Each test is supposed to run simultaneously in several places, and the ranklists will be merged immediately after the test. Now it is your job to write a program to correctly merge all the ranklists and generate the final rank.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive number N (<=100), the number of test locations. Then N ranklists follow, each starts with a line containing a positive integer K (<=300), the number of testees, and then K lines containing the registration number (a 13-digit number) and the total score of each testee. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, first print in one line the total number of testees. Then print the final ranklist in the following format:

registration\_number final\_rank location\_number local\_rank

The locations are numbered from 1 to N. The output must be sorted in nondecreasing order of the final ranks. The testees with the same score must have the same rank, and the output must be sorted in nondecreasing order of their registration numbers.

**Sample Input:**

2

5

1234567890001 95

1234567890005 100

1234567890003 95

1234567890002 77

1234567890004 85

4

1234567890013 65

1234567890011 25

1234567890014 100

1234567890012 85

**Sample Output:**

9

1234567890005 1 1 1

1234567890014 1 2 1

1234567890001 3 1 2

1234567890003 3 1 2

1234567890004 5 1 4

1234567890012 5 2 2

1234567890002 7 1 5

1234567890013 8 2 3

1234567890011 9 2 4

# 1026. Table Tennis (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A table tennis club has N tables available to the public. The tables are numbered from 1 to N. For any pair of players, if there are some tables open when they arrive, they will be assigned to the available table with the smallest number. If all the tables are occupied, they will have to wait in a queue. It is assumed that every pair of players can play for at most 2 hours.

Your job is to count for everyone in queue their waiting time, and for each table the number of players it has served for the day.

One thing that makes this procedure a bit complicated is that the club reserves some tables for their VIP members. When a VIP table is open, the first VIP pair in the queue will have the priviledge to take it. However, if there is no VIP in the queue, the next pair of players can take it. On the other hand, if when it is the turn of a VIP pair, yet no VIP table is available, they can be assigned as any ordinary players.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains an integer N (<=10000) - the total number of pairs of players. Then N lines follow, each contains 2 times and a VIP tag: HH:MM:SS - the arriving time, P - the playing time in minutes of a pair of players, and tag - which is 1 if they hold a VIP card, or 0 if not. It is guaranteed that the arriving time is between 08:00:00 and 21:00:00 while the club is open. It is assumed that no two customers arrives at the same time. Following the players' info, there are 2 positive integers: K (<=100) - the number of tables, and M (< K) - the number of VIP tables. The last line contains M table numbers.

**Output Specification:**

For each test case, first print the arriving time, serving time and the waiting time for each pair of players in the format shown by the sample. Then print in a line the number of players served by each table. Notice that the output must be listed in chronological order of the serving time. The waiting time must be rounded up to an integer minute(s). If one cannot get a table before the closing time, their information must NOT be printed.

**Sample Input:**

9

20:52:00 10 0

08:00:00 20 0

08:02:00 30 0

20:51:00 10 0

08:10:00 5 0

08:12:00 10 1

20:50:00 10 0

08:01:30 15 1

20:53:00 10 1

3 1

2

**Sample Output:**

08:00:00 08:00:00 0

08:01:30 08:01:30 0

08:02:00 08:02:00 0

08:12:00 08:16:30 5

08:10:00 08:20:00 10

20:50:00 20:50:00 0

20:51:00 20:51:00 0

20:52:00 20:52:00 0

3 3 2

来自：http://www.cnblogs.com/huhuuu/p/3360207.html

题意自己理解了，主要是两个队列维护，一个VIP队列，一个普通队列

搜集了一些坑（有些坑转自别的网站用于广大同学的测试之用）

普通人也有VIP的权益！！！ 屌丝逆袭有木有！！！

9  
20:52:00 10 0  
08:00:00 20 0  
08:02:00 30 0  
20:51:00 10 0  
08:10:00 5 0  
08:12:00 10 1  
20:50:00 10 0  
08:01:30 15 1  
20:53:00 10 1  
10 10  
1 2 3 4 5 6 7 8 9 10

08:00:00 08:00:00 0  
08:01:30 08:01:30 0  
08:02:00 08:02:00 0  
08:10:00 08:10:00 0  
08:12:00 08:12:00 0  
20:50:00 20:50:00 0  
20:51:00 20:51:00 0  
20:52:00 20:52:00 0  
20:53:00 20:53:00 0  
2 2 2 2 1 0 0 0 0 0

1.当有多个乒乓球台空闲时，vip顾客到了会使用最小id的vip球台，而不是最小id的球台，测试以下用例：

2

10:00:00 30 1

12:00:00 30 1

5 1

3

输出正确结果应为：

10:00:00 10:00:00 0

12:00:00 12:00:00 0

0 0 2 0 0

2.题目要求每对顾客玩的时间不超过2小时，那么当顾客要求玩的时间>2小时的时候，应该截断控制，测试以下用例：

2

18:00:00 180 1

20:00:00 60 1

1 1

1

输出的正确结果应为：

18:00:00 18:00:00 0

20:00:00 20:00:00 0

2

3.虽然题目中保证客户到达时间在08:00:00到21:00:00之间，但是根据最后的8个case来看，里面还是有不在这个时间区间内到达的顾客，所以建议还是稍加控制，测试以下用例：

1

21:00:00 80 1

1 1

1

输出的正确结果应为：

0

4.题目中说的round up to an integer minutes是严格的四舍五入，需要如下做：

wtime = (stime - atime + 30) / 60

而不是：

wtime = (stime - atime + 59) / 60

# 1027. Colors in Mars (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

People in Mars represent the colors in their computers in a similar way as the Earth people. That is, a color is represented by a 6-digit number, where the first 2 digits are for Red, the middle 2 digits for Green, and the last 2 digits for Blue. The only difference is that they use radix 13 (0-9 and A-C) instead of 16. Now given a color in three decimal numbers (each between 0 and 168), you are supposed to output their Mars RGB values.

**Input**

Each input file contains one test case which occupies a line containing the three decimal color values.

**Output**

For each test case you should output the Mars RGB value in the following format: first output "#", then followed by a 6-digit number where all the English characters must be upper-cased. If a single color is only 1-digit long, you must print a "0" to the left.

**Sample Input**

15 43 71

**Sample Output**

#123456

# 1028. List Sorting (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Excel can sort records according to any column. Now you are supposed to imitate this function.

**Input**

Each input file contains one test case. For each case, the first line contains two integers N (<=100000) and C, where N is the number of records and C is the column that you are supposed to sort the records with. Then N lines follow, each contains a record of a student. A student's record consists of his or her distinct ID (a 6-digit number), name (a string with no more than 8 characters without space), and grade (an integer between 0 and 100, inclusive).

**Output**

For each test case, output the sorting result in N lines. That is, if C = 1 then the records must be sorted in increasing order according to ID's; if C = 2 then the records must be sorted in non-decreasing order according to names; and if C = 3 then the records must be sorted in non-decreasing order according to grades. If there are several students who have the same name or grade, they must be sorted according to their ID's in increasing order.

**Sample Input 1**

3 1

000007 James 85

000010 Amy 90

000001 Zoe 60

**Sample Output 1**

000001 Zoe 60

000007 James 85

000010 Amy 90

**Sample Input 2**

4 2

000007 James 85

000010 Amy 90

000001 Zoe 60

000002 James 98

**Sample Output 2**

000010 Amy 90

000002 James 98

000007 James 85

000001 Zoe 60

**Sample Input 3**

4 3

000007 James 85

000010 Amy 90

000001 Zoe 60

000002 James 90

**Sample Output 3**

000001 Zoe 60

000007 James 85

000002 James 90

000010 Amy 90

# 1029. Median (25)

时间限制

1000 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given an increasing sequence S of N integers, the *median* is the number at the middle position. For example, the median of S1={11, 12, 13, 14} is 12, and the median of S2={9, 10, 15, 16, 17} is 15. The median of two sequences is defined to be the median of the nondecreasing sequence which contains all the elements of both sequences. For example, the median of S1 and S2 is 13.

Given two increasing sequences of integers, you are asked to find their median.

**Input**

Each input file contains one test case. Each case occupies 2 lines, each gives the information of a sequence. For each sequence, the first positive integer N (<=1000000) is the size of that sequence. Then N integers follow, separated by a space. It is guaranteed that all the integers are in the range of **long int**.

**Output**

For each test case you should output the median of the two given sequences in a line.

**Sample Input**

4 11 12 13 14

5 9 10 15 16 17

**Sample Output**

13

这题就用int类型就可以了

# 1030. Travel Plan (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A traveler's map gives the distances between cities along the highways, together with the cost of each highway. Now you are supposed to write a program to help a traveler to decide the shortest path between his/her starting city and the destination. If such a shortest path is not unique, you are supposed to output the one with the minimum cost, which is guaranteed to be unique.

**Input Specification:**

Each input file contains one test case. Each case starts with a line containing 4 positive integers N, M, S, and D, where N (<=500) is the number of cities (and hence the cities are numbered from 0 to N-1); M is the number of highways; S and D are the starting and the destination cities, respectively. Then M lines follow, each provides the information of a highway, in the format:

City1 City2 Distance Cost

where the numbers are all integers no more than 500, and are separated by a space.

**Output Specification:**

For each test case, print in one line the cities along the shortest path from the starting point to the destination, followed by the total distance and the total cost of the path. The numbers must be separated by a space and there must be no extra space at the end of output.

**Sample Input**

4 5 0 3

0 1 1 20

1 3 2 30

0 3 4 10

0 2 2 20

2 3 1 20

**Sample Output**

0 2 3 3 40

# 1031. Hello World for U (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given any string of N (>=5) characters, you are asked to form the characters into the shape of U. For example, "helloworld" can be printed as:

h d

e l

l r

lowo

That is, the characters must be printed in the original order, starting top-down from the left vertical line with n1characters, then left to right along the bottom line with n2 characters, and finally bottom-up along the vertical line with n3 characters. And more, we would like U to be as squared as possible -- that is, it must be satisfied that n1 = n3 = max { k| k <= n2 for all 3 <= n2 <= N } with n1 + n2 + n3 - 2 = N.

**Input Specification:**

Each input file contains one test case. Each case contains one string with no less than 5 and no more than 80 characters in a line. The string contains no white space.

**Output Specification:**

For each test case, print the input string in the shape of U as specified in the description.

**Sample Input:**

helloworld!

**Sample Output:**

h !

e d

l l

lowor

# 1032. Sharing (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

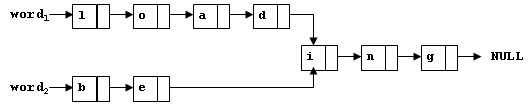
判题程序

Standard

作者

CHEN, Yue

To store English words, one method is to use linked lists and store a word letter by letter. To save some space, we may let the words share the same sublist if they share the same suffix. For example, "loading" and "being" are stored as showed in Figure 1.

  
Figure 1

You are supposed to find the starting position of the common suffix (e.g. the position of "i" in Figure 1).

**Input Specification:**

Each input file contains one test case. For each case, the first line contains two addresses of nodes and a positive N (<= 105), where the two addresses are the addresses of the first nodes of the two words, and N is the total number of nodes. The address of a node is a 5-digit positive integer, and NULL is represented by -1.

Then N lines follow, each describes a node in the format:

*Address Data Next*

where *Address* is the position of the node, *Data* is the letter contained by this node which is an English letter chosen from {a-z, A-Z}, and *Next* is the position of the next node.

**Output Specification:**

For each case, simply output the 5-digit starting position of the common suffix. If the two words have no common suffix, output "-1" instead.

**Sample Input 1:**

11111 22222 9

67890 i 00002

00010 a 12345

00003 g -1

12345 D 67890

00002 n 00003

22222 B 23456

11111 L 00001

23456 e 67890

00001 o 00010

**Sample Output 1:**

67890

**Sample Input 2:**

00001 00002 4

00001 a 10001

10001 s -1

00002 a 10002

10002 t -1

**Sample Output 2:**

-1

这题最坑的地方是有可能出现L2的开始地址就是-1，直接输出-1即可

# 1033. To Fill or Not to Fill (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

ZHANG, Guochuan

With highways available, driving a car from Hangzhou to any other city is easy. But since the tank capacity of a car is limited, we have to find gas stations on the way from time to time. Different gas station may give different price. You are asked to carefully design the cheapest route to go.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 4 positive numbers: Cmax (<= 100), the maximum capacity of the tank; D (<=30000), the distance between Hangzhou and the destination city; Davg (<=20), the average distance per unit gas that the car can run; and N (<= 500), the total number of gas stations. Then N lines follow, each contains a pair of non-negative numbers: Pi, the unit gas price, and Di (<=D), the distance between this station and Hangzhou, for i=1,...N. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print the cheapest price in a line, accurate up to 2 decimal places. It is assumed that the tank is empty at the beginning. If it is impossible to reach the destination, print "The maximum travel distance = X" where X is the maximum possible distance the car can run, accurate up to 2 decimal places.

**Sample Input 1:**

50 1300 12 8

6.00 1250

7.00 600

7.00 150

7.10 0

7.20 200

7.50 400

7.30 1000

6.85 300

**Sample Output 1:**

749.17

**Sample Input 2:**

50 1300 12 2

7.10 0

7.00 600

**Sample Output 2:**

The maximum travel distance = 1200.00

# 1034. Head of a Gang (30)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

One way that the police finds the head of a gang is to check people's phone calls. If there is a phone call between A and B, we say that A and B is related. The weight of a relation is defined to be the total time length of all the phone calls made between the two persons. A "Gang" is a cluster of more than 2 persons who are related to each other with total relation weight being greater than a given threshold K. In each gang, the one with maximum total weight is the head. Now given a list of phone calls, you are supposed to find the gangs and the heads.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains two positive numbers N and K (both less than or equal to 1000), the number of phone calls and the weight threthold, respectively. Then N lines follow, each in the following format:

Name1 Name2 Time

where Name1 and Name2 are the names of people at the two ends of the call, and Time is the length of the call. A name is a string of three capital letters chosen from A-Z. A time length is a positive integer which is no more than 1000 minutes.

**Output Specification:**

For each test case, first print in a line the total number of gangs. Then for each gang, print in a line the name of the head and the total number of the members. It is guaranteed that the head is unique for each gang. The output must be sorted according to the alphabetical order of the names of the heads.

**Sample Input 1:**

8 59

AAA BBB 10

BBB AAA 20

AAA CCC 40

DDD EEE 5

EEE DDD 70

FFF GGG 30

GGG HHH 20

HHH FFF 10

**Sample Output 1:**

2

AAA 3

GGG 3

**Sample Input 2:**

8 70

AAA BBB 10

BBB AAA 20

AAA CCC 40

DDD EEE 5

EEE DDD 70

FFF GGG 30

GGG HHH 20

HHH FFF 10

**Sample Output 2:**

0

# 1035. Password (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

To prepare for PAT, the judge sometimes has to generate random passwords for the users. The problem is that there are always some confusing passwords since it is hard to distinguish 1 (one) from l (L in lowercase), or 0 (zero) from O (o in uppercase). One solution is to replace 1 (one) by @, 0 (zero) by %, l by L, and O by o. Now it is your job to write a program to check the accounts generated by the judge, and to help the juge modify the confusing passwords.

**Input Specification:**

Each input file contains one test case. Each case contains a positive integer N (<= 1000), followed by N lines of accounts. Each account consists of a user name and a password, both are strings of no more than 10 characters with no space.

**Output Specification:**

For each test case, first print the number M of accounts that have been modified, then print in the following M lines the modified accounts info, that is, the user names and the corresponding modified passwords. The accounts must be printed in the same order as they are read in. If no account is modified, print in one line "There are N accounts and no account is modified" where N is the total number of accounts. However, if N is one, you must print "There is 1 account and no account is modified" instead.

**Sample Input 1:**

3

Team000002 Rlsp0dfa

Team000003 perfectpwd

Team000001 R1spOdfa

**Sample Output 1:**

2

Team000002 RLsp%dfa

Team000001 R@spodfa

**Sample Input 2:**

1

team110 abcdefg332

**Sample Output 2:**

There is 1 account and no account is modified

**Sample Input 3:**

2

team110 abcdefg222

team220 abcdefg333

**Sample Output 3:**

There are 2 accounts and no account is modified

注意输出的句子是有单复数变化的！

# 1036. Boys vs Girls (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

This time you are asked to tell the difference between the lowest grade of all the male students and the highest grade of all the female students.

**Input Specification:**

Each input file contains one test case. Each case contains a positive integer N, followed by N lines of student information. Each line contains a student's name, gender, ID and grade, separated by a space, where name and ID are strings of no more than 10 characters with no space, gender is either F (female) or M (male), and grade is an integer between 0 and 100. It is guaranteed that all the grades are distinct.

**Output Specification:**

For each test case, output in 3 lines. The first line gives the name and ID of the female student with the highest grade, and the second line gives that of the male student with the lowest grade. The third line gives the difference gradeF-gradeM. If one such kind of student is missing, output "Absent" in the corresponding line, and output "NA" in the third line instead.

**Sample Input 1:**

3

Joe M Math990112 89

Mike M CS991301 100

Mary F EE990830 95

**Sample Output 1:**

Mary EE990830

Joe Math990112

6

**Sample Input 2:**

1

Jean M AA980920 60

**Sample Output 2:**

Absent

Jean AA980920

NA

# 1037. Magic Coupon (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The magic shop in Mars is offering some magic coupons. Each coupon has an integer N printed on it, meaning that when you use this coupon with a product, you may get N times the value of that product back! What is more, the shop also offers some bonus product for free. However, if you apply a coupon with a positive N to this bonus product, you will have to pay the shop N times the value of the bonus product... but hey, magically, they have some coupons with negative N's!

For example, given a set of coupons {1 2 4 -1}, and a set of product values {7 6 -2 -3} (in Mars dollars M$) where a negative value corresponds to a bonus product. You can apply coupon 3 (with N being 4) to product 1 (with value M$7) to get M$28 back; coupon 2 to product 2 to get M$12 back; and coupon 4 to product 4 to get M$3 back. On the other hand, if you apply coupon 3 to product 4, you will have to pay M$12 to the shop.

Each coupon and each product may be selected at most once. Your task is to get as much money back as possible.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains the number of coupons NC, followed by a line with NC coupon integers. Then the next line contains the number of products NP, followed by a line with NP product values. Here 1<= NC, NP <= 105, and it is guaranteed that all the numbers will not exceed 230.

**Output Specification:**

For each test case, simply print in a line the maximum amount of money you can get back.

**Sample Input:**

4

1 2 4 -1

4

7 6 -2 -3

**Sample Output:**

43

# 1038. Recover the Smallest Number (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a collection of number segments, you are supposed to recover the smallest number from them. For example, given {32, 321, 3214, 0229, 87}, we can recover many numbers such like 32-321-3214-0229-87 or 0229-32-87-321-3214 with respect to different orders of combinations of these segments, and the smallest number is 0229-321-3214-32-87.

**Input Specification:**

Each input file contains one test case. Each case gives a positive integer N (<=10000) followed by N number segments. Each segment contains a non-negative integer of no more than 8 digits. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print the smallest number in one line. Do not output leading zeros.

**Sample Input:**

5 32 321 3214 0229 87

**Sample Output:**

22932132143287

# 1039. Course List for Student (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Zhejiang University has 40000 students and provides 2500 courses. Now given the student name lists of all the courses, you are supposed to output the registered course list for each student who comes for a query.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive integers: N (<=40000), the number of students who look for their course lists, and K (<=2500), the total number of courses. Then the student name lists are given for the courses (numbered from 1 to K) in the following format: for each course *i*, first the course index *i* and the number of registered students N*i* (<= 200) are given in a line. Then in the next line, N*i* student names are given. A student name consists of 3 capital English letters plus a one-digit number. Finally the last line contains the N names of students who come for a query. All the names and numbers in a line are separated by a space.

**Output Specification:**

For each test case, print your results in N lines. Each line corresponds to one student, in the following format: first print the student's name, then the total number of registered courses of that student, and finally the indices of the courses in increasing order. The query results must be printed in the same order as input. All the data in a line must be separated by a space, with no extra space at the end of the line.

**Sample Input:**

11 5

4 7

BOB5 DON2 FRA8 JAY9 KAT3 LOR6 ZOE1

1 4

ANN0 BOB5 JAY9 LOR6

2 7

ANN0 BOB5 FRA8 JAY9 JOE4 KAT3 LOR6

3 1

BOB5

5 9

AMY7 ANN0 BOB5 DON2 FRA8 JAY9 KAT3 LOR6 ZOE1

ZOE1 ANN0 BOB5 JOE4 JAY9 FRA8 DON2 AMY7 KAT3 LOR6 NON9

**Sample Output:**

ZOE1 2 4 5

ANN0 3 1 2 5

BOB5 5 1 2 3 4 5

JOE4 1 2

JAY9 4 1 2 4 5

FRA8 3 2 4 5

DON2 2 4 5

AMY7 1 5

KAT3 3 2 4 5

LOR6 4 1 2 4 5

NON9 0

# 1040. Longest Symmetric String (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a string, you are supposed to output the length of the longest symmetric sub-string. For example, given "Is PAT&TAP symmetric?", the longest symmetric sub-string is "s PAT&TAP s", hence you must output 11.

**Input Specification:**

Each input file contains one test case which gives a non-empty string of length no more than 1000.

**Output Specification:**

For each test case, simply print the maximum length in a line.

**Sample Input:**

Is PAT&TAP symmetric?

**Sample Output:**

11

# 1041. Be Unique (20)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Being unique is so important to people on Mars that even their lottery is designed in a unique way. The rule of winning is simple: one bets on a number chosen from [1, 104]. The first one who bets on a unique number wins. For example, if there are 7 people betting on 5 31 5 88 67 88 17, then the second one who bets on 31 wins.

**Input Specification:**

Each input file contains one test case. Each case contains a line which begins with a positive integer N (<=105) and then followed by N bets. The numbers are separated by a space.

**Output Specification:**

For each test case, print the winning number in a line. If there is no winner, print "None" instead.

**Sample Input 1:**

7 5 31 5 88 67 88 17

**Sample Output 1:**

31

**Sample Input 2:**

5 888 666 666 888 888

**Sample Output 2:**

None

# 1042. Shuffling Machine (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Shuffling is a procedure used to randomize a deck of playing cards. Because standard shuffling techniques are seen as weak, and in order to avoid "inside jobs" where employees collaborate with gamblers by performing inadequate shuffles, many casinos employ **automatic shuffling machines**. Your task is to simulate a shuffling machine.

The machine shuffles a deck of 54 cards according to a given random order and repeats for a given number of times. It is assumed that the initial status of a card deck is in the following order:

S1, S2, ..., S13, H1, H2, ..., H13, C1, C2, ..., C13, D1, D2, ..., D13, J1, J2

where "S" stands for "Spade", "H" for "Heart", "C" for "Club", "D" for "Diamond", and "J" for "Joker". A given order is a permutation of distinct integers in [1, 54]. If the number at the i-th position is j, it means to move the card from position i to position j. For example, suppose we only have 5 cards: S3, H5, C1, D13 and J2. Given a shuffling order {4, 2, 5, 3, 1}, the result will be: J2, H5, D13, S3, C1. If we are to repeat the shuffling again, the result will be: C1, H5, S3, J2, D13.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer K (<= 20) which is the number of repeat times. Then the next line contains the given order. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print the shuffling results in one line. All the cards are separated by a space, and there must be no extra space at the end of the line.

**Sample Input:**

2

36 52 37 38 3 39 40 53 54 41 11 12 13 42 43 44 2 4 23 24 25 26 27 6 7 8 48 49 50 51 9 10 14 15 16 5 17 18 19 1 20 21 22 28 29 30 31 32 33 34 35 45 46 47

**Sample Output:**

S7 C11 C10 C12 S1 H7 H8 H9 D8 D9 S11 S12 S13 D10 D11 D12 S3 S4 S6 S10 H1 H2 C13 D2 D3 D4 H6 H3 D13 J1 J2 C1 C2 C3 C4 D1 S5 H5 H11 H12 C6 C7 C8 C9 S2 S8 S9 H10 D5 D6 D7 H4 H13 C5

# 1043. Is It a Binary Search Tree (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A Binary Search Tree (BST) is recursively defined as a binary tree which has the following properties:

* The left subtree of a node contains only nodes with keys less than the node's key.
* The right subtree of a node contains only nodes with keys greater than or equal to the node's key.
* Both the left and right subtrees must also be binary search trees.

If we swap the left and right subtrees of every node, then the resulting tree is called the Mirror Image of a BST.

Now given a sequence of integer keys, you are supposed to tell if it is the preorder traversal sequence of a BST or the mirror image of a BST.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (<=1000). Then N integer keys are given in the next line. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, first print in a line "YES" if the sequence is the preorder traversal sequence of a BST or the mirror image of a BST, or "NO" if not. Then if the answer is "YES", print in the next line the postorder traversal sequence of that tree. All the numbers in a line must be separated by a space, and there must be no extra space at the end of the line.

**Sample Input 1:**

7

8 6 5 7 10 8 11

**Sample Output 1:**

YES

5 7 6 8 11 10 8

**Sample Input 2:**

7

8 10 11 8 6 7 5

**Sample Output 2:**

YES

11 8 10 7 5 6 8

**Sample Input 3:**

7

8 6 8 5 10 9 11

**Sample Output 3:**

NO

# 1044. Shopping in Mars (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Shopping in Mars is quite a different experience. The Mars people pay by chained diamonds. Each diamond has a value (in Mars dollars M$). When making the payment, the chain can be cut at any position for only once and some of the diamonds are taken off the chain one by one. Once a diamond is off the chain, it cannot be taken back. For example, if we have a chain of 8 diamonds with values M$3, 2, 1, 5, 4, 6, 8, 7, and we must pay M$15. We may have 3 options:

1. Cut the chain between 4 and 6, and take off the diamonds from the position 1 to 5 (with values 3+2+1+5+4=15).  
2. Cut before 5 or after 6, and take off the diamonds from the position 4 to 6 (with values 5+4+6=15).  
3. Cut before 8, and take off the diamonds from the position 7 to 8 (with values 8+7=15).

Now given the chain of diamond values and the amount that a customer has to pay, you are supposed to list all the paying options for the customer.

If it is impossible to pay the exact amount, you must suggest solutions with minimum lost.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 numbers: N (<=105), the total number of diamonds on the chain, and M (<=108), the amount that the customer has to pay. Then the next line contains N positive numbers D1 ... DN (Di<=103 for all i=1, ..., N) which are the values of the diamonds. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print "i-j" in a line for each pair of i <= j such that Di + ... + Dj = M. Note that if there are more than one solution, all the solutions must be printed in increasing order of i.

If there is no solution, output "i-j" for pairs of i <= j such that Di + ... + Dj > M with (Di + ... + Dj - M) minimized. Again all the solutions must be printed in increasing order of i.

It is guaranteed that the total value of diamonds is sufficient to pay the given amount.

**Sample Input 1:**

16 15

3 2 1 5 4 6 8 7 16 10 15 11 9 12 14 13

**Sample Output 1:**

1-5

4-6

7-8

11-11

**Sample Input 2:**

5 13

2 4 5 7 9

**Sample Output 2:**

2-4

4-5

# 1045. Favorite Color Stripe (30)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Eva is trying to make her own color stripe out of a given one. She would like to keep only her favorite colors in her favorite order by cutting off those unwanted pieces and sewing the remaining parts together to form her favorite color stripe.

It is said that a normal human eye can distinguish about less than 200 different colors, so Eva's favorite colors are limited. However the original stripe could be very long, and Eva would like to have the remaining favorite stripe with the maximum length. So she needs your help to find her the best result.

Note that the solution might not be unique, but you only have to tell her the maximum length. For example, given a stripe of colors {2 2 4 1 5 5 6 3 1 1 5 6}. If Eva's favorite colors are given in her favorite order as {2 3 1 5 6}, then she has 4 possible best solutions {2 2 1 1 1 5 6}, {2 2 1 5 5 5 6}, {2 2 1 5 5 6 6}, and {2 2 3 1 1 5 6}.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (<=200) which is the total number of colors involved (and hence the colors are numbered from 1 to N). Then the next line starts with a positive integer M (<=200) followed by M Eva's favorite color numbers given in her favorite order. Finally the third line starts with a positive integer L (<=10000) which is the length of the given stripe, followed by L colors on the stripe. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, simply print in a line the maximum length of Eva's favorite stripe.

**Sample Input:**

6

5 2 3 1 5 6

12 2 2 4 1 5 5 6 3 1 1 5 6

**Sample Output:**

7

# 1046. Shortest Distance (20)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The task is really simple: given N exits on a highway which forms a simple cycle, you are supposed to tell the shortest distance between any pair of exits.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains an integer N (in [3, 105]), followed by N integer distances D1 D2 ... DN, where Di is the distance between the i-th and the (i+1)-st exits, and DN is between the N-th and the 1st exits. All the numbers in a line are separated by a space. The second line gives a positive integer M (<=104), with M lines follow, each contains a pair of exit numbers, provided that the exits are numbered from 1 to N. It is guaranteed that the total round trip distance is no more than 107.

**Output Specification:**

For each test case, print your results in M lines, each contains the shortest distance between the corresponding given pair of exits.

**Sample Input:**

5 1 2 4 14 9

3

1 3

2 5

4 1

**Sample Output:**

3

10

7

# 1047. Student List for Course (25)

时间限制

400 ms

内存限制

64000 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Zhejiang University has 40000 students and provides 2500 courses. Now given the registered course list of each student, you are supposed to output the student name lists of all the courses.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 numbers: N (<=40000), the total number of students, and K (<=2500), the total number of courses. Then N lines follow, each contains a student's name (3 capital English letters plus a one-digit number), a positive number C (<=20) which is the number of courses that this student has registered, and then followed by C course numbers. For the sake of simplicity, the courses are numbered from 1 to K.

**Output Specification:**

For each test case, print the student name lists of all the courses in increasing order of the course numbers. For each course, first print in one line the course number and the number of registered students, separated by a space. Then output the students' names in alphabetical order. Each name occupies a line.

**Sample Input:**

10 5

ZOE1 2 4 5

ANN0 3 5 2 1

BOB5 5 3 4 2 1 5

JOE4 1 2

JAY9 4 1 2 5 4

FRA8 3 4 2 5

DON2 2 4 5

AMY7 1 5

KAT3 3 5 4 2

LOR6 4 2 4 1 5

**Sample Output:**

1 4

ANN0

BOB5

JAY9

LOR6

2 7

ANN0

BOB5

FRA8

JAY9

JOE4

KAT3

LOR6

3 1

BOB5

4 7

BOB5

DON2

FRA8

JAY9

KAT3

LOR6

ZOE1

5 9

AMY7

ANN0

BOB5

DON2

FRA8

JAY9

KAT3

LOR6

ZOE1

# 1048. Find Coins (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Eva loves to collect coins from all over the universe, including some other planets like Mars. One day she visited a universal shopping mall which could accept all kinds of coins as payments. However, there was a special requirement of the payment: for each bill, she could only use exactly two coins to pay the exact amount. Since she has as many as 105 coins with her, she definitely needs your help. You are supposed to tell her, for any given amount of money, whether or not she can find two coins to pay for it.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive numbers: N (<=105, the total number of coins) and M(<=103, the amount of money Eva has to pay). The second line contains N face values of the coins, which are all positive numbers no more than 500. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print in one line the two face values V1 and V2 (separated by a space) such that V1 + V2 = M and V1 <= V2. If such a solution is not unique, output the one with the smallest V1. If there is no solution, output "No Solution" instead.

**Sample Input 1:**

8 15

1 2 8 7 2 4 11 15

**Sample Output 1:**

4 11

**Sample Input 2:**

7 14

1 8 7 2 4 11 15

**Sample Output 2:**

No Solution

# 1049. Counting Ones (30)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The task is simple: given any positive integer N, you are supposed to count the total number of 1's in the decimal form of the integers from 1 to N. For example, given N being 12, there are five 1's in 1, 10, 11, and 12.

**Input Specification:**

Each input file contains one test case which gives the positive N (<=230).

**Output Specification:**

For each test case, print the number of 1's in one line.

**Sample Input:**

12

**Sample Output:**

5

# 1050. String Subtraction (20)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given two strings S1 and S2, S = S1 - S2 is defined to be the remaining string after taking all the characters in S2 from S1. Your task is simply to calculate S1 - S2 for any given strings. However, it might not be that simple to do it *fast*.

**Input Specification:**

Each input file contains one test case. Each case consists of two lines which gives S1 and S2, respectively. The string lengths of both strings are no more than 104. It is guaranteed that all the characters are visible ASCII codes and white space, and a new line character signals the end of a string.

**Output Specification:**

For each test case, print S1 - S2 in one line.

**Sample Input:**

They are students.

aeiou

**Sample Output:**

Thy r stdnts.

# 1051. Pop Sequence (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a stack which can keep M numbers at most. Push N numbers in the order of 1, 2, 3, ..., N and pop randomly. You are supposed to tell if a given sequence of numbers is a possible pop sequence of the stack. For example, if M is 5 and N is 7, we can obtain 1, 2, 3, 4, 5, 6, 7 from the stack, but not 3, 2, 1, 7, 5, 6, 4.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 3 numbers (all no more than 1000): M (the maximum capacity of the stack), N (the length of push sequence), and K (the number of pop sequences to be checked). Then K lines follow, each contains a pop sequence of N numbers. All the numbers in a line are separated by a space.

**Output Specification:**

For each pop sequence, print in one line "YES" if it is indeed a possible pop sequence of the stack, or "NO" if not.

**Sample Input:**

5 7 5

1 2 3 4 5 6 7

3 2 1 7 5 6 4

7 6 5 4 3 2 1

5 6 4 3 7 2 1

1 7 6 5 4 3 2

**Sample Output:**

YES

NO

NO

YES

NO

# 1052. Linked List Sorting (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A linked list consists of a series of structures, which are not necessarily adjacent in memory. We assume that each structure contains an integer key and a *Next* pointer to the next structure. Now given a linked list, you are supposed to sort the structures according to their key values in increasing order.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive N (< 105) and an address of the head node, where N is the total number of nodes in memory and the address of a node is a 5-digit positive integer. NULL is represented by -1.

Then N lines follow, each describes a node in the format:

Address Key Next

where Address is the address of the node in memory, Key is an integer in [-105, 105], and Next is the address of the next node. It is guaranteed that all the keys are distinct and there is no cycle in the linked list starting from the head node.

**Output Specification:**

For each test case, the output format is the same as that of the input, where N is the total number of nodes in the list and all the nodes must be sorted order.

**Sample Input:**

5 00001

11111 100 -1

00001 0 22222

33333 100000 11111

12345 -1 33333

22222 1000 12345

**Sample Output:**

5 12345

12345 -1 00001

00001 0 11111

11111 100 22222

22222 1000 33333

33333 100000 -1

# 1053. Path of Equal Weight (30)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

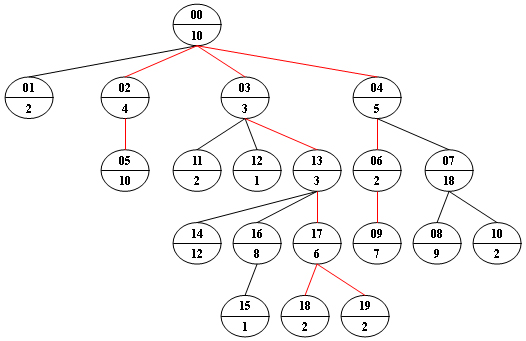
Standard

作者

CHEN, Yue

Given a non-empty tree with root R, and with weight Wi assigned to each tree node Ti. The **weight of a path from R to L** is defined to be the sum of the weights of all the nodes along the path from R to any leaf node L.

Now given any weighted tree, you are supposed to find all the paths with their weights equal to a given number. For example, let's consider the tree showed in Figure 1: for each node, the upper number is the node ID which is a two-digit number, and the lower number is the weight of that node. Suppose that the given number is 24, then there exists 4 different paths which have the same given weight: {10 5 2 7}, {10 4 10}, {10 3 3 6 2} and {10 3 3 6 2}, which correspond to the red edges in Figure 1.

  
Figure 1

**Input Specification:**

Each input file contains one test case. Each case starts with a line containing 0 < N <= 100, the number of nodes in a tree, M (< N), the number of non-leaf nodes, and 0 < S < 230, the given weight number. The next line contains N positive numbers where Wi (<1000) corresponds to the tree node Ti. Then M lines follow, each in the format:

ID K ID[1] ID[2] ... ID[K]

where ID is a two-digit number representing a given non-leaf node, K is the number of its children, followed by a sequence of two-digit ID's of its children. For the sake of simplicity, let us fix the root ID to be 00.

**Output Specification:**

For each test case, print all the paths with weight S in **non-increasing** order. Each path occupies a line with printed weights from the root to the leaf in order. All the numbers must be separated by a space with no extra space at the end of the line.

Note: sequence {A1, A2, ..., An} is said to be **greater than** sequence {B1, B2, ..., Bm} if there exists 1 <= k < min{n, m} such that Ai = Bifor i=1, ... k, and Ak+1 > Bk+1.

**Sample Input:**

20 9 24

10 2 4 3 5 10 2 18 9 7 2 2 1 3 12 1 8 6 2 2

00 4 01 02 03 04

02 1 05

04 2 06 07

03 3 11 12 13

06 1 09

07 2 08 10

16 1 15

13 3 14 16 17

17 2 18 19

**Sample Output:**

10 5 2 7

10 4 10

10 3 3 6 2

10 3 3 6 2

# 1054. The Dominant Color (20)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Behind the scenes in the computer's memory, color is always talked about as a series of 24 bits of information for each pixel. In an image, the color with the largest proportional area is called the dominant color. A *strictly* dominant color takes more than half of the total area. Now given an image of resolution M by N (for example, 800x600), you are supposed to point out the strictly dominant color.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive numbers: M (<=800) and N (<=600) which are the resolutions of the image. Then N lines follow, each contains M digital colors in the range [0, 224). It is guaranteed that the strictly dominant color exists for each input image. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, simply print the dominant color in a line.

**Sample Input:**

5 3

0 0 255 16777215 24

24 24 0 0 24

24 0 24 24 24

**Sample Output:**

24

# 1055. The World's Richest (25)

时间限制

400 ms

内存限制

128000 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Forbes magazine publishes every year its list of billionaires based on the annual ranking of the world's wealthiest people. Now you are supposed to simulate this job, but concentrate only on the people in a certain range of ages. That is, given the net worths of N people, you must find the M richest people in a given range of their ages.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive integers: N (<=105) - the total number of people, and K (<=103) - the number of queries. Then N lines follow, each contains the name (string of no more than 8 characters without space), age (integer in (0, 200]), and the net worth (integer in [-106, 106]) of a person. Finally there are K lines of queries, each contains three positive integers: M (<= 100) - the maximum number of outputs, and [Amin, Amax] which are the range of ages. All the numbers in a line are separated by a space.

**Output Specification:**

For each query, first print in a line "Case #X:" where X is the query number starting from 1. Then output the M richest people with their ages in the range [Amin, Amax]. Each person's information occupies a line, in the format

Name Age Net\_Worth

The outputs must be in non-increasing order of the net worths. In case there are equal worths, it must be in non-decreasing order of the ages. If both worths and ages are the same, then the output must be in non-decreasing alphabetical order of the names. It is guaranteed that there is no two persons share all the same of the three pieces of information. In case no one is found, output "None".

**Sample Input:**

12 4

Zoe\_Bill 35 2333

Bob\_Volk 24 5888

Anny\_Cin 95 999999

Williams 30 -22

Cindy 76 76000

Alice 18 88888

Joe\_Mike 32 3222

Michael 5 300000

Rosemary 40 5888

Dobby 24 5888

Billy 24 5888

Nobody 5 0

4 15 45

4 30 35

4 5 95

1 45 50

**Sample Output:**

Case #1:

Alice 18 88888

Billy 24 5888

Bob\_Volk 24 5888

Dobby 24 5888

Case #2:

Joe\_Mike 32 3222

Zoe\_Bill 35 2333

Williams 30 -22

Case #3:

Anny\_Cin 95 999999

Michael 5 300000

Alice 18 88888

Cindy 76 76000

Case #4:

None

# 1056. Mice and Rice (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

*Mice and Rice* is the name of a programming contest in which each programmer must write a piece of code to control the movements of a mouse in a given map. The goal of each mouse is to eat as much rice as possible in order to become a FatMouse.

First the playing order is randomly decided for NP programmers. Then every NG programmers are grouped in a match. The fattest mouse in a group wins and enters the next turn. All the losers in this turn are ranked the same. Every NGwinners are then grouped in the next match until a final winner is determined.

For the sake of simplicity, assume that the weight of each mouse is fixed once the programmer submits his/her code. Given the weights of all the mice and the initial playing order, you are supposed to output the ranks for the programmers.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive integers: NP and NG (<= 1000), the number of programmers and the maximum number of mice in a group, respectively. If there are less than NG mice at the end of the player's list, then all the mice left will be put into the last group. The second line contains NP distinct non-negative numbers Wi (i=0,...NP-1) where each Wiis the weight of the i-th mouse respectively. The third line gives the initial playing order which is a permutation of 0,...NP-1 (assume that the programmers are numbered from 0 to NP-1). All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print the final ranks in a line. The i-th number is the rank of the i-th programmer, and all the numbers must be separated by a space, with no extra space at the end of the line.

**Sample Input:**

11 3

25 18 0 46 37 3 19 22 57 56 10

6 0 8 7 10 5 9 1 4 2 3

**Sample Output:**

5 5 5 2 5 5 5 3 1 3 5

注意：

1、赶脚这道题目读懂题目比写出代码要难多了。。。

2、把题目中的例子分析一遍你就懂了：

1）input中的第三行是索引，表示的是以第三行索引的顺序进行比较，也就是说第6、0、8个mice先组成一组进行比较，而这3个mice吃到的rice就是第二行中对应的值19、25、57，因此第8个mice吃到的是57数量最多晋级，同理第7、10、5个mice吃到的rice是22、10、3，第7个晋级，第9、1、4个mice吃到的rice是56、18、37，第9个晋级，剩下的只有第2、3个mice，分别吃到的rice是0、46，第3个晋级，于是第8、7、9、3一共4个mice晋级到下一轮，并继续按照这个顺序比较，那么其他没有晋级的mice的排名都是第5名。

2）第8、7、9、3个mice吃到的rice分别是57、22、56、46，前三个中第8个晋级，剩下的第3个也晋级，所以未晋级的7、9的排名都是第3。

3）第8、3个mice的排名分别是第1和第2了。

3、这里有一点比较关键的就是未晋级的mice的排名名次的计算，知道了当前要比赛的mice的数量之后就可以计算出一共会有多少晋级，那么未晋级的mice的名次就都是本轮晋级数目+1了。

分析转自：<http://blog.csdn.net/yangsongtao1991/article/details/43417363>

# 1057. Stack (30)

时间限制

150 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Stack is one of the most fundamental data structures, which is based on the principle of Last In First Out (LIFO). The basic operations include Push (inserting an element onto the top position) and Pop (deleting the top element). Now you are supposed to implement a stack with an extra operation: PeekMedian -- return the median value of all the elements in the stack. With N elements, the median value is defined to be the (N/2)-th smallest element if N is even, or ((N+1)/2)-th if N is odd.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (<= 105). Then N lines follow, each contains a command in one of the following 3 formats:

Push *key*  
Pop  
PeekMedian

where *key* is a positive integer no more than 105.

**Output Specification:**

For each Push command, insert *key* into the stack and output nothing. For each Pop or PeekMedian command, print in a line the corresponding returned value. If the command is invalid, print "Invalid" instead.

**Sample Input:**

17

Pop

PeekMedian

Push 3

PeekMedian

Push 2

PeekMedian

Push 1

PeekMedian

Pop

Pop

Push 5

Push 4

PeekMedian

Pop

Pop

Pop

Pop

**Sample Output:**

Invalid

Invalid

3

2

2

1

2

4

4

5

3

Invalid

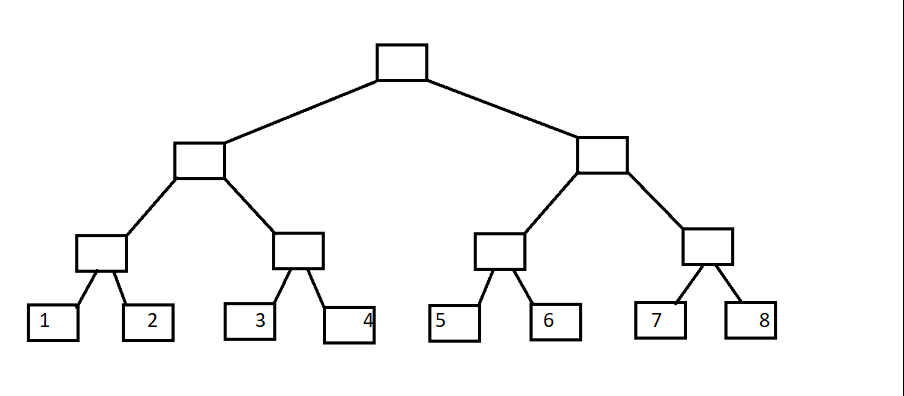
树状数组

思路转自：https://www.cnblogs.com/hsd-/p/6139376.html

重点是在**树状**的数组

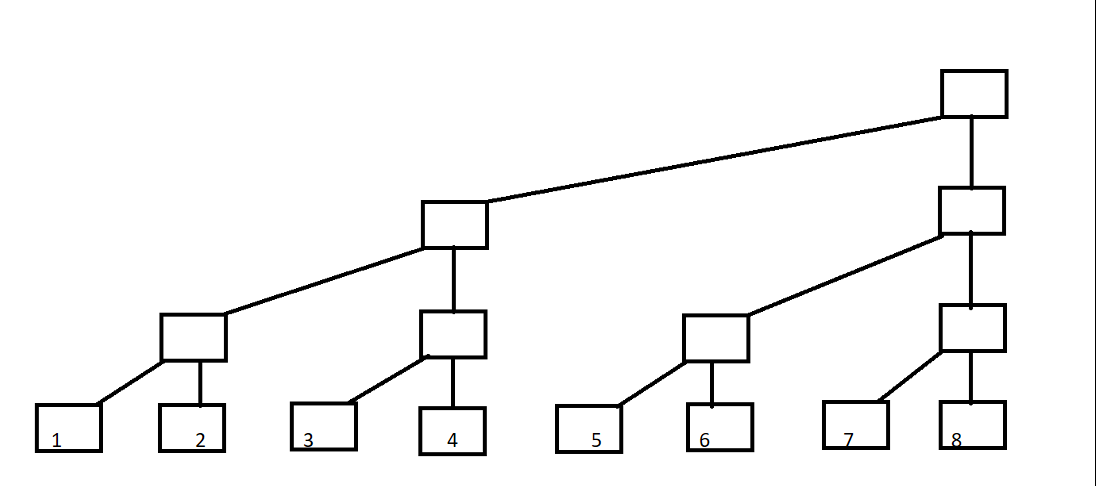
大家都知道二叉树吧

叶子结点代表A数组A[1]~A[8]

[](http://images2015.cnblogs.com/blog/786945/201612/786945-20161206222443741-1201716038.png)

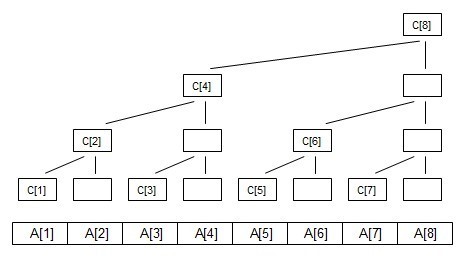
 .......

现在变形一下

[](http://images2015.cnblogs.com/blog/786945/201612/786945-20161206222444069-504520694.png)

 现在定义每一列的顶端结点C[]数组

 如下图



C[i]代表 子树的叶子结点的权值之和// **这里以求和举例**

如图可以知道

C[1]=A[1];

C[2]=A[1]+A[2];

C[3]=A[3];

C[4]=A[1]+A[2]+A[3]+A[4];

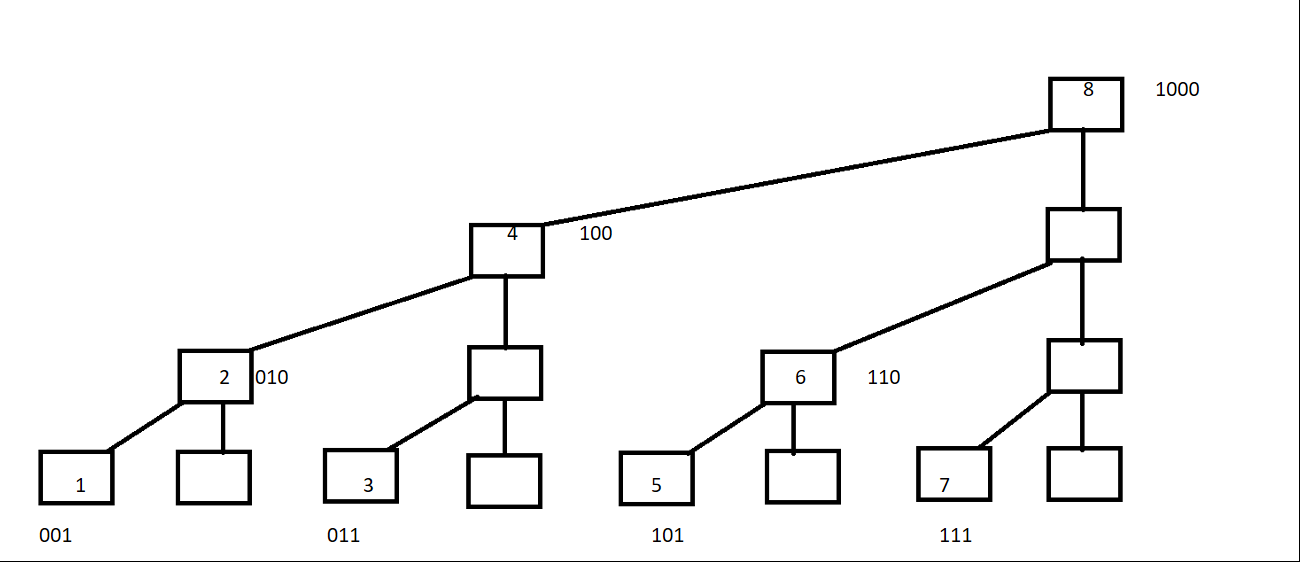
C[5]=A[5];

C[6]=A[5]+A[6];

C[7]=A[7];

C[8]=A[1]+A[2]+A[3]+A[4]+A[5]+A[6]+A[7]+A[8];

下面观察如下图

[](http://images2015.cnblogs.com/blog/786945/201612/786945-20161206222444679-518511660.png)

将C[]数组的结点序号转化为**二进制**

1=(001)      C[1]=A[1];

2=(010)      C[2]=A[1]+A[2];

3=(011)      C[3]=A[3];

4=(100)      C[4]=A[1]+A[2]+A[3]+A[4];

5=(101)      C[5]=A[5];

6=(110)      C[6]=A[5]+A[6];

7=(111)      C[7]=A[7];

8=(1000)    C[8]=A[1]+A[2]+A[3]+A[4]+A[5]+A[6]+A[7]+A[8];

对照式子可以发现  **C[i]=A[i-2^k+1]+A[i-2^k+2]+......A[i]; （k为i的二进制中从最低位到高位连续零的长度）例如i=8时，k=3;**

可以自行带入验证;

现在引入lowbit(x)

lowbit(x) 其实就是取出x的最低位1  换言之**lowbit(x)=2^k  k的含义与上面相同 理解一下**

下面说代码

1. int lowbit(int t)
2. {
3. return t&(-t);
4. }
5. //-t 代表t的负数 计算机中负数使用对应的正数的补码来表示
6. //例如 :
7. // t=6（0110） 此时 k=1
8. //-t=-6=(1001+1)=(1010)
9. // t&(-t)=(0010)=2=2^1

**C[i]=A[i-2^k+1]+A[i-2^k+2]+......A[i];**

**C[i]=A[i-lowbit(i)+1]+A[i-lowbit(i)+2]+......A[i];**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*分割线

**区间查询**

ok 下面利用C[i]数组，求A数组中前i项的和

举个例子 i=7;

sum[7]=A[1]+A[2]+A[3]+A[4]+A[5]+A[6]+A[7] ;   前i项和

C[4]=A[1]+A[2]+A[3]+A[4];   C[6]=A[5]+A[6];   C[7]=A[7];

可以推出:   sum[7]=C[4]+C[6]+C[7];

序号写为二进制: sum[(111)]=C[(100)]+C[(110)]+C[(111)];

再举个例子 i=5

sum[7]=A[1]+A[2]+A[3]+A[4]+A[5] ;   前i项和

C[4]=A[1]+A[2]+A[3]+A[4];   C[5]=A[5];

可以推出:   sum[5]=C[4]+C[5];

序号写为二进制: sum[(101)]=C[(100)]+C[(101)];

**细细观察二进制 树状数组追其根本就是二进制的应用**

结合代码

1. int getsum(int x)
2. {
3. int ans=0;
4. for(int i=x;i>0;i-=lowbit(i))
5. ans+=C[i];
6. return ans；
7. }

对于i=7 进行演示

                               7(111)     **ans+=C[7]**

lowbit(7)=001  7-lowbit(7)=6(110) **ans+=C[6]**

lowbit(6)=010  6-lowbit(6)=4(100)    **ans+=C[4]**

lowbit(4)=100  4-lowbit(4)=0(000)

对于i=5 进行演示

                               5(101)     **ans+=C[5]**

lowbit(5)=001  5-lowbit(5)=4(100) **ans+=C[4]**

lowbit(4)=100  4-lowbit(4)=0(000)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*分割线

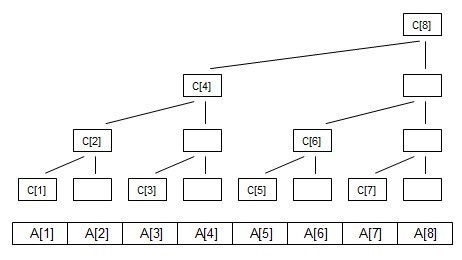
**单点更新**

当我们修改A[]数组中的某一个值时  应当如何更新C[]数组呢？

回想一下 区间查询的过程，再看一下上文中列出的图

结合代码分析

1. void add(int x,int y)
2. {
3. for(int i=x;i<=n;i+=lowbit(i))
4. tree[i]+=y;
5. }
6. //可以发现 更新过程是查询过程的逆过程
7. //由叶子结点向上更新C[]数组



如图：

当更新A[1]时  需要向上更新C[1] ,C[2],C[4],C[8]

                     C[1],   C[2],    C[4],     C[8]

写为二进制  C[(001)],C[(010)],C[(100)],C[(1000)]

                              1(001)        **C[1]+=A[1]**

lowbit(1)=001 1+lowbit(1)=2(010)     **C[2]+=A[1]**

lowbit(2)=010 2+lowbit(2)=4(100)     **C[4]+=A[1]**

lowbit(4)=100 4+lowbit(4)=8(1000)    **C[8]+=A[1]**

思路转自：http://blog.csdn.net/eli850934234/article/details/8863839

前几天在做[PAT1057](http://pat.zju.edu.cn/contests/pat-a-practise/1057)题时，老是有几个case超时，后来发现有人用树状数组来解此题，通过了所有的case，第一次听到这个名词，现记下：

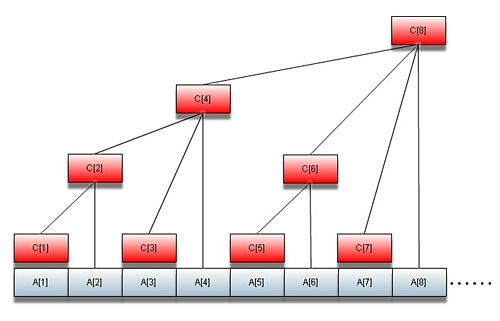
在解题过程中，我们有时需要维护一个数组的前缀和S[i]=A[1]+A[2]+...+A[i]。

但是不难发现，如果我们修改了任意一个A[i],S[i]、S[i+1]...S[n]都会发生变化。

可以说，每次修改A[i]后，调整前缀和S[]在最坏情况下会需要O(n)的时间。

当n非常大时，程序会运行得非常缓慢。

因此，这里我们引入“树状数组”，它的修改与求和都是O(logn)的，效率非常高。



令这棵树的结点编号为C1，C2...Cn。令每个结点的值为这棵树的值的总和，那么容易发现：C1 = A1C2 = A1 + A2C3 = A3C4 = A1 + A2 + A3 + A4C5 = A5C6 = A5 + A6C7 = A7C8 = A1 + A2 + A3 + A4 + A5 + A6 + A7 + A8...C16 = A1 + A2 + A3 + A4 + A5 + A6 + A7 + A8 + A9 + A10 + A11 + A12 + A13 + A14 + A15 + A16

这里有一个有趣的性质：设节点编号为x，那么这个节点管辖的区间为2^k（其中k为x二进制末尾0的个数）个元素。因为这个区间最后一个元素必然为Ax，所以很明显：Cn = A(n – 2^k + 1) + ... + An

算这个2^k有一个快捷的办法，定义一个函数如下即可：

**[cpp]** [view plain](http://blog.csdn.net/eli850934234/article/details/8863839) [copy](http://blog.csdn.net/eli850934234/article/details/8863839)

1. **int** lowbit(**int** x){
2. **return** x&(x^(x–1));
3. }

利用机器补码特性，也可以写成：

**[cpp]** [view plain](http://blog.csdn.net/eli850934234/article/details/8863839) [copy](http://blog.csdn.net/eli850934234/article/details/8863839)

1. **int** lowbit(**int** x){
2. **return** x&(-x);
3. }

例如对于节点5，补码为000...0101，-5的原码为100...0101，补码为011...1011，机器计算的时候根据补码来计算，可知 5&(-5)=000...0001，那么5节点管辖的区间为2^0，即一个节点”5“。

下面给出树状数组应用中的几个关键函数：

**[cpp]** [view plain](http://blog.csdn.net/eli850934234/article/details/8863839) [copy](http://blog.csdn.net/eli850934234/article/details/8863839)

1. **int** lowbit(**int** x)
2. {
3. **return**  x&(-x);
4. }

**[cpp]** [view plain](http://blog.csdn.net/eli850934234/article/details/8863839) [copy](http://blog.csdn.net/eli850934234/article/details/8863839)

1. **int** sum(**int** pos)
2. {
3. **int** res= 0;
4. **while**(pos > 0)
5. {
6. res += c[pos];
7. pos -= lowbit(pos);
8. }
9. **return** res;
10. }

**[cpp]** [view plain](http://blog.csdn.net/eli850934234/article/details/8863839) [copy](http://blog.csdn.net/eli850934234/article/details/8863839)

1. **void** add(**int** pos , **int** value)
2. {
3. **while**(pos <= n)
4. {
5. c[pos] += value;
6. pos += lowbit(pos);
7. }
8. }

五种解法，转自：http://blog.csdn.net/sinat\_29278271/article/details/47291659

## 解题思路

     题目的大意是维护一个能随时返回中位数的栈，这个问题其实可以直接简化为维护一个能返回中位数同时支持插入和删除的数据结构。本身解法比较多样，网上能找到的解法有三种，个人感觉都很有启发性，所以加上我的解法在此做个总结。

     因为题目中说明每个数据都在[1,100000]之间，所以很朴素的一种解法就是设立一个Count[100005]。插入n的时候Count[n]++，删除的时候Count[n]--，查找的时候遍历数组，寻找前缀和为（size+1)/2的下标。但是10^5本身是个比较大的数字，在多次查找之后果断超时了，怎么办呢？当然要做优化。

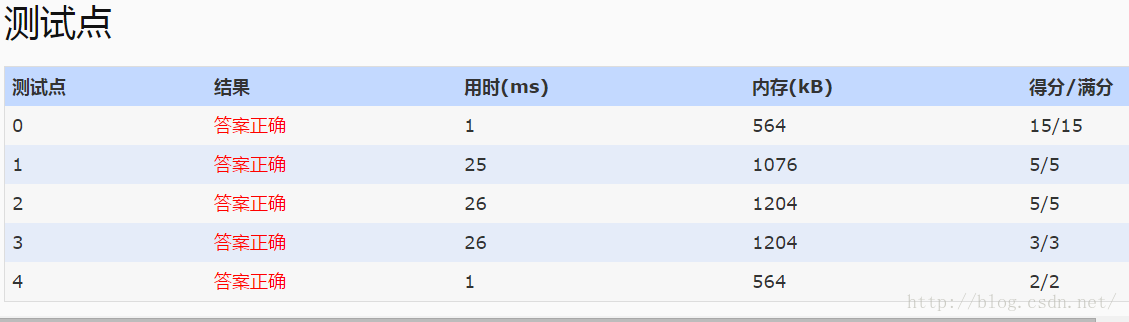
## 1.树状数组+二分查找

    树状数组(Binary Indexed Tree(BIT))是一种能高效查找前缀和的数据结构，。使用树状数组是为了能进行二分查找，原先遍历Count数组，最多的时候能遍历10^5次，运用二分查找可以将查找次数优化为lg(10^5)/lg(2)  < 15

下面是代码

**[cpp]** [view plain](http://blog.csdn.net/sinat_29278271/article/details/47291659) [copy](http://blog.csdn.net/sinat_29278271/article/details/47291659)

1. # include <cstdio>
2. # include <stack>
3. **using** **namespace** std;
4. **class** BIT
5. {
6. **private**:
7. **int** \*Elem;
8. **int** Size;
9. **int** lowbit(**int** n)
10. {
11. **return** n&(-n);
12. }
13. **public**:
14. BIT(**int** size):Size(size+1)  /\*想想看还是+1好了，要不申请了100的空间只能用到99感觉太奇怪了\*/
15. {
16. Elem = **new** **int**[Size];
17. **for** (**int** i=0;i<Size;i++)/\*还没试过用memset初始化，下次试试\*/
18. Elem[i] = 0;
19. }
20. **int** GetSum(**int** right)/\*[0,right]\*/
21. {
22. **int** sum = 0;
23. **while** (right)
24. {
25. sum += Elem[right];
26. right -= lowbit(right);
27. }
28. **return** sum;
29. }
30. **int** GetSum(**int** left,**int** right)/\*[left,right]\*/
31. {
32. **return** GetSum(left-1) - GetSum(right);
33. }
34. **void** Add(**int** value,**int** index)
35. {
36. **while** (index < Size)
37. {
38. Elem[index] += value;
39. index += lowbit(index);
40. }
41. }
42. ~BIT()
43. {
44. **delete**[] Elem;
45. }
46. };
47. BIT bit(100000);
48. **int** getmid(**int** size)
49. {
50. **int** index = (size+1)/2;
51. **int** left = 1,right = 100000,mid;
52. **while**(left<right)
53. {
54. mid = (left+right)/2;
55. **if**(bit.GetSum(mid)<index)
56. left = mid+1;
57. **else**
58. right = mid;
59. }
60. **return** left;
61. }
62. **int** main()
63. {
64. **int** n,tmp;
65. scanf("%d",&n);
66. stack<**int**> s;
67. **char** str[10];
68. **while** (n--)
69. {
70. scanf("%s",str);
71. **switch**(str[1])
72. {
73. **case** 'e':
74. {
75. **if** (s.empty())
76. printf("Invalid\n");
77. **else**
78. printf("%d\n",getmid(s.size()));
79. **break**;
80. }
81. **case** 'o':
82. {
83. **if** (s.empty())
84. printf("Invalid\n");
85. **else**
86. {
87. tmp = s.top();s.pop();
88. printf("%d\n",tmp);
89. bit.Add(-1,tmp);
90. }
91. **break**;
92. }
93. **case** 'u':
94. {
95. scanf("%d",&tmp);s.push(tmp);
96. bit.Add(1,tmp);
97. }
98. **break**;
99. }
100. }
101. **return** 0;
102. }</span>



## 2.分桶法（分治，分层HASH，平方分割）本人快乐的原创

     分桶法的基本思路是分治，在一开始的暴力解法中，我们可以认为Count数组是一个大的桶，这个大的桶里有5\*10^5个小桶，每个小桶能装一个数，在分桶法中，我们建立多个大桶，每个桶中又有小桶，比如，我们建立多个500个大桶，每个桶的容量是100，同时记录每个大桶中存放的数据的个数，在查找的时候我们可以通过每个大桶中元素的快速定位到放置中位数的那个小桶。当然你可以认为这是一种HASH，hash(key) = key/10。设每个大桶中含有k个小桶，共有m个大桶，m\*k = n为定值。则一开始我们需要遍历大小为m的大桶数组，后来要遍历大小为k的单个大桶，时间复杂度为O（max(k,m))在n\*k为定值的情况下，易知m = k = （m\*k）^(1/2)的时候效率最高为n^(1/2)。

     本题中为了方便，采用分层hash的策略，将值为key的元素放入bucke[k/100][k%100]中。

**[cpp]** [view plain](http://blog.csdn.net/sinat_29278271/article/details/47291659) [copy](http://blog.csdn.net/sinat_29278271/article/details/47291659)

1. # include <cstdio>
2. # include <stack>
3. **using** **namespace** std;
5. **const** **int** \_size = 100000;
6. **const** **int** capi  = 500;
7. **int** bucket[\_size/capi][capi];
8. **int** count[\_size/capi];
9. **int** getmid(**int** size)
10. {
11. **int** ind = (size+1)/2,cnt=0,i,j;
12. **for** (i=0;i<\_size/capi;i++)
13. {
14. **if** (cnt + count[i]>=ind)
15. **break**;
16. cnt += count[i];
17. }
18. **for** (j=0;j<capi;j++)
19. {
20. cnt += bucket[i][j];
21. **if** (cnt>=ind)
22. **return** j+i\*capi;
23. }
25. }
26. **char** str[10];
27. **int** main()
28. {
29. **int** n,tmp;
30. scanf("%d",&n);
31. stack<**int**> s;
32. **while** (n--)
33. {
34. scanf("%s",str);
35. **switch**(str[1])
36. {
37. **case** 'e':
38. {
39. **if** (s.empty())
40. printf("Invalid\n");
41. **else**
42. printf("%d\n",getmid(s.size())+1);
43. **break**;
44. }
45. **case** 'o':
46. {
47. **if** (s.empty())
48. printf("Invalid\n");
49. **else**
50. {
51. tmp = s.top();s.pop();
52. printf("%d\n",tmp);
53. tmp--;
54. bucket[tmp/capi][tmp%capi]--;
55. count[tmp/capi]--;
56. }
57. **break**;
58. }
59. **case** 'u':
60. {
61. scanf("%d",&tmp);s.push(tmp);
62. tmp--;
63. bucket[tmp/capi][tmp%capi]++;
64. count[tmp/capi]++;
65. }
66. **break**;
67. }
68. }
69. **return** 0;
70. }

最后我想说的就是，

1.这个方法和树状数组+二分的方法并无矛盾，你同样可以用树状数组优化大桶元素的前缀和。

2.还有就是如果你乐意你完全可以多分几个层玩,比如key放在bucket[...][...][...],分层分多了以后，你会发现这个桶变成了一棵树，如果你分层的依据是二分法，你还会发现，你分出了一棵线段树。

3.如果数据范围增大，你可以修改hash使其映射到更小的空间，同时将每个大桶改为vector<int>数组，查询是对每个vector<int>中的元素排序，个人感觉不会很慢

## 3.线段树（分治）有种杀鸡用牛刀的感觉

           线段树是个霸气的数据结构，基本上包含了分桶法和树状数组的全部功能。线段树的基础思想是分治，但是时间复杂度上比分桶法更加高效，能将时间优化到O（lgn）然而在PAT的小数据之下，普通的线段树因为常数上的差距花费的时间更长。具体的树的创建我就不说了，这里采用一点zkw线段树的思想，直接找到树的叶子自底向上走到树根，每个节点维护一个Cnt记录经过这里的路径的个数，查找中位数的时候根据每个节点的Cnt进入合适的子树进行查找。

**[cpp]** [view plain](http://blog.csdn.net/sinat_29278271/article/details/47291659) [copy](http://blog.csdn.net/sinat_29278271/article/details/47291659)

1. # include <cstdio>
2. # include <stack>
3. **using** **namespace** std;
5. **typedef** **int** Node;
6. **class** zkw\_segtree
7. {
8. **private**:
9. Node \*T;
10. **int** size;
11. **public**:
12. zkw\_segtree(**int** range)
13. {
14. **for** (size = 1;size < range+2;size<<=1);
15. T = **new** Node[2\*size];
16. **for** (**int** i=1;i<size+size;i++)
17. T[i] = 0;
18. }
19. **void** Add(**int** value,**int** i)
20. {
21. **for** (i+=size;i;i>>=1)
22. T[i] += value;
23. }
24. **int** Query(**int** s,**int** t)
25. {
26. **int** ret = 0;
27. **for** (s+=size-1,t+=size+1;s^t^1;s>>=1,t>>=1)
28. {
29. **if** (~s^1) ret += T[s^1];
30. **if** (t^1)  ret += T[t^1];
31. }
32. **return** ret;
33. }
34. **int** Find\_Kth(**int** k,**int** root = 1)
35. {
36. **while** (root<<1 < size<<1)
37. {
38. **if** (T[root<<1]>=k)  root = root<<1;
39. **else**
40. {
41. k -= T[root<<1];
42. root = (root<<1) + 1;
43. }
44. }
45. **return** root - size;
46. }
47. ~zkw\_segtree()
48. {
49. **delete**[] T;
50. }
51. };
52. zkw\_segtree segtree(100000);
53. **int** main()
54. {
55. **int** n,tmp;
56. scanf("%d",&n);
57. stack<**int**> s;
58. **char** str[10];
59. **while** (n--)
60. {
61. scanf("%s",str);
62. **switch**(str[1])
63. {
64. **case** 'e':
65. {
66. **if** (s.empty())
67. printf("Invalid\n");
68. **else**
69. printf("%d\n",segtree.Find\_Kth((s.size()+1)/2));
70. **break**;
71. }
72. **case** 'o':
73. {
74. **if** (s.empty())
75. printf("Invalid\n");
76. **else**
77. {
78. tmp = s.top();s.pop();
79. printf("%d\n",tmp);
80. segtree.Add(-1,tmp);
81. }
82. **break**;
83. }
84. **case** 'u':
85. {
86. scanf("%d",&tmp);s.push(tmp);
87. segtree.Add(1,tmp);
88. }
89. **break**;
90. }
91. }
92. **return** 0;
93. }

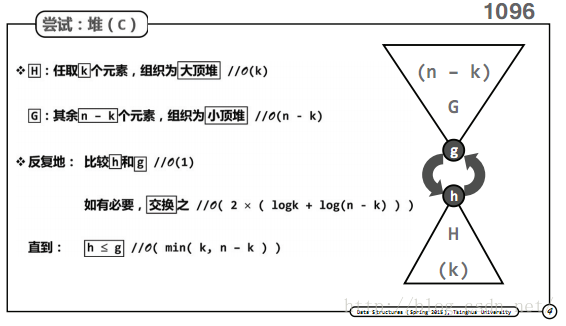


      测试完发现是最快的，不愧是zkw大神的杰作

## 4.Prioriry Queue On Multiset(红黑树是支持插入与删除的堆)真正的牛刀

        讲了那么多终于讲到最终的解法了，前三种方法归根结底是利用了输入数据范围有限这一条件，要想完美解决这一问题，我们不得不借助插入查找删除效率都为O（lgn）的高级搜索树。

      有一种利用堆查找第K个元素的算法，维持一个大顶堆，一个小顶堆，将K个元素压入小顶堆，其余压入大顶堆，随后如果大顶堆的堆顶元素大于小顶堆的堆顶元素，就将将两个堆得元素弹出并压入另一个堆中，直到大顶堆的堆顶元素小于小顶堆的堆顶元素。这样小顶堆的堆顶元素就是第K个元素。



       然而对于这道题，这种算法并没有什么用处，这道题要求随时删除某一个元素，然而堆并不具备这种功能（）。。

       犹记得当初学数据结构的时候，是先教的高级搜索树，后教的优先队列。老师在教优先队列的时候，为了引出堆，说，其实想要实现优先队列，用高级搜索树是完全可以的，但高级搜索树实在是太强大了，实现也比较复杂，我们需要一种更简单高效的数据结构去实现优先队列。

    于是传说中的堆诞生了，因为堆在结构上的特点，我们也无法做到随意删除其中的某个元素。

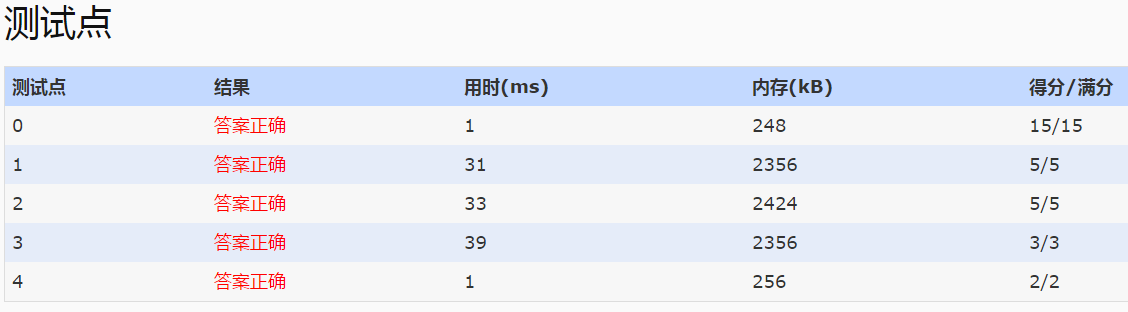
    然而此时我们需要一个能删除任意元素的优先队列，OK，。以上文字过于中二，请谨慎阅读。

    从时间复杂度上分析，插入查找和删除都是O（lgn）这里的n是指总元素的个数。

    然后，估计大家都知道要怎么做了，下面是代码，为了简单起见，并没有扩展出寻找第K个数的功能，而是直接寻找中位数。  
以下是代码

**[cpp]** [view plain](http://blog.csdn.net/sinat_29278271/article/details/47291659) [copy](http://blog.csdn.net/sinat_29278271/article/details/47291659)

1. # include <cstdio>
2. # include <stack>
3. # include <set>
4. **using** **namespace** std;
6. **const** **int** debug = 1;
7. **typedef** **int** T;
8. **class** Find\_Median
9. {
10. **private**:
11. multiset<T,greater<T> > maxheap;
12. multiset<T,less<T> > minheap;
13. **public**:
14. **void** Push(T data)
15. {
16. **if** (maxheap.size() < minheap.size())
17. maxheap.insert(data);
18. **else**
19. minheap.insert(data);
20. }
21. **bool** Erase(T data)
22. {
23. multiset<T>::iterator it;
24. **if** ((it=maxheap.find(data))!=maxheap.end())
25. maxheap.erase(it);
26. **else** **if** ((it=minheap.find(data))!=minheap.end())
27. minheap.erase(it);
28. **else**
29. **return** **false**;
30. **return** **true**;
31. }
32. T Query()
33. {
34. **while** (maxheap.size() < minheap.size())
35. {
36. maxheap.insert(\*minheap.begin());
37. minheap.erase(minheap.begin());
38. }
39. **while** (minheap.size() < maxheap.size())
40. {
41. minheap.insert(\*maxheap.begin());
42. maxheap.erase(maxheap.begin());
43. }
45. **if** (maxheap.size()==0) **return** \*minheap.begin();
46. **if** (minheap.size()==0) **return** \*maxheap.begin();
48. multiset<T>::iterator maxtop = maxheap.begin();
49. multiset<T>::iterator mintop = minheap.begin();
50. **while** (\*maxtop > \*mintop)
51. {
52. maxheap.insert(\*mintop);
53. minheap.insert(\*maxtop);
54. maxheap.erase(maxtop);
55. minheap.erase(mintop);
56. maxtop = maxheap.begin();
57. mintop = minheap.begin();
58. }
59. **return** \*(maxheap.size() >= minheap.size()?maxtop:mintop);
60. }
61. };
62. Find\_Median FM;
63. **int** main()
64. {
65. **int** n,tmp;
66. scanf("%d",&n);
67. stack<**int**> s;
68. **char** str[10];
69. **while** (n--)
70. {
71. scanf("%s",str);
72. **switch**(str[1])
73. {
74. **case** 'e':
75. {
76. **if** (s.empty())  printf("Invalid\n");
77. **else**
78. printf("%d\n",FM.Query());
79. **break**;
80. }
81. **case** 'o':
82. {
83. **if** (s.empty())  printf("Invalid\n");
84. **else**
85. {
86. tmp = s.top();s.pop();
87. printf("%d\n",tmp);
88. FM.Erase(tmp);
89. }
90. **break**;
91. }
92. **case** 'u':
93. {
94. scanf("%d",&tmp);s.push(tmp);
95. FM.Push(tmp);
96. }
97. **break**;
98. }
99. }
100. **return** 0;
101. }



看起来是最慢的，一半怪stl太慢，一半怪我的实现太渣

## 5. 平衡搜索树——Treap（我觉得从前维护两个set求中间值是牛刀，那时我还小）

        很久没有更新这篇文章了，这段时间我考完了PAT，虽然没有拿到满分，但是这个成绩我也比较满意了，后来发生一些乱七八糟的事，现在也总算混进了集训队，也学到了一些东西吧。

        Treap就是其中一个东西，不过主标题是平衡搜索树，就是说这道题可以使用平衡搜索树来实现，而treap只是其中一种方法。

        我们先考虑不是平衡搜索树，而是一棵普通搜索树的情况，对于每棵树我们记录这个树中总共有几个节点。

        现在记lsize是左子树的规模，rsize是右子树的规模。

        当我们需要查找第K小的数时

                如果K<=lsize，说明我们要到左子树中去查找第K大的数。

                如果K>lsize+1，说明我们要到右子树中查找第第K-(lsize+1)的数

                如果前两项不符合，说明当前节点就是我们需要寻找的第K小值，直接返回结果

        然后解释为什么要使用平衡树，不用平衡树查找时间会很长，虽然我没有试过会不会超时。

        最后安利一下Treap这个数据结构，毫无疑问是最好写的平衡二叉树，没有之一。我至今不会写红黑书，AVL树写了两次，差点把我写哭出来。但是Treap能在需要的时候随手敲出一个来。

以下是代码，虽然这段代码看起来很长，但是前头那个名片可以删除，头文件可以删除一部分，正文部分比较长是因为我出于效率考虑在单个节点中添加cnt记录重复节点的个数，这让代码变得很长，不这样做应该也是可以过的。

PS：习惯使用cin，cout的同学请小心，这道题目如果不取消cin，cout和标准输入输出流的同步，输入输出起码能用掉80ms，不要问我是怎么知道的。

**[cpp]** [view plain](http://blog.csdn.net/sinat_29278271/article/details/47291659) [copy](http://blog.csdn.net/sinat_29278271/article/details/47291659)

1. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
2. > File Name: tmp.cpp
3. > Author: Uncle\_Sugar
4. > Mail: uncle\_sugar@qq.com
5. > Created Time: 2016年02月29日 星期一 13时28分28秒
6. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
7. # include <cstdio>
8. # include <cstring>
9. # include <cmath>
10. # include <cstdlib>
11. # include <climits>
12. # include <iostream>
13. # include <iomanip>
14. # include <set>
15. # include <map>
16. # include <vector>
17. # include <stack>
18. # include <queue>
19. # include <algorithm>
20. **using** **namespace** std;
22. **const** **int** debug = 1;
23. **const** **int** size  = 5000 + 10;
24. **typedef** **long** **long** ll;
26. **struct** Treap\_Node{
27. **int** value;
28. **int** fix,cnt,size;
29. Treap\_Node \*left,\*right;
30. Treap\_Node():cnt(0),size(0),left(NULL),right(NULL){}
31. Treap\_Node(**int** \_value):value(\_value),cnt(0),size(0),left(NULL),right(NULL){}
32. }\*root = NULL;
33. **inline** **void** Treap\_SetSize(Treap\_Node \*&P){
34. **if** (P){
35. P->size = P->cnt;
36. **if** (P->left) P->size += P->left->size;
37. **if** (P->right)    P->size += P->right->size;
38. }
39. }
40. **inline** **int** lsize(Treap\_Node \*&P){
41. **return** P->left?P->left->size:0;
42. }
43. **inline** **int** rsize(Treap\_Node \*&P){
44. **return** P->right?P->right->size:0;
45. }
46. **void** Treap\_Left\_Rotate(Treap\_Node \*&a){
47. Treap\_Node \*b = a->right;
48. a->right = b->left;
49. b->left = a;
50. a = b;
51. Treap\_SetSize(a->left);
52. Treap\_SetSize(a->right);
53. Treap\_SetSize(a);
54. }
55. **void** Treap\_Right\_Rotate(Treap\_Node \*&a){
56. Treap\_Node \*b = a->left;
57. a->left = b->right;
58. b->right = a;
59. a = b;
60. Treap\_SetSize(a->left);
61. Treap\_SetSize(a->right);
62. Treap\_SetSize(a);
63. }
64. **void** Treap\_Insert(Treap\_Node \*&P,**int** value){
65. **if** (!P){
66. P = **new** Treap\_Node;
67. P->value = value;
68. P->fix = rand();
69. }
70. **if** (value < P->value){
71. Treap\_Insert(P->left,value);
72. **if** (P->left->fix < P->fix)
73. Treap\_Right\_Rotate(P);
74. }
75. **else** **if** (P->value < value){
76. Treap\_Insert(P->right,value);
77. **if** (P->right->fix < P->fix)
78. Treap\_Left\_Rotate(P);
79. }
80. **else** {
81. P->cnt++;
82. }
83. Treap\_SetSize(P);
84. }
85. **bool** Treap\_Delete(Treap\_Node \*&P,**int** value){
86. **bool** ret = **false**;
87. **if** (!P) {
88. ret = **false**;
89. }
90. **else** {
91. **if** (value < P->value)
92. Treap\_Delete(P->left,value);
93. **else** **if** (P->value < value)
94. Treap\_Delete(P->right,value);
95. **else** {
96. **if** (P->cnt==0||(--P->cnt)==0){
97. **if** (!P->left||!P->right){
98. Treap\_Node \*t = P;
99. **if** (!P->right)
100. P = P->left;
101. **else**
102. P = P->right;
103. **delete** t;
104. ret = **true**;
105. }
106. **else** **if** (P->left->fix < P->right->fix){
107. Treap\_Right\_Rotate(P);
108. ret = Treap\_Delete(P->right,value);
109. }
110. **else** {
111. Treap\_Left\_Rotate(P);
112. ret = Treap\_Delete(P->left,value);
113. }
114. }
115. }
116. Treap\_SetSize(P);
117. }
118. **return** ret;
119. }
120. Treap\_Node\* Treap\_Findkth(Treap\_Node \*&P,**int** k){
121. **if** (k <= lsize(P))
122. **return** Treap\_Findkth(P->left,k);
123. **else** **if** (k > lsize(P)+P->cnt)
124. **return** Treap\_Findkth(P->right,k-(lsize(P)+P->cnt));
125. **else**
126. **return** P;
127. }
128. **void** Treap\_Clear(Treap\_Node \*&root){
129. **if** (root->left)
130. Treap\_Clear(root->left);
131. **if** (root->right)
132. Treap\_Clear(root->right);
133. **delete** root;
134. root = NULL;
135. }
137. stack<**int**> stk;
138. **void** push(){
139. **int** tmp;
140. cin >> tmp;
141. stk.push(tmp);
142. Treap\_Insert(root,tmp);
143. }
144. **void** pop(){
145. **if** (stk.empty()){
146. cout << "Invalid\n";
147. }**else**{
148. cout << stk.top() << '\n';
149. Treap\_Delete(root,stk.top());
150. stk.pop();
151. }
152. }
153. **void** peekmedian(){
154. **if** (stk.empty())
155. cout << "Invalid\n";
156. **else** {
157. Treap\_Node \*T = Treap\_Findkth(root,(stk.size()+1)/2);
158. cout << T->value << '\n';
159. }
160. }
162. **void** InOrderTravel(Treap\_Node\* root){
163. **if** (root->left)  InOrderTravel(root->left);
164. cout << root->value << ' ' << root->cnt << '\n';
165. **if** (root->right) InOrderTravel(root->right);
166. }
167. **int** main()
168. {
169. std::ios::sync\_with\_stdio(**false**);cin.tie(0);
170. **int** i,j,k;
171. **int** n;
172. cin >> n;
173. **char** cmd[100];
174. **for** (i=1;i<=n;i++){
175. cin >> cmd;
176. **switch** (cmd[1]){
177. **case** 'u':push();**break**;
178. **case** 'o':pop();**break**;
179. **case** 'e':peekmedian();**break**;
180. }
181. }
182. **return** 0;
183. }



速度吗，还是可以的，貌似3号样例有一点点化时间

两个multiset的思路：

来自：https://blog.csdn.net/kakitgogogo/article/details/51926600

模拟stack，不过这里的stack加多一个功能，就是输出中位数。用一个stack数据结构来模拟栈的功能，而为了方便得到中位数，要同时维护两个multiset数据结构（因为值可以重复，所以要用multiset）。一个set储存前一半的数（设为s1），一个set储存后一半的数（设为s2），这里s1的大小要和s2的大小一样或比s2的大小大1。同时更新中位数值mid，mid就是s1的最后一个数。在push的时候，把值push到stack中，同时，如果push的值小于等于mid就插入到s1，否则插入到s2，最后为了使他们的大小符合上面的描述，所以要调整一下，更新s1，s2和mid。pop的时候，对stack进行pop操作，同时，如果pop出来的值小于等于mid，就在s1中找出该值然后删除，否则在s2中找出该值然后删除，当前最后也要调整一下，更新s1，s2和mid。最后找中位数就直接找就行了。这里要注意的是multiset删除操作中，不能用实值作为参数，因为这样会把所有的这个值都删去，这里要先用find函数找出其中一个然后再删除。

# 1058. A+B in Hogwarts (20)

时间限制

50 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

If you are a fan of Harry Potter, you would know the world of magic has its own currency system -- as Hagrid explained it to Harry, "Seventeen silver Sickles to a Galleon and twenty-nine Knuts to a Sickle, it's easy enough." Your job is to write a program to compute A+B where A and B are given in the standard form of "Galleon.Sickle.Knut" (Galleon is an integer in [0, 107], Sickle is an integer in [0, 17), and Knut is an integer in [0, 29)).

**Input Specification:**

Each input file contains one test case which occupies a line with A and B in the standard form, separated by one space.

**Output Specification:**

For each test case you should output the sum of A and B in one line, with the same format as the input.

**Sample Input:**

3.2.1 10.16.27

**Sample Output:**

14.1.28

注意都转换成K的话有可能超int，要用long long

# 1059. Prime Factors (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

HE, Qinming

Given any positive integer N, you are supposed to find all of its prime factors, and write them in the format N = p1^k1\* p2^k2 \*…\*pm^km.

**Input Specification:**

Each input file contains one test case which gives a positive integer N in the range of long int.

**Output Specification:**

Factor N in the format N = p1^k1 \* p2^k2 \*…\*pm^km, where pi's are prime factors of N in increasing order, and the exponent ki is the number of pi -- hence when there is only one pi, ki is 1 and must NOT be printed out.

**Sample Input:**

97532468

**Sample Output:**

97532468=2^2\*11\*17\*101\*1291

# 1060. Are They Equal (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

If a machine can save only 3 significant digits, the float numbers 12300 and 12358.9 are considered equal since they are both saved as 0.123\*105 with simple chopping. Now given the number of significant digits on a machine and two float numbers, you are supposed to tell if they are treated equal in that machine.

**Input Specification:**

Each input file contains one test case which gives three numbers N, A and B, where N (<100) is the number of significant digits, and A and B are the two float numbers to be compared. Each float number is non-negative, no greater than 10100, and that its total digit number is less than 100.

**Output Specification:**

For each test case, print in a line "YES" if the two numbers are treated equal, and then the number in the standard form "0.d1...dN\*10^k" (d1>0 unless the number is 0); or "NO" if they are not treated equal, and then the two numbers in their standard form. All the terms must be separated by a space, with no extra space at the end of a line.

Note: Simple chopping is assumed without rounding.

**Sample Input 1:**

3 12300 12358.9

**Sample Output 1:**

YES 0.123\*10^5

**Sample Input 2:**

3 120 128

**Sample Output 2:**

NO 0.120\*10^3 0.128\*10^3

# 1061. Dating (20)

时间限制

150 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Sherlock Holmes received a note with some strange strings: "Let's date! 3485djDkxh4hhGE 2984akDfkkkkggEdsb s&hgsfdk d&Hyscvnm". It took him only a minute to figure out that those strange strings are actually referring to the coded time "Thursday 14:04" -- since the first common capital English letter (case sensitive) shared by the first two strings is the 4th capital letter 'D', representing the 4th day in a week; the second common character is the 5th capital letter 'E', representing the 14th hour (hence the hours from 0 to 23 in a day are represented by the numbers from 0 to 9 and the capital letters from A to N, respectively); and the English letter shared by the last two strings is 's' at the 4th position, representing the 4th minute. Now given two pairs of strings, you are supposed to help Sherlock decode the dating time.

**Input Specification:**

Each input file contains one test case. Each case gives 4 non-empty strings of no more than 60 characters without white space in 4 lines.

**Output Specification:**

For each test case, print the decoded time in one line, in the format "DAY HH:MM", where "DAY" is a 3-character abbreviation for the days in a week -- that is, "MON" for Monday, "TUE" for Tuesday, "WED" for Wednesday, "THU" for Thursday, "FRI" for Friday, "SAT" for Saturday, and "SUN" for Sunday. It is guaranteed that the result is unique for each case.

**Sample Input:**

3485djDkxh4hhGE

2984akDfkkkkggEdsb

s&hgsfdk

d&Hyscvnm

**Sample Output:**

THU 14:04

# 1062. Talent and Virtue (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Li

About 900 years ago, a Chinese philosopher Sima Guang wrote a history book in which he talked about people's talent and virtue. According to his theory, a man being outstanding in both talent and virtue must be a "sage（圣人）"; being less excellent but with one's virtue outweighs talent can be called a "nobleman（君子）"; being good in neither is a "fool man（愚人）"; yet a fool man is better than a "small man（小人）" who prefers talent than virtue.

Now given the grades of talent and virtue of a group of people, you are supposed to rank them according to Sima Guang's theory.

**Input Specification:**

Each input file contains one test case. Each case first gives 3 positive integers in a line: N (<=105), the total number of people to be ranked; L (>=60), the lower bound of the qualified grades -- that is, only the ones whose grades of talent and virtue are both not below this line will be ranked; and H (<100), the higher line of qualification -- that is, those with both grades not below this line are considered as the "sages", and will be ranked in non-increasing order according to their total grades. Those with talent grades below H but virtue grades not are cosidered as the "noblemen", and are also ranked in non-increasing order according to their total grades, but they are listed after the "sages". Those with both grades below H, but with virtue not lower than talent are considered as the "fool men". They are ranked in the same way but after the "noblemen". The rest of people whose grades both pass the L line are ranked after the "fool men".

Then N lines follow, each gives the information of a person in the format:

ID\_Number Virtue\_Grade Talent\_Grade

where ID\_Number is an 8-digit number, and both grades are integers in [0, 100]. All the numbers are separated by a space.

**Output Specification:**

The first line of output must give M (<=N), the total number of people that are actually ranked. Then M lines follow, each gives the information of a person in the same format as the input, according to the ranking rules. If there is a tie of the total grade, they must be ranked with respect to their virtue grades in non-increasing order. If there is still a tie, then output in increasing order of their ID's.

**Sample Input:**

14 60 80

10000001 64 90

10000002 90 60

10000011 85 80

10000003 85 80

10000004 80 85

10000005 82 77

10000006 83 76

10000007 90 78

10000008 75 79

10000009 59 90

10000010 88 45

10000012 80 100

10000013 90 99

10000014 66 60

**Sample Output:**

12

10000013 90 99

10000012 80 100

10000003 85 80

10000011 85 80

10000004 80 85

10000007 90 78

10000006 83 76

10000005 82 77

10000002 90 60

10000014 66 60

10000008 75 79

10000001 64 90

# 1063. Set Similarity (25)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given two sets of integers, the similarity of the sets is defined to be Nc/Nt\*100%, where Nc is the number of distinct common numbers shared by the two sets, and Nt is the total number of distinct numbers in the two sets. Your job is to calculate the similarity of any given pair of sets.

**Input Specification:**

Each input file contains one test case. Each case first gives a positive integer N (<=50) which is the total number of sets. Then N lines follow, each gives a set with a positive M (<=104) and followed by M integers in the range [0, 109]. After the input of sets, a positive integer K (<=2000) is given, followed by K lines of queries. Each query gives a pair of set numbers (the sets are numbered from 1 to N). All the numbers in a line are separated by a space.

**Output Specification:**

For each query, print in one line the similarity of the sets, in the percentage form accurate up to 1 decimal place.

**Sample Input:**

3

3 99 87 101

4 87 101 5 87

7 99 101 18 5 135 18 99

2

1 2

1 3

**Sample Output:**

50.0%

33.3%

# 1064. Complete Binary Search Tree (30)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A Binary Search Tree (BST) is recursively defined as a binary tree which has the following properties:

* The left subtree of a node contains only nodes with keys less than the node's key.
* The right subtree of a node contains only nodes with keys greater than or equal to the node's key.
* Both the left and right subtrees must also be binary search trees.

A Complete Binary Tree (CBT) is a tree that is completely filled, with the possible exception of the bottom level, which is filled from left to right.

Now given a sequence of distinct non-negative integer keys, a unique BST can be constructed if it is required that the tree must also be a CBT. You are supposed to output the level order traversal sequence of this BST.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (<=1000). Then N distinct non-negative integer keys are given in the next line. All the numbers in a line are separated by a space and are no greater than 2000.

**Output Specification:**

For each test case, print in one line the level order traversal sequence of the corresponding complete binary search tree. All the numbers in a line must be separated by a space, and there must be no extra space at the end of the line.

**Sample Input:**

10

1 2 3 4 5 6 7 8 9 0

**Sample Output:**

6 3 8 1 5 7 9 0 2 4

# 1065. A+B and C (64bit) (20)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

HOU, Qiming

Given three integers A, B and C in [-263, 263], you are supposed to tell whether A+B > C.

**Input Specification:**

The first line of the input gives the positive number of test cases, T (<=10). Then T test cases follow, each consists of a single line containing three integers A, B and C, separated by single spaces.

**Output Specification:**

For each test case, output in one line "Case #X: true" if A+B>C, or "Case #X: false" otherwise, where X is the case number (starting from 1).

**Sample Input:**

3

1 2 3

2 3 4

9223372036854775807 -9223372036854775808 0

**Sample Output:**

Case #1: false

Case #2: true

Case #3: false

# 1066. Root of AVL Tree (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

An AVL tree is a self-balancing binary search tree. In an AVL tree, the heights of the two child subtrees of any node differ by at most one; if at any time they differ by more than one, rebalancing is done to restore this property. Figures 1-4 illustrate the rotation rules.

Now given a sequence of insertions, you are supposed to tell the root of the resulting AVL tree.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (<=20) which is the total number of keys to be inserted. Then N distinct integer keys are given in the next line. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print ythe root of the resulting AVL tree in one line.

**Sample Input 1:**

5

88 70 61 96 120

**Sample Output 1:**

70

**Sample Input 2:**

7

88 70 61 96 120 90 65

**Sample Output 2:**

88

# 1067. Sort with Swap(0,\*) (25)

时间限制

150 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given any permutation of the numbers {0, 1, 2,..., N-1}, it is easy to sort them in increasing order. But what if Swap(0, \*) is the ONLY operation that is allowed to use? For example, to sort {4, 0, 2, 1, 3} we may apply the swap operations in the following way:

Swap(0, 1) => {4, 1, 2, 0, 3}  
Swap(0, 3) => {4, 1, 2, 3, 0}  
Swap(0, 4) => {0, 1, 2, 3, 4}

Now you are asked to find the minimum number of swaps need to sort the given permutation of the first N nonnegative integers.

**Input Specification:**

Each input file contains one test case, which gives a positive N (<=105) followed by a permutation sequence of {0, 1, ..., N-1}. All the numbers in a line are separated by a space.

**Output Specification:**

For each case, simply print in a line the minimum number of swaps need to sort the given permutation.

**Sample Input:**

10 3 5 7 2 6 4 9 0 8 1

**Sample Output:**

9

# 1068. Find More Coins (30)

时间限制

150 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Eva loves to collect coins from all over the universe, including some other planets like Mars. One day she visited a universal shopping mall which could accept all kinds of coins as payments. However, there was a special requirement of the payment: for each bill, she must pay the exact amount. Since she has as many as 104 coins with her, she definitely needs your help. You are supposed to tell her, for any given amount of money, whether or not she can find some coins to pay for it.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive numbers: N (<=104, the total number of coins) and M(<=102, the amount of money Eva has to pay). The second line contains N face values of the coins, which are all positive numbers. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print in one line the face values V1 <= V2 <= ... <= Vk such that V1 + V2 + ... + Vk = M. All the numbers must be separated by a space, and there must be no extra space at the end of the line. If such a solution is not unique, output the smallest sequence. If there is no solution, output "No Solution" instead.

Note: sequence {A[1], A[2], ...} is said to be "smaller" than sequence {B[1], B[2], ...} if there exists k >= 1 such that A[i]=B[i] for all i < k, and A[k] < B[k].

**Sample Input 1:**

8 9

5 9 8 7 2 3 4 1

**Sample Output 1:**

1 3 5

**Sample Input 2:**

4 8

7 2 4 3

**Sample Output 2:**

No Solution

# 1069. The Black Hole of Numbers (20)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

For any 4-digit integer except the ones with all the digits being the same, if we sort the digits in non-increasing order first, and then in non-decreasing order, a new number can be obtained by taking the second number from the first one. Repeat in this manner we will soon end up at the number 6174 -- the "black hole" of 4-digit numbers. This number is named Kaprekar Constant.

For example, start from 6767, we'll get:

7766 - 6677 = 1089  
9810 - 0189 = 9621  
9621 - 1269 = 8352  
8532 - 2358 = 6174  
7641 - 1467 = 6174  
... ...

Given any 4-digit number, you are supposed to illustrate the way it gets into the black hole.

**Input Specification:**

Each input file contains one test case which gives a positive integer N in the range (0, 10000).

**Output Specification:**

If all the 4 digits of N are the same, print in one line the equation "N - N = 0000". Else print each step of calculation in a line until 6174 comes out as the difference. All the numbers must be printed as 4-digit numbers.

**Sample Input 1:**

6767

**Sample Output 1:**

7766 - 6677 = 1089

9810 - 0189 = 9621

9621 - 1269 = 8352

8532 - 2358 = 6174

**Sample Input 2:**

2222

**Sample Output 2:**

2222 - 2222 = 0000

# 1070. Mooncake (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Mooncake is a Chinese bakery product traditionally eaten during the Mid-Autumn Festival. Many types of fillings and crusts can be found in traditional mooncakes according to the region's culture. Now given the inventory amounts and the prices of all kinds of the mooncakes, together with the maximum total demand of the market, you are supposed to tell the maximum profit that can be made.

Note: partial inventory storage can be taken. The sample shows the following situation: given three kinds of mooncakes with inventory amounts being 180, 150, and 100 thousand tons, and the prices being 7.5, 7.2, and 4.5 billion yuans. If the market demand can be at most 200 thousand tons, the best we can do is to sell 150 thousand tons of the second kind of mooncake, and 50 thousand tons of the third kind. Hence the total profit is 7.2 + 4.5/2 = 9.45 (billion yuans).

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive integers N (<=1000), the number of different kinds of mooncakes, and D (<=500 thousand tons), the maximum total demand of the market. Then the second line gives the positive inventory amounts (in thousand tons), and the third line gives the positive prices (in billion yuans) of N kinds of mooncakes. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print the maximum profit (in billion yuans) in one line, accurate up to 2 decimal places.

**Sample Input:**

3 200

180 150 100

7.5 7.2 4.5

**Sample Output:**

9.45

# 1071. Speech Patterns (25)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

HOU, Qiming

People often have a preference among synonyms of the same word. For example, some may prefer "the police", while others may prefer "the cops". Analyzing such patterns can help to narrow down a speaker's identity, which is useful when validating, for example, whether it's still the same person behind an online avatar.

Now given a paragraph of text sampled from someone's speech, can you find the person's most commonly used word?

**Input Specification:**

Each input file contains one test case. For each case, there is one line of text no more than 1048576 characters in length, terminated by a carriage return '\n'. The input contains at least one alphanumerical character, i.e., one character from the set [0-9 A-Z a-z].

**Output Specification:**

For each test case, print in one line the most commonly occurring word in the input text, followed by a space and the number of times it has occurred in the input. If there are more than one such words, print the lexicographically smallest one. The word should be printed in all lower case. Here a "word" is defined as a continuous sequence of alphanumerical characters separated by non-alphanumerical characters or the line beginning/end.

Note that words are case **insensitive**.

**Sample Input:**

Can1: "Can a can can a can? It can!"

**Sample Output:**

can 5

# 1072. Gas Station (30)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A gas station has to be built at such a location that the minimum distance between the station and any of the residential housing is as far away as possible. However it must guarantee that all the houses are in its service range.

Now given the map of the city and several candidate locations for the gas station, you are supposed to give the best recommendation. If there are more than one solution, output the one with the smallest average distance to all the houses. If such a solution is still not unique, output the one with the smallest index number.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 4 positive integers: N (<= 103), the total number of houses; M (<= 10), the total number of the candidate locations for the gas stations; K (<= 104), the number of roads connecting the houses and the gas stations; and DS, the maximum service range of the gas station. It is hence assumed that all the houses are numbered from 1 to N, and all the candidate locations are numbered from G1 to GM.

Then K lines follow, each describes a road in the format  
P1 P2 Dist  
where P1 and P2 are the two ends of a road which can be either house numbers or gas station numbers, and Dist is the integer length of the road.

**Output Specification:**

For each test case, print in the first line the index number of the best location. In the next line, print the minimum and the average distances between the solution and all the houses. The numbers in a line must be separated by a space and be accurate up to 1 decimal place. If the solution does not exist, simply output “No Solution”.

**Sample Input 1:**

4 3 11 5

1 2 2

1 4 2

1 G1 4

1 G2 3

2 3 2

2 G2 1

3 4 2

3 G3 2

4 G1 3

G2 G1 1

G3 G2 2

**Sample Output 1:**

G1

2.0 3.3

**Sample Input 2:**

2 1 2 10

1 G1 9

2 G1 20

**Sample Output 2:**

No Solution

# 1073. Scientific Notation (20)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

HOU, Qiming

Scientific notation is the way that scientists easily handle very large numbers or very small numbers. The notation matches the regular expression [+-][1-9]"."[0-9]+E[+-][0-9]+ which means that the integer portion has exactly one digit, there is at least one digit in the fractional portion, and the number and its exponent's signs are always provided even when they are positive.

Now given a real number A in scientific notation, you are supposed to print A in the conventional notation while keeping all the significant figures.

**Input Specification:**

Each input file contains one test case. For each case, there is one line containing the real number A in scientific notation. The number is no more than 9999 bytes in length and the exponent's absolute value is no more than 9999.

**Output Specification:**

For each test case, print in one line the input number A in the conventional notation, with all the significant figures kept, including trailing zeros,

**Sample Input 1:**

+1.23400E-03

**Sample Output 1:**

0.00123400

**Sample Input 2:**

-1.2E+10

**Sample Output 2:**

-12000000000

# 1074. Reversing Linked List (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a constant K and a singly linked list L, you are supposed to reverse the links of every K elements on L. For example, given L being 1→2→3→4→5→6, if K = 3, then you must output 3→2→1→6→5→4; if K = 4, you must output 4→3→2→1→5→6.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains the address of the first node, a positive N (<= 105) which is the total number of nodes, and a positive K (<=N) which is the length of the sublist to be reversed. The address of a node is a 5-digit nonnegative integer, and NULL is represented by -1.

Then N lines follow, each describes a node in the format:

*Address Data Next*

where *Address* is the position of the node, *Data* is an integer, and *Next* is the position of the next node.

**Output Specification:**

For each case, output the resulting ordered linked list. Each node occupies a line, and is printed in the same format as in the input.

**Sample Input:**

00100 6 4

00000 4 99999

00100 1 12309

68237 6 -1

33218 3 00000

99999 5 68237

12309 2 33218

**Sample Output:**

00000 4 33218

33218 3 12309

12309 2 00100

00100 1 99999

99999 5 68237

68237 6 -1

# 1075. PAT Judge (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The ranklist of PAT is generated from the status list, which shows the scores of the submittions. This time you are supposed to generate the ranklist for PAT.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 3 positive integers, N (<=104), the total number of users, K (<=5), the total number of problems, and M (<=105), the total number of submittions. It is then assumed that the user id's are 5-digit numbers from 00001 to N, and the problem id's are from 1 to K. The next line contains K positive integers p[i] (i=1, ..., K), where p[i] corresponds to the full mark of the i-th problem. Then M lines follow, each gives the information of a submittion in the following format:

user\_id problem\_id partial\_score\_obtained

where **partial\_score\_obtained** is either -1 if the submittion cannot even pass the compiler, or is an integer in the range [0, p[**problem\_id**]]. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, you are supposed to output the ranklist in the following format:

rank user\_id total\_score s[1] ... s[K]

where **rank** is calculated according to the **total\_score**, and all the users with the same **total\_score** obtain the same **rank**; and **s[i]** is the partial score obtained for the i-th problem. If a user has never submitted a solution for a problem, then "-" must be printed at the corresponding position. If a user has submitted several solutions to solve one problem, then the highest score will be counted.

The ranklist must be printed in non-decreasing order of the ranks. For those who have the same rank, users must be sorted in nonincreasing order according to the number of perfectly solved problems. And if there is still a tie, then they must be printed in increasing order of their id's. For those who has never submitted any solution that can pass the compiler, or has never submitted any solution, they must NOT be shown on the ranklist. It is guaranteed that at least one user can be shown on the ranklist.

**Sample Input:**

7 4 20

20 25 25 30

00002 2 12

00007 4 17

00005 1 19

00007 2 25

00005 1 20

00002 2 2

00005 1 15

00001 1 18

00004 3 25

00002 2 25

00005 3 22

00006 4 -1

00001 2 18

00002 1 20

00004 1 15

00002 4 18

00001 3 4

00001 4 2

00005 2 -1

00004 2 0

**Sample Output:**

1 00002 63 20 25 - 18

2 00005 42 20 0 22 -

2 00007 42 - 25 - 17

2 00001 42 18 18 4 2

5 00004 40 15 0 25 -

# 1076. Forwards on Weibo (30)

时间限制

3000 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Weibo is known as the Chinese version of Twitter. One user on Weibo may have many followers, and may follow many other users as well. Hence a social network is formed with followers relations. When a user makes a post on Weibo, all his/her followers can view and forward his/her post, which can then be forwarded again by their followers. Now given a social network, you are supposed to calculate the maximum potential amount of forwards for any specific user, assuming that only L levels of indirect followers are counted.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive integers: N (<=1000), the number of users; and L (<=6), the number of levels of indirect followers that are counted. Hence it is assumed that all the users are numbered from 1 to N. Then N lines follow, each in the format:

M[i] user\_list[i]

where **M[i]** (<=100) is the total number of people that user[i] follows; and **user\_list[i]** is a list of the M[i] users that are followed by user[i]. It is guaranteed that no one can follow oneself. All the numbers are separated by a space.

Then finally a positive K is given, followed by K **UserID**'s for query.

**Output Specification:**

For each **UserID**, you are supposed to print in one line the maximum potential amount of forwards this user can triger, assuming that everyone who can view the initial post will forward it once, and that only L levels of indirect followers are counted.

**Sample Input:**

7 3

3 2 3 4

0

2 5 6

2 3 1

2 3 4

1 4

1 5

2 2 6

**Sample Output:**

4

5

# 1077. Kuchiguse (20)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

HOU, Qiming

The Japanese language is notorious for its sentence ending particles. Personal preference of such particles can be considered as a reflection of the speaker's personality. Such a preference is called "Kuchiguse" and is often exaggerated artistically in Anime and Manga. For example, the artificial sentence ending particle "nyan~" is often used as a stereotype for characters with a cat-like personality:

 Itai nyan~ (It hurts, nyan~)

 Ninjin wa iyada nyan~ (I hate carrots, nyan~)

Now given a few lines spoken by the same character, can you find her Kuchiguse?

**Input Specification:**

Each input file contains one test case. For each case, the first line is an integer N (2<=N<=100). Following are N file lines of 0~256 (inclusive) characters in length, each representing a character's spoken line. The spoken lines are case sensitive.

**Output Specification:**

For each test case, print in one line the kuchiguse of the character, i.e., the longest common suffix of all N lines. If there is no such suffix, write "nai".

**Sample Input 1:**

3

Itai nyan~

Ninjin wa iyadanyan~

uhhh nyan~

**Sample Output 1:**

nyan~

**Sample Input 2:**

3

Itai!

Ninjinnwaiyada T\_T

T\_T

**Sample Output 2:**

nai

# 1078. Hashing (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The task of this problem is simple: insert a sequence of distinct positive integers into a hash table, and output the positions of the input numbers. The hash function is defined to be "H(key) = key % TSize" where TSize is the maximum size of the hash table. Quadratic probing (with positive increments only) is used to solve the collisions.

Note that the table size is better to be prime. If the maximum size given by the user is not prime, you must re-define the table size to be the smallest prime number which is larger than the size given by the user.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains two positive numbers: MSize (<=104) and N (<=MSize) which are the user-defined table size and the number of input numbers, respectively. Then N distinct positive integers are given in the next line. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print the corresponding positions (index starts from 0) of the input numbers in one line. All the numbers in a line are separated by a space, and there must be no extra space at the end of the line. In case it is impossible to insert the number, print "-" instead.

**Sample Input:**

4 4

10 6 4 15

**Sample Output:**

0 1 4 -

# 1079. Total Sales of Supply Chain (25)

时间限制

250 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A supply chain is a network of retailers（零售商）, distributors（经销商）, and suppliers（供应商）-- everyone involved in moving a product from supplier to customer.

Starting from one root supplier, everyone on the chain buys products from one's supplier in a price P and sell or distribute them in a price that is r% higher than P. Only the retailers will face the customers. It is assumed that each member in the supply chain has exactly one supplier except the root supplier, and there is no supply cycle.

Now given a supply chain, you are supposed to tell the total sales from all the retailers.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains three positive numbers: N (<=105), the total number of the members in the supply chain (and hence their ID's are numbered from 0 to N-1, and the root supplier's ID is 0); P, the unit price given by the root supplier; and r, the percentage rate of price increment for each distributor or retailer. Then N lines follow, each describes a distributor or retailer in the following format:

Ki ID[1] ID[2] ... ID[Ki]

where in the i-th line, Ki is the total number of distributors or retailers who receive products from supplier i, and is then followed by the ID's of these distributors or retailers. Kj being 0 means that the j-th member is a retailer, then instead the total amount of the product will be given after Kj. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print in one line the total sales we can expect from all the retailers, accurate up to 1 decimal place. It is guaranteed that the number will not exceed 1010.

**Sample Input:**

10 1.80 1.00

3 2 3 5

1 9

1 4

1 7

0 7

2 6 1

1 8

0 9

0 4

0 3

**Sample Output:**

42.4

# 1080. Graduate Admission (30)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

It is said that in 2013, there were about 100 graduate schools ready to proceed over 40,000 applications in Zhejiang Province. It would help a lot if you could write a program to automate the admission procedure.

Each applicant will have to provide two grades: the national entrance exam grade GE, and the interview grade GI. The final grade of an applicant is (GE + GI) / 2. The admission rules are:

* The applicants are ranked according to their final grades, and will be admitted one by one from the top of the rank list.
* If there is a tied final grade, the applicants will be ranked according to their national entrance exam grade GE. If still tied, their ranks must be the same.
* Each applicant may have K choices and the admission will be done according to his/her choices: if according to the rank list, it is one's turn to be admitted; and if the quota of one's most preferred shcool is not exceeded, then one will be admitted to this school, or one's other choices will be considered one by one in order. If one gets rejected by all of preferred schools, then this unfortunate applicant will be rejected.
* If there is a tied rank, and if the corresponding applicants are applying to the same school, then that school must admit all the applicants with the same rank, *even if its quota will be exceeded.*

**Input Specification:**

Each input file contains one test case. Each case starts with a line containing three positive integers: N (<=40,000), the total number of applicants; M (<=100), the total number of graduate schools; and K (<=5), the number of choices an applicant may have.

In the next line, separated by a space, there are M positive integers. The *i*-th integer is the quota of the *i*-th graduate school respectively.

Then N lines follow, each contains 2+K integers separated by a space. The first 2 integers are the applicant's GE and GI, respectively. The next K integers represent the preferred schools. For the sake of simplicity, we assume that the schools are numbered from 0 to M-1, and the applicants are numbered from 0 to N-1.

**Output Specification:**

For each test case you should output the admission results for all the graduate schools. The results of each school must occupy a line, which contains the applicants' numbers that school admits. The numbers must be in increasing order and be separated by a space. There must be no extra space at the end of each line. If no applicant is admitted by a school, you must output an empty line correspondingly.

**Sample Input:**

11 6 3

2 1 2 2 2 3

100 100 0 1 2

60 60 2 3 5

100 90 0 3 4

90 100 1 2 0

90 90 5 1 3

80 90 1 0 2

80 80 0 1 2

80 80 0 1 2

80 70 1 3 2

70 80 1 2 3

100 100 0 2 4

**Sample Output:**

0 10

3

5 6 7

2 8

1 4

# 1081. Rational Sum (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given N rational numbers in the form "numerator/denominator", you are supposed to calculate their sum.

**Input Specification:**

Each input file contains one test case. Each case starts with a positive integer N (<=100), followed in the next line N rational numbers "a1/b1 a2/b2 ..." where all the numerators and denominators are in the range of "long int". If there is a negative number, then the sign must appear in front of the numerator.

**Output Specification:**

For each test case, output the sum in the simplest form "integer numerator/denominator" where "integer" is the integer part of the sum, "numerator" < "denominator", and the numerator and the denominator have no common factor. You must output only the fractional part if the integer part is 0.

**Sample Input 1:**

5

2/5 4/15 1/30 -2/60 8/3

**Sample Output 1:**

3 1/3

**Sample Input 2:**

2

4/3 2/3

**Sample Output 2:**

2

**Sample Input 3:**

3

1/3 -1/6 1/8

**Sample Output 3:**

7/24

# 1082. Read Number in Chinese (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given an integer with no more than 9 digits, you are supposed to read it in the traditional Chinese way. Output "Fu" first if it is negative. For example, -123456789 is read as "Fu yi Yi er Qian san Bai si Shi wu Wan liu Qian qi Bai ba Shi jiu". Note: zero ("ling") must be handled correctly according to the Chinese tradition. For example, 100800 is "yi Shi Wan ling ba Bai".

**Input Specification:**

Each input file contains one test case, which gives an integer with no more than 9 digits.

**Output Specification:**

For each test case, print in a line the Chinese way of reading the number. The characters are separated by a space and there must be no extra space at the end of the line.

**Sample Input 1:**

-123456789

**Sample Output 1:**

Fu yi Yi er Qian san Bai si Shi wu Wan liu Qian qi Bai ba Shi jiu

**Sample Input 2:**

100800

**Sample Output 2:**

yi Shi Wan ling ba Bai

# 1083. List Grades (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a list of N student records with name, ID and grade. You are supposed to sort the records with respect to the grade in non-increasing order, and output those student records of which the grades are in a given interval.

**Input Specification:**

Each input file contains one test case. Each case is given in the following format:

N

name[1] ID[1] grade[1]

name[2] ID[2] grade[2]

... ...

name[N] ID[N] grade[N]

grade1 grade2

where name[i] and ID[i] are strings of no more than 10 characters with no space, grade[i] is an integer in [0, 100], grade1 and grade2 are the boundaries of the grade's interval. It is guaranteed that all the grades are *distinct*.

**Output Specification:**

For each test case you should output the student records of which the grades are in the given interval [grade1, grade2] and are in non-increasing order. Each student record occupies a line with the student's name and ID, separated by one space. If there is no student's grade in that interval, output "NONE" instead.

**Sample Input 1:**

4

Tom CS000001 59

Joe Math990112 89

Mike CS991301 100

Mary EE990830 95

60 100

**Sample Output 1:**

Mike CS991301

Mary EE990830

Joe Math990112

**Sample Input 2:**

2

Jean AA980920 60

Ann CS01 80

90 95

**Sample Output 2:**

NONE

# 1084. Broken Keyboard (20)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

On a broken keyboard, some of the keys are worn out. So when you type some sentences, the characters corresponding to those keys will not appear on screen.

Now given a string that you are supposed to type, and the string that you actually type out, please list those keys which are for sure worn out.

**Input Specification:**

Each input file contains one test case. For each case, the 1st line contains the original string, and the 2nd line contains the typed-out string. Each string contains no more than 80 characters which are either English letters [A-Z] (case insensitive), digital numbers [0-9], or "\_" (representing the space). It is guaranteed that both strings are non-empty.

**Output Specification:**

For each test case, print in one line the keys that are worn out, in the order of being detected. The English letters must be capitalized. Each worn out key must be printed once only. It is guaranteed that there is at least one worn out key.

**Sample Input:**

7\_This\_is\_a\_test

\_hs\_s\_a\_es

**Sample Output:**

7TI

# 1085. Perfect Sequence (25)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CAO, Peng

Given a sequence of positive integers and another positive integer p. The sequence is said to be a "perfect sequence" if M <= m \* p where M and m are the maximum and minimum numbers in the sequence, respectively.

Now given a sequence and a parameter p, you are supposed to find from the sequence as many numbers as possible to form a perfect subsequence.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains two positive integers N and p, where N (<= 105) is the number of integers in the sequence, and p (<= 109) is the parameter. In the second line there are N positive integers, each is no greater than 109.

**Output Specification:**

For each test case, print in one line the maximum number of integers that can be chosen to form a perfect subsequence.

**Sample Input:**

10 8

2 3 20 4 5 1 6 7 8 9

**Sample Output:**

8

# 1086. Tree Traversals Again (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

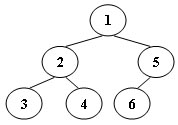
判题程序

Standard

作者

CHEN, Yue

An inorder binary tree traversal can be implemented in a non-recursive way with a stack. For example, suppose that when a 6-node binary tree (with the keys numbered from 1 to 6) is traversed, the stack operations are: push(1); push(2); push(3); pop(); pop(); push(4); pop(); pop(); push(5); push(6); pop(); pop(). Then a unique binary tree (shown in Figure 1) can be generated from this sequence of operations. Your task is to give the postorder traversal sequence of this tree.

  
Figure 1

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (<=30) which is the total number of nodes in a tree (and hence the nodes are numbered from 1 to N). Then 2N lines follow, each describes a stack operation in the format: "Push X" where X is the index of the node being pushed onto the stack; or "Pop" meaning to pop one node from the stack.

**Output Specification:**

For each test case, print the postorder traversal sequence of the corresponding tree in one line. A solution is guaranteed to exist. All the numbers must be separated by exactly one space, and there must be no extra space at the end of the line.

**Sample Input:**

6

Push 1

Push 2

Push 3

Pop

Pop

Push 4

Pop

Pop

Push 5

Push 6

Pop

Pop

**Sample Output:**

3 4 2 6 5 1

# 1087. All Roads Lead to Rome (30)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Indeed there are many different tourist routes from our city to Rome. You are supposed to find your clients the route with the least cost while gaining the most happiness.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive integers N (2<=N<=200), the number of cities, and K, the total number of routes between pairs of cities; followed by the name of the starting city. The next N-1 lines each gives the name of a city and an integer that represents the happiness one can gain from that city, except the starting city. Then K lines follow, each describes a route between two cities in the format "City1 City2 Cost". Here the name of a city is a string of 3 capital English letters, and the destination is always ROM which represents Rome.

**Output Specification:**

For each test case, we are supposed to find the route with the least cost. If such a route is not unique, the one with the maximum happiness will be recommended. If such a route is still not unique, then we output the one with the maximum average happiness -- it is guaranteed by the judge that such a solution exists and is unique.

Hence in the first line of output, you must print 4 numbers: the number of different routes with the least cost, the cost, the happiness, and the average happiness (take the integer part only) of the recommended route. Then in the next line, you are supposed to print the route in the format "City1->City2->...->ROM".

**Sample Input:**

6 7 HZH

ROM 100

PKN 40

GDN 55

PRS 95

BLN 80

ROM GDN 1

BLN ROM 1

HZH PKN 1

PRS ROM 2

BLN HZH 2

PKN GDN 1

HZH PRS 1

**Sample Output:**

3 3 195 97

HZH->PRS->ROM

# 1088. Rational Arithmetic (20)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

For two rational numbers, your task is to implement the basic arithmetics, that is, to calculate their sum, difference, product and quotient.

**Input Specification:**

Each input file contains one test case, which gives in one line the two rational numbers in the format "a1/b1 a2/b2". The numerators and the denominators are all in the range of *long int*. If there is a negative sign, it must appear only in front of the numerator. The denominators are guaranteed to be non-zero numbers.

**Output Specification:**

For each test case, print in 4 lines the sum, difference, product and quotient of the two rational numbers, respectively. The format of each line is "number1 operator number2 = result". Notice that all the rational numbers must be in their simplest form "k a/b", where **k** is the integer part, and **a/b** is the simplest fraction part. If the number is negative, it must be included in a pair of parentheses. If the denominator in the division is zero, output "Inf" as the result. It is guaranteed that all the output integers are in the range of *long int*.

**Sample Input 1:**

2/3 -4/2

**Sample Output 1:**

2/3 + (-2) = (-1 1/3)

2/3 - (-2) = 2 2/3

2/3 \* (-2) = (-1 1/3)

2/3 / (-2) = (-1/3)

**Sample Input 2:**

5/3 0/6

**Sample Output 2:**

1 2/3 + 0 = 1 2/3

1 2/3 - 0 = 1 2/3

1 2/3 \* 0 = 0

1 2/3 / 0 = Inf

# 1089. Insert or Merge (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

According to Wikipedia:

**Insertion sort** iterates, consuming one input element each repetition, and growing a sorted output list. Each iteration, insertion sort removes one element from the input data, finds the location it belongs within the sorted list, and inserts it there. It repeats until no input elements remain.

**Merge sort** works as follows: Divide the unsorted list into N sublists, each containing 1 element (a list of 1 element is considered sorted). Then repeatedly merge two adjacent sublists to produce new sorted sublists until there is only 1 sublist remaining.

Now given the initial sequence of integers, together with a sequence which is a result of several iterations of some sorting method, can you tell which sorting method we are using?

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=100). Then in the next line, N integers are given as the initial sequence. The last line contains the partially sorted sequence of the N numbers. It is assumed that the target sequence is always ascending. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print in the first line either "Insertion Sort" or "Merge Sort" to indicate the method used to obtain the partial result. Then run this method for one more iteration and output in the second line the resulting sequence. It is guaranteed that the answer is unique for each test case. All the numbers in a line must be separated by a space, and there must be no extra space at the end of the line.

**Sample Input 1:**

10

3 1 2 8 7 5 9 4 6 0

1 2 3 7 8 5 9 4 6 0

**Sample Output 1:**

Insertion Sort

1 2 3 5 7 8 9 4 6 0

**Sample Input 2:**

10

3 1 2 8 7 5 9 4 0 6

1 3 2 8 5 7 4 9 0 6

**Sample Output 2:**

Merge Sort

1 2 3 8 4 5 7 9 0 6

# 1090. Highest Price in Supply Chain (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A supply chain is a network of retailers（零售商）, distributors（经销商）, and suppliers（供应商）-- everyone involved in moving a product from supplier to customer.

Starting from one root supplier, everyone on the chain buys products from one's supplier in a price P and sell or distribute them in a price that is r% higher than P. It is assumed that each member in the supply chain has exactly one supplier except the root supplier, and there is no supply cycle.

Now given a supply chain, you are supposed to tell the highest price we can expect from some retailers.

**Input Specification:**

Each input file contains one test case. For each case, The first line contains three positive numbers: N (<=105), the total number of the members in the supply chain (and hence they are numbered from 0 to N-1); P, the price given by the root supplier; and r, the percentage rate of price increment for each distributor or retailer. Then the next line contains N numbers, each number Si is the index of the supplier for the i-th member. Sroot for the root supplier is defined to be -1. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print in one line the highest price we can expect from some retailers, accurate up to 2 decimal places, and the number of retailers that sell at the highest price. There must be one space between the two numbers. It is guaranteed that the price will not exceed 1010.

**Sample Input:**

9 1.80 1.00

1 5 4 4 -1 4 5 3 6

**Sample Output:**

1.85 2

# 1091. Acute Stroke (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

One important factor to identify acute stroke (急性脑卒中) is the volume of the stroke core. Given the results of image analysis in which the core regions are identified in each MRI slice, your job is to calculate the volume of the stroke core.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 4 positive integers: M, N, L and T, where M and N are the sizes of each slice (i.e. pixels of a slice are in an M by N matrix, and the maximum resolution is 1286 by 128); L (<=60) is the number of slices of a brain; and T is the integer threshold (i.e. if the volume of a connected core is less than T, then that core must not be counted).

Then L slices are given. Each slice is represented by an M by N matrix of 0's and 1's, where 1 represents a pixel of stroke, and 0 means normal. Since the thickness of a slice is a constant, we only have to count the number of 1's to obtain the volume. However, there might be several separated core regions in a brain, and only those with their volumes no less than T are counted. Two pixels are "connected" and hence belong to the same region if they share a common side, as shown by Figure 1 where all the 6 red pixels are connected to the blue one.

  
Figure 1

**Output Specification:**

For each case, output in a line the total volume of the stroke core.

**Sample Input:**

3 4 5 2

1 1 1 1

1 1 1 1

1 1 1 1

0 0 1 1

0 0 1 1

0 0 1 1

1 0 1 1

0 1 0 0

0 0 0 0

1 0 1 1

0 0 0 0

0 0 0 0

0 0 0 1

0 0 0 1

1 0 0 0

**Sample Output:**

26

# 1092. To Buy or Not to Buy (20)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Eva would like to make a string of beads with her favorite colors so she went to a small shop to buy some beads. There were many colorful strings of beads. However the owner of the shop would only sell the strings in whole pieces. Hence Eva must check whether a string in the shop contains all the beads she needs. She now comes to you for help: if the answer is "Yes", please tell her the number of extra beads she has to buy; or if the answer is "No", please tell her the number of beads missing from the string.

For the sake of simplicity, let's use the characters in the ranges [0-9], [a-z], and [A-Z] to represent the colors. For example, the 3rd string in Figure 1 is the one that Eva would like to make. Then the 1st string is okay since it contains all the necessary beads with 8 extra ones; yet the 2nd one is not since there is no black bead and one less red bead.

  
Figure 1

**Input Specification:**

Each input file contains one test case. Each case gives in two lines the strings of no more than 1000 beads which belong to the shop owner and Eva, respectively.

**Output Specification:**

For each test case, print your answer in one line. If the answer is "Yes", then also output the number of extra beads Eva has to buy; or if the answer is "No", then also output the number of beads missing from the string. There must be exactly 1 space between the answer and the number.

**Sample Input 1:**

ppRYYGrrYBR2258

YrR8RrY

**Sample Output 1:**

Yes 8

**Sample Input 2:**

ppRYYGrrYB225

YrR8RrY

**Sample Output 1:**

No 2

# 1093. Count PAT's (25)

时间限制

120 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CAO, Peng

The string *APPAPT* contains two *PAT*'s as substrings. The first one is formed by the 2nd, the 4th, and the 6th characters, and the second one is formed by the 3rd, the 4th, and the 6th characters.

Now given any string, you are supposed to tell the number of *PAT*'s contained in the string.

**Input Specification:**

Each input file contains one test case. For each case, there is only one line giving a string of no more than 105characters containing only P, A, or T.

**Output Specification:**

For each test case, print in one line the number of *PAT*'s contained in the string. Since the result may be a huge number, you only have to output the result moded by 1000000007.

**Sample Input:**

APPAPT

**Sample Output:**

2

# 1094. The Largest Generation (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A family hierarchy is usually presented by a pedigree tree where all the nodes on the same level belong to the same generation. Your task is to find the generation with the largest population.

**Input Specification:**

Each input file contains one test case. Each case starts with two positive integers N (<100) which is the total number of family members in the tree (and hence assume that all the members are numbered from 01 to N), and M (<N) which is the number of family members who have children. Then M lines follow, each contains the information of a family member in the following format:

ID K ID[1] ID[2] ... ID[K]

where ID is a two-digit number representing a family member, K (>0) is the number of his/her children, followed by a sequence of two-digit ID's of his/her children. For the sake of simplicity, let us fix the root ID to be 01. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print in one line the largest population number and the level of the corresponding generation. It is assumed that such a generation is unique, and the root level is defined to be 1.

**Sample Input:**

23 13

21 1 23

01 4 03 02 04 05

03 3 06 07 08

06 2 12 13

13 1 21

08 2 15 16

02 2 09 10

11 2 19 20

17 1 22

05 1 11

07 1 14

09 1 17

10 1 18

**Sample Output:**

9 4

# 1095. Cars on Campus (30)

时间限制

220 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Zhejiang University has 6 campuses and a lot of gates. From each gate we can collect the in/out times and the plate numbers of the cars crossing the gate. Now with all the information available, you are supposed to tell, at any specific time point, the number of cars parking on campus, and at the end of the day find the cars that have parked for the longest time period.

**Input Specification:**

Each input file contains one test case. Each case starts with two positive integers N (<= 10000), the number of records, and K (<= 80000) the number of queries. Then N lines follow, each gives a record in the format

plate\_number hh:mm:ss status

where **plate\_number** is a string of 7 English capital letters or 1-digit numbers; **hh:mm:ss** represents the time point in a day by hour:minute:second, with the earliest time being 00:00:00 and the latest 23:59:59; and **status** is either **in** or **out**.

Note that all times will be within a single day. Each "in" record is paired with the chronologically next record for the same car provided it is an "out" record. Any "in" records that are not paired with an "out" record are ignored, as are "out" records not paired with an "in" record. It is guaranteed that at least one car is well paired in the input, and no car is both "in" and "out" at the same moment. Times are recorded using a 24-hour clock.

Then K lines of queries follow, each gives a time point in the format **hh:mm:ss**. Note: the queries are given in **ascending** order of the times.

**Output Specification:**

For each query, output in a line the total number of cars parking on campus. The last line of output is supposed to give the plate number of the car that has parked for the longest time period, and the corresponding time length. If such a car is not unique, then output all of their plate numbers in a line in alphabetical order, separated by a space.

**Sample Input:**

16 7

JH007BD 18:00:01 in

ZD00001 11:30:08 out

DB8888A 13:00:00 out

ZA3Q625 23:59:50 out

ZA133CH 10:23:00 in

ZD00001 04:09:59 in

JH007BD 05:09:59 in

ZA3Q625 11:42:01 out

JH007BD 05:10:33 in

ZA3Q625 06:30:50 in

JH007BD 12:23:42 out

ZA3Q625 23:55:00 in

JH007BD 12:24:23 out

ZA133CH 17:11:22 out

JH007BD 18:07:01 out

DB8888A 06:30:50 in

05:10:00

06:30:50

11:00:00

12:23:42

14:00:00

18:00:00

23:59:00

**Sample Output:**

1

4

5

2

1

0

1

JH007BD ZD00001 07:20:09

# 1096. Consecutive Factors (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Among all the factors of a positive integer N, there may exist several consecutive numbers. For example, 630 can be factored as 3\*5\*6\*7, where 5, 6, and 7 are the three consecutive numbers. Now given any positive N, you are supposed to find the maximum number of consecutive factors, and list the smallest sequence of the consecutive factors.

**Input Specification:**

Each input file contains one test case, which gives the integer N (1<N<231).

**Output Specification:**

For each test case, print in the first line the maximum number of consecutive factors. Then in the second line, print the smallest sequence of the consecutive factors in the format "factor[1]\*factor[2]\*...\*factor[k]", where the factors are listed in increasing order, and 1 is NOT included.

**Sample Input:**

630

**Sample Output:**

3

5\*6\*7

# 1097. Deduplication on a Linked List (25)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a singly linked list L with integer keys, you are supposed to remove the nodes with duplicated absolute values of the keys. That is, for each value K, only the first node of which the value or absolute value of its key equals K will be kept. At the mean time, all the removed nodes must be kept in a separate list. For example, given L being 21→-15→-15→-7→15, you must output 21→-15→-7, and the removed list -15→15.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains the address of the first node, and a positive N (<= 105) which is the total number of nodes. The address of a node is a 5-digit nonnegative integer, and NULL is represented by -1.

Then N lines follow, each describes a node in the format:

*Address Key Next*

where *Address* is the position of the node, *Key* is an integer of which absolute value is no more than 104, and *Next* is the position of the next node.

**Output Specification:**

For each case, output the resulting linked list first, then the removed list. Each node occupies a line, and is printed in the same format as in the input.

**Sample Input:**

00100 5

99999 -7 87654

23854 -15 00000

87654 15 -1

00000 -15 99999

00100 21 23854

**Sample Output:**

00100 21 23854

23854 -15 99999

99999 -7 -1

00000 -15 87654

87654 15 -1

# 1098. Insertion or Heap Sort (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

According to Wikipedia:

**Insertion sort** iterates, consuming one input element each repetition, and growing a sorted output list. At each iteration, insertion sort removes one element from the input data, finds the location it belongs within the sorted list, and inserts it there. It repeats until no input elements remain.

**Heap sort** divides its input into a sorted and an unsorted region, and it iteratively shrinks the unsorted region by extracting the largest element and moving that to the sorted region. it involves the use of a heap data structure rather than a linear-time search to find the maximum.

Now given the initial sequence of integers, together with a sequence which is a result of several iterations of some sorting method, can you tell which sorting method we are using?

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=100). Then in the next line, N integers are given as the initial sequence. The last line contains the partially sorted sequence of the N numbers. It is assumed that the target sequence is always ascending. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print in the first line either "Insertion Sort" or "Heap Sort" to indicate the method used to obtain the partial result. Then run this method for one more iteration and output in the second line the resuling sequence. It is guaranteed that the answer is unique for each test case. All the numbers in a line must be separated by a space, and there must be no extra space at the end of the line.

**Sample Input 1:**

10

3 1 2 8 7 5 9 4 6 0

1 2 3 7 8 5 9 4 6 0

**Sample Output 1:**

Insertion Sort

1 2 3 5 7 8 9 4 6 0

**Sample Input 2:**

10

3 1 2 8 7 5 9 4 6 0

6 4 5 1 0 3 2 7 8 9

**Sample Output 2:**

Heap Sort

5 4 3 1 0 2 6 7 8 9

# 1099. Build A Binary Search Tree (30)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

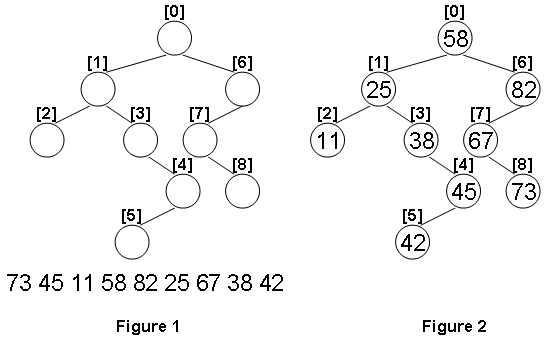
A Binary Search Tree (BST) is recursively defined as a binary tree which has the following properties:

 The left subtree of a node contains only nodes with keys less than the node's key.

 The right subtree of a node contains only nodes with keys greater than or equal to the node's key.

 Both the left and right subtrees must also be binary search trees.

Given the structure of a binary tree and a sequence of distinct integer keys, there is only one way to fill these keys into the tree so that the resulting tree satisfies the definition of a BST. You are supposed to output the level order traversal sequence of that tree. The sample is illustrated by Figure 1 and 2.



**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=100) which is the total number of nodes in the tree. The next N lines each contains the left and the right children of a node in the format "left\_index right\_index", provided that the nodes are numbered from 0 to N-1, and 0 is always the root. If one child is missing, then -1 will represent the NULL child pointer. Finally N distinct integer keys are given in the last line.

**Output Specification:**

For each test case, print in one line the level order traversal sequence of that tree. All the numbers must be separated by a space, with no extra space at the end of the line.

**Sample Input:**

9

1 6

2 3

-1 -1

-1 4

5 -1

-1 -1

7 -1

-1 8

-1 -1

73 45 11 58 82 25 67 38 42

**Sample Output:**

58 25 82 11 38 67 45 73 42

# 1100. Mars Numbers (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

People on Mars count their numbers with base 13:

* Zero on Earth is called "tret" on Mars.
* The numbers 1 to 12 on Earch is called "jan, feb, mar, apr, may, jun, jly, aug, sep, oct, nov, dec" on Mars, respectively.
* For the next higher digit, Mars people name the 12 numbers as "tam, hel, maa, huh, tou, kes, hei, elo, syy, lok, mer, jou", respectively.

For examples, the number 29 on Earth is called "hel mar" on Mars; and "elo nov" on Mars corresponds to 115 on Earth. In order to help communication between people from these two planets, you are supposed to write a program for mutual translation between Earth and Mars number systems.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (< 100). Then N lines follow, each contains a number in [0, 169), given either in the form of an Earth number, or that of Mars.

**Output Specification:**

For each number, print in a line the corresponding number in the other language.

**Sample Input:**

4

29

5

elo nov

tam

**Sample Output:**

hel mar

may

115

13

# 1101. Quick Sort (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CAO, Peng

There is a classical process named *partition* in the famous quick sort algorithm. In this process we typically choose one element as the pivot. Then the elements less than the pivot are moved to its left and those larger than the pivot to its right. Given N distinct positive integers after a run of partition, could you tell how many elements could be the selected pivot for this partition?

For example, given N = 5 and the numbers 1, 3, 2, 4, and 5. We have:

 1 could be the pivot since there is no element to its left and all the elements to its right are larger than it;

 3 must not be the pivot since although all the elements to its left are smaller, the number 2 to its right is less than it as well;

 2 must not be the pivot since although all the elements to its right are larger, the number 3 to its left is larger than it as well;

 and for the similar reason, 4 and 5 could also be the pivot.

Hence in total there are 3 pivot candidates.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<= 105). Then the next line contains N distinct positive integers no larger than 109. The numbers in a line are separated by spaces.

**Output Specification:**

For each test case, output in the first line the number of pivot candidates. Then in the next line print these candidates in increasing order. There must be exactly 1 space between two adjacent numbers, and no extra space at the end of each line.

**Sample Input:**

5

1 3 2 4 5

**Sample Output:**

3

1 4 5

# 1102. Invert a Binary Tree (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The following is from Max Howell @twitter:

*Google: 90% of our engineers use the software you wrote (Homebrew), but you can't invert a binary tree on a whiteboard so fuck off.*

Now it's your turn to prove that YOU CAN invert a binary tree!

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=10) which is the total number of nodes in the tree -- and hence the nodes are numbered from 0 to N-1. Then N lines follow, each corresponds to a node from 0 to N-1, and gives the indices of the left and right children of the node. If the child does not exist, a "-" will be put at the position. Any pair of children are separated by a space.

**Output Specification:**

For each test case, print in the first line the level-order, and then in the second line the in-order traversal sequences of the inverted tree. There must be exactly one space between any adjacent numbers, and no extra space at the end of the line.

**Sample Input:**

8

1 -

- -

0 -

2 7

- -

- -

5 -

4 6

**Sample Output:**

3 7 2 6 4 0 5 1

6 5 7 4 3 2 0 1

# 1103. Integer Factorization (30)

时间限制

1200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The K-P factorization of a positive integer N is to write N as the sum of the P-th power of K positive integers. You are supposed to write a program to find the K-P factorization of N for any positive integers N, K and P.

**Input Specification:**

Each input file contains one test case which gives in a line the three positive integers N (<=400), K (<=N) and P (1<P<=7). The numbers in a line are separated by a space.

**Output Specification:**

For each case, if the solution exists, output in the format:

N = n1^P + ... nK^P

where **ni** (i=1, ... K) is the i-th factor. All the factors must be printed in non-increasing order.

Note: the solution may not be unique. For example, the 5-2 factorization of 169 has 9 solutions, such as 122 + 42 + 22 + 22 + 12, or 112+ 62 + 22 + 22 + 22, or more. You must output the one with the maximum sum of the factors. If there is a tie, the largest factor sequence must be chosen -- sequence { a1, a2, ... aK } is said to be **larger** than { b1, b2, ... bK } if there exists 1<=L<=K such that ai=bi for i<L and aL>bL

If there is no solution, simple output "Impossible".

**Sample Input 1:**

169 5 2

**Sample Output 1:**

169 = 6^2 + 6^2 + 6^2 + 6^2 + 5^2

**Sample Input 2:**

169 167 3

**Sample Output 2:**

Impossible

# 1104. Sum of Number Segments (20)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CAO, Peng

Given a sequence of positive numbers, a *segment* is defined to be a consecutive subsequence. For example, given the sequence {0.1, 0.2, 0.3, 0.4}, we have 10 segments: (0.1) (0.1, 0.2) (0.1, 0.2, 0.3) (0.1, 0.2, 0.3, 0.4) (0.2) (0.2, 0.3) (0.2, 0.3, 0.4) (0.3) (0.3, 0.4) (0.4).

Now given a sequence, you are supposed to find the sum of all the numbers in all the segments. For the previous example, the sum of all the 10 segments is 0.1 + 0.3 + 0.6 + 1.0 + 0.2 + 0.5 + 0.9 + 0.3 + 0.7 + 0.4 = 5.0.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N, the size of the sequence which is no more than 105. The next line contains N positive numbers in the sequence, each no more than 1.0, separated by a space.

**Output Specification:**

For each test case, print in one line the sum of all the numbers in all the segments, accurate up to 2 decimal places.

**Sample Input:**

4

0.1 0.2 0.3 0.4

**Sample Output:**

5.00

# 1105. Spiral Matrix (25)

时间限制

150 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

This time your job is to fill a sequence of N positive integers into a **spiral matrix** in non-increasing order. A **spiral matrix** is filled in from the first element at the upper-left corner, then move in a clockwise spiral. The matrix has **m** rows and **n** columns, where **m** and **n** satisfy the following: **m**\***n** must be equal to N; **m**>=**n**; and **m**-**n** is the minimum of all the possible values.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N. Then the next line contains N positive integers to be filled into the spiral matrix. All the numbers are no more than 104. The numbers in a line are separated by spaces.

**Output Specification:**

For each test case, output the resulting matrix in **m** lines, each contains **n** numbers. There must be exactly 1 space between two adjacent numbers, and no extra space at the end of each line.

**Sample Input:**

12

37 76 20 98 76 42 53 95 60 81 58 93

**Sample Output:**

98 95 93

42 37 81

53 20 76

58 60 76

# 1106. Lowest Price in Supply Chain (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A supply chain is a network of retailers（零售商）, distributors（经销商）, and suppliers（供应商）-- everyone involved in moving a product from supplier to customer.

Starting from one root supplier, everyone on the chain buys products from one's supplier in a price P and sell or distribute them in a price that is r% higher than P. Only the retailers will face the customers. It is assumed that each member in the supply chain has exactly one supplier except the root supplier, and there is no supply cycle.

Now given a supply chain, you are supposed to tell the lowest price a customer can expect from some retailers.

**Input Specification:**

Each input file contains one test case. For each case, The first line contains three positive numbers: N (<=105), the total number of the members in the supply chain (and hence their ID's are numbered from 0 to N-1, and the root supplier's ID is 0); P, the price given by the root supplier; and r, the percentage rate of price increment for each distributor or retailer. Then N lines follow, each describes a distributor or retailer in the following format:

Ki ID[1] ID[2] ... ID[Ki]

where in the i-th line, Ki is the total number of distributors or retailers who receive products from supplier i, and is then followed by the ID's of these distributors or retailers. Kj being 0 means that the j-th member is a retailer. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print in one line the lowest price we can expect from some retailers, accurate up to 4 decimal places, and the number of retailers that sell at the lowest price. There must be one space between the two numbers. It is guaranteed that the all the prices will not exceed 1010.

**Sample Input:**

10 1.80 1.00

3 2 3 5

1 9

1 4

1 7

0

2 6 1

1 8

0

0

0

**Sample Output:**

1.8362 2

# 1107. Social Clusters (30)

时间限制

1000 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

When register on a social network, you are always asked to specify your hobbies in order to find some potential friends with the same hobbies. A "social cluster" is a set of people who have some of their hobbies in common. You are supposed to find all the clusters.

**Input Specification:**

Each input file contains one test case. For each test case, the first line contains a positive integer N (<=1000), the total number of people in a social network. Hence the people are numbered from 1 to N. Then N lines follow, each gives the hobby list of a person in the format:

Ki: hi[1] hi[2] ... hi[Ki]

where Ki (>0) is the number of hobbies, and hi[j] is the index of the j-th hobby, which is an integer in [1, 1000].

**Output Specification:**

For each case, print in one line the total number of clusters in the network. Then in the second line, print the numbers of people in the clusters in non-increasing order. The numbers must be separated by exactly one space, and there must be no extra space at the end of the line.

**Sample Input:**

8

3: 2 7 10

1: 4

2: 5 3

1: 4

1: 3

1: 4

4: 6 8 1 5

1: 4

**Sample Output:**

3

4 3 1

# 1108. Finding Average (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The basic task is simple: given N real numbers, you are supposed to calculate their average. But what makes it complicated is that some of the input numbers might not be legal. A "legal" input is a real number in [-1000, 1000] and is accurate up to no more than 2 decimal places. When you calculate the average, those illegal numbers must not be counted in.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=100). Then N numbers are given in the next line, separated by one space.

**Output Specification:**

For each illegal input number, print in a line "ERROR: X is not a legal number" where X is the input. Then finally print in a line the result: "The average of K numbers is Y" where K is the number of legal inputs and Y is their average, accurate to 2 decimal places. In case the average cannot be calculated, output "Undefined" instead of Y. In case K is only 1, output "The average of 1 number is Y" instead.

**Sample Input 1:**

7

5 -3.2 aaa 9999 2.3.4 7.123 2.35

**Sample Output 1:**

ERROR: aaa is not a legal number

ERROR: 9999 is not a legal number

ERROR: 2.3.4 is not a legal number

ERROR: 7.123 is not a legal number

The average of 3 numbers is 1.38

**Sample Input 2:**

2

aaa -9999

**Sample Output 2:**

ERROR: aaa is not a legal number

ERROR: -9999 is not a legal number

The average of 0 numbers is Undefined

# 1109. Group Photo (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Formation is very important when taking a group photo. Given the rules of forming K rows with N people as the following:

* The number of people in each row must be N/K (round down to the nearest integer), with all the extra people (if any) standing in the last row;
* All the people in the rear row must be no shorter than anyone standing in the front rows;
* In each row, the tallest one stands at the central position (which is defined to be the position (m/2+1), where m is the total number of people in that row, and the division result must be rounded down to the nearest integer);
* In each row, other people must enter the row in non-increasing order of their heights, alternately taking their positions first to the right and then to the left of the tallest one (For example, given five people with their heights 190, 188, 186, 175, and 170, the final formation would be 175, 188, 190, 186, and 170. Here we assume that you are facing the group so your left-hand side is the right-hand side of the one at the central position.);
* When there are many people having the same height, they must be ordered in alphabetical (increasing) order of their names, and it is guaranteed that there is no duplication of names.

Now given the information of a group of people, you are supposed to write a program to output their formation.

**Input Specification:**

Each input file contains one test case. For each test case, the first line contains two positive integers N (<=10000), the total number of people, and K (<=10), the total number of rows. Then N lines follow, each gives the name of a person (no more than 8 English letters without space) and his/her height (an integer in [30, 300]).

**Output Specification:**

For each case, print the formation -- that is, print the names of people in K lines. The names must be separated by exactly one space, but there must be no extra space at the end of each line. Note: since you are facing the group, people in the rear rows must be printed above the people in the front rows.

**Sample Input:**

10 3

Tom 188

Mike 170

Eva 168

Tim 160

Joe 190

Ann 168

Bob 175

Nick 186

Amy 160

John 159

**Sample Output:**

Bob Tom Joe Nick

Ann Mike Eva

Tim Amy John

# 1110. Complete Binary Tree (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a tree, you are supposed to tell if it is a complete binary tree.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=20) which is the total number of nodes in the tree -- and hence the nodes are numbered from 0 to N-1. Then N lines follow, each corresponds to a node, and gives the indices of the left and right children of the node. If the child does not exist, a "-" will be put at the position. Any pair of children are separated by a space.

**Output Specification:**

For each case, print in one line "YES" and the index of the last node if the tree is a complete binary tree, or "NO" and the index of the root if not. There must be exactly one space separating the word and the number.

**Sample Input 1:**

9

7 8

- -

- -

- -

0 1

2 3

4 5

- -

- -

**Sample Output 1:**

YES 8

**Sample Input 2:**

8

- -

4 5

0 6

- -

2 3

- 7

- -

- -

**Sample Output 2:**

NO 1

# 1111. Online Map (30)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Input our current position and a destination, an online map can recommend several paths. Now your job is to recommend two paths to your user: one is the shortest, and the other is the fastest. It is guaranteed that a path exists for any request.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives two positive integers N (2 <= N <= 500), and M, being the total number of streets intersections on a map, and the number of streets, respectively. Then M lines follow, each describes a street in the format:

V1 V2 one-way length time

where **V1** and **V2** are the indices (from 0 to N-1) of the two ends of the street; **one-way** is 1 if the street is one-way from **V1** to **V2**, or 0 if not; **length** is the length of the street; and **time** is the time taken to pass the street.

Finally a pair of source and destination is given.

**Output Specification:**

For each case, first print the shortest path from the source to the destination with distance D in the format:

Distance = D: source -> v1 -> ... -> destination

Then in the next line print the fastest path with total time T:

Time = T: source -> w1 -> ... -> destination

In case the shortest path is not unique, output the fastest one among the shortest paths, which is guaranteed to be unique. In case the fastest path is not unique, output the one that passes through the fewest intersections, which is guaranteed to be unique.

In case the shortest and the fastest paths are identical, print them in one line in the format:

Distance = D; Time = T: source -> u1 -> ... -> destination

**Sample Input 1:**

10 15

0 1 0 1 1

8 0 0 1 1

4 8 1 1 1

3 4 0 3 2

3 9 1 4 1

0 6 0 1 1

7 5 1 2 1

8 5 1 2 1

2 3 0 2 2

2 1 1 1 1

1 3 0 3 1

1 4 0 1 1

9 7 1 3 1

5 1 0 5 2

6 5 1 1 2

3 5

**Sample Output 1:**

Distance = 6: 3 -> 4 -> 8 -> 5

Time = 3: 3 -> 1 -> 5

**Sample Input 2:**

7 9

0 4 1 1 1

1 6 1 1 3

2 6 1 1 1

2 5 1 2 2

3 0 0 1 1

3 1 1 1 3

3 2 1 1 2

4 5 0 2 2

6 5 1 1 2

3 5

**Sample Output 2:**

Distance = 3; Time = 4: 3 -> 2 -> 5

# 1112. Stucked Keyboard (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

On a broken keyboard, some of the keys are always stucked. So when you type some sentences, the characters corresponding to those keys will appear repeatedly on screen for k times.

Now given a resulting string on screen, you are supposed to list all the possible stucked keys, and the original string.

Notice that there might be some characters that are typed repeatedly. The stucked key will always repeat output for a fixed k times whenever it is pressed. For example, when k=3, from the string "thiiis iiisss a teeeeeest" we know that the keys "i" and "e" might be stucked, but "s" is not even though it appears repeatedly sometimes. The original string could be "this isss a teest".

**Input Specification:**

Each input file contains one test case. For each case, the 1st line gives a positive integer k ( 1<k<=100 ) which is the output repeating times of a stucked key. The 2nd line contains the resulting string on screen, which consists of no more than 1000 characters from {a-z}, {0-9} and "\_". It is guaranteed that the string is non-empty.

**Output Specification:**

For each test case, print in one line the possible stucked keys, in the order of being detected. Make sure that each key is printed once only. Then in the next line print the original string. It is guaranteed that there is at least one stucked key.

**Sample Input:**

3

caseee1\_\_thiiis\_iiisss\_a\_teeeeeest

**Sample Output:**

ei

case1\_\_this\_isss\_a\_teest

# 1113. Integer Set Partition (25)

时间限制

150 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a set of N (> 1) positive integers, you are supposed to partition them into two disjoint sets A1 and A2 of n1 and n2 numbers, respectively. Let S1 and S2 denote the sums of all the numbers in A1 and A2, respectively. You are supposed to make the partition so that |n1 - n2| is minimized first, and then |S1 - S2| is maximized.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives an integer N (2 <= N <= 105), and then N positive integers follow in the next line, separated by spaces. It is guaranteed that all the integers and their sum are less than 231.

**Output Specification:**

For each case, print in a line two numbers: |n1 - n2| and |S1 - S2|, separated by exactly one space.

**Sample Input 1:**

10

23 8 10 99 46 2333 46 1 666 555

**Sample Output 1:**

0 3611

**Sample Input 2:**

13

110 79 218 69 3721 100 29 135 2 6 13 5188 85

**Sample Output 2:**

1 9359

# 1114. Family Property (25)

时间限制

150 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

This time, you are supposed to help us collect the data for family-owned property. Given each person's family members, and the estate（房产）info under his/her own name, we need to know the size of each family, and the average area and number of sets of their real estate.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=1000). Then N lines follow, each gives the infomation of a person who owns estate in the format:

**ID Father Mother k Child1 ... Childk M\_estate Area**

where **ID** is a unique 4-digit identification number for each person; **Father** and **Mother** are the ID's of this person's parents (if a parent has passed away, **-1** will be given instead); **k** (0<=k<=5) is the number of children of this person; **Childi**'s are the ID's of his/her children; **M\_estate** is the total number of sets of the real estate under his/her name; and **Area** is the total area of his/her estate.

**Output Specification:**

For each case, first print in a line the number of families (all the people that are related directly or indirectly are considered in the same family). Then output the family info in the format:

**ID M AVG\_sets AVG\_area**

where **ID** is the smallest ID in the family; **M** is the total number of family members; **AVG\_sets** is the average number of sets of their real estate; and **AVG\_area** is the average area. The average numbers must be accurate up to 3 decimal places. The families must be given in descending order of their average areas, and in ascending order of the ID's if there is a tie.

**Sample Input:**

10

6666 5551 5552 1 7777 1 100

1234 5678 9012 1 0002 2 300

8888 -1 -1 0 1 1000

2468 0001 0004 1 2222 1 500

7777 6666 -1 0 2 300

3721 -1 -1 1 2333 2 150

9012 -1 -1 3 1236 1235 1234 1 100

1235 5678 9012 0 1 50

2222 1236 2468 2 6661 6662 1 300

2333 -1 3721 3 6661 6662 6663 1 100

**Sample Output:**

3

8888 1 1.000 1000.000

0001 15 0.600 100.000

5551 4 0.750 100.000

# 1116. Come on! Let's C (20)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

"Let's C" is a popular and fun programming contest hosted by the College of Computer Science and Technology, Zhejiang University. Since the idea of the contest is for fun, the award rules are funny as the following:

0. The Champion will receive a "Mystery Award" (such as a BIG collection of students' research papers...).  
1. Those who ranked as a prime number will receive the best award -- the Minions (小黄人)!  
2. Everyone else will receive chocolates.

Given the final ranklist and a sequence of contestant ID's, you are supposed to tell the corresponding awards.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=10000), the total number of contestants. Then N lines of the ranklist follow, each in order gives a contestant's ID (a 4-digit number). After the ranklist, there is a positive integer K followed by K query ID's.

**Output Specification:**

For each query, print in a line "ID: award" where the award is "Mystery Award", or "Minion", or "Chocolate". If the ID is not in the ranklist, print "Are you kidding?" instead. If the ID has been checked before, print "ID: Checked".

**Sample Input:**

6

1111

6666

8888

1234

5555

0001

6

8888

0001

1111

2222

8888

2222

**Sample Output:**

8888: Minion

0001: Chocolate

1111: Mystery Award

2222: Are you kidding?

8888: Checked

2222: Are you kidding?

# 1117. Eddington Number(25)

时间限制

250 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

British astronomer Eddington liked to ride a bike. It is said that in order to show off his skill, he has even defined an "Eddington number", E -- that is, the maximum integer E such that it is for E days that one rides more than E miles. Eddington's own E was 87.

Now given everyday's distances that one rides for N days, you are supposed to find the corresponding E (<=N).

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N(<=105), the days of continuous riding. Then N non-negative integers are given in the next line, being the riding distances of everyday.

**Output Specification:**

For each case, print in a line the Eddington number for these N days.

**Sample Input:**

10

6 7 6 9 3 10 8 2 7 8

**Sample Output:**

6

# 1118. Birds in Forest (25)

时间限制

150 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Some scientists took pictures of thousands of birds in a forest. Assume that all the birds appear in the same picture belong to the same tree. You are supposed to help the scientists to count the maximum number of trees in the forest, and for any pair of birds, tell if they are on the same tree.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive number N (<= 104) which is the number of pictures. Then N lines follow, each describes a picture in the format:  
K B1 B2 ... BK  
where K is the number of birds in this picture, and Bi's are the indices of birds. It is guaranteed that the birds in all the pictures are numbered continuously from 1 to some number that is no more than 104.

After the pictures there is a positive number Q (<= 104) which is the number of queries. Then Q lines follow, each contains the indices of two birds.

**Output Specification:**

For each test case, first output in a line the maximum possible number of trees and the number of birds. Then for each query, print in a line "Yes" if the two birds belong to the same tree, or "No" if not.

**Sample Input:**

4

3 10 1 2

2 3 4

4 1 5 7 8

3 9 6 4

2

10 5

3 7

**Sample Output:**

2 10

Yes

No

# 1119. Pre- and Post-order Traversals (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Special

作者

CHEN, Yue

Suppose that all the keys in a binary tree are distinct positive integers. A unique binary tree can be determined by a given pair of postorder and inorder traversal sequences, or preorder and inorder traversal sequences. However, if only the postorder and preorder traversal sequences are given, the corresponding tree may no longer be unique.

Now given a pair of postorder and preorder traversal sequences, you are supposed to output the corresponding inorder traversal sequence of the tree. If the tree is not unique, simply output any one of them.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=30), the total number of nodes in the binary tree. The second line gives the preorder sequence and the third line gives the postorder sequence. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, first printf in a line "Yes" if the tree is unique, or "No" if not. Then print in the next line the inorder traversal sequence of the corresponding binary tree. If the solution is not unique, any answer would do. It is guaranteed that at least one solution exists. All the numbers in a line must be separated by exactly one space, and there must be no extra space at the end of the line.

**Sample Input 1:**

7

1 2 3 4 6 7 5

2 6 7 4 5 3 1

**Sample Output 1:**

Yes

2 1 6 4 7 3 5

**Sample Input 2:**

4

1 2 3 4

2 4 3 1

**Sample Output 2:**

No

2 1 3 4

# 1120. Friend Numbers (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Two integers are called "friend numbers" if they share the same sum of their digits, and the sum is their "friend ID". For example, 123 and 51 are friend numbers since 1+2+3 = 5+1 = 6, and 6 is their friend ID. Given some numbers, you are supposed to count the number of different friend ID's among them. Note: a number is considered a friend of itself.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N. Then N positive integers are given in the next line, separated by spaces. All the numbers are less than 104.

**Output Specification:**

For each case, print in the first line the number of different frind ID's among the given integers. Then in the second line, output the friend ID's in increasing order. The numbers must be separated by exactly one space and there must be no extra space at the end of the line.

**Sample Input:**

8

123 899 51 998 27 33 36 12

**Sample Output:**

4

3 6 9 26

# 1121. Damn Single (25)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

"Damn Single (单身狗)" is the Chinese nickname for someone who is being single. You are supposed to find those who are alone in a big party, so they can be taken care of.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=50000), the total number of couples. Then N lines of the couples follow, each gives a couple of ID's which are 5-digit numbers (i.e. from 00000 to 99999). After the list of couples, there is a positive integer M (<=10000) followed by M ID's of the party guests. The numbers are separated by spaces. It is guaranteed that nobody is having bigamous marriage (重婚) or dangling with more than one companion.

**Output Specification:**

First print in a line the total number of lonely guests. Then in the next line, print their ID's in increasing order. The numbers must be separated by exactly 1 space, and there must be no extra space at the end of the line.

**Sample Input:**

3

11111 22222

33333 44444

55555 66666

7

55555 44444 10000 88888 22222 11111 23333

**Sample Output:**

5

10000 23333 44444 55555 88888

# 1122. Hamiltonian Cycle (25)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The "Hamilton cycle problem" is to find a simple cycle that contains every vertex in a graph. Such a cycle is called a "Hamiltonian cycle".

In this problem, you are supposed to tell if a given cycle is a Hamiltonian cycle.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive integers N (2< N <= 200), the number of vertices, and M, the number of edges in an undirected graph. Then M lines follow, each describes an edge in the format "Vertex1 Vertex2", where the vertices are numbered from 1 to N. The next line gives a positive integer K which is the number of queries, followed by K lines of queries, each in the format:

n V1 V2 ... Vn

where n is the number of vertices in the list, and Vi's are the vertices on a path.

**Output Specification:**

For each query, print in a line "YES" if the path does form a Hamiltonian cycle, or "NO" if not.

**Sample Input:**

6 10

6 2

3 4

1 5

2 5

3 1

4 1

1 6

6 3

1 2

4 5

6

7 5 1 4 3 6 2 5

6 5 1 4 3 6 2

9 6 2 1 6 3 4 5 2 6

4 1 2 5 1

7 6 1 3 4 5 2 6

7 6 1 2 5 4 3 1

**Sample Output:**

YES

NO

NO

NO

YES

NO

# 1123. Is It a Complete AVL Tree (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

An AVL tree is a self-balancing binary search tree. In an AVL tree, the heights of the two child subtrees of any node differ by at most one; if at any time they differ by more than one, rebalancing is done to restore this property. Figures 1-4 illustrate the rotation rules.

Now given a sequence of insertions, you are supposed to output the level-order traversal sequence of the resulting AVL tree, and to tell if it is a complete binary tree.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (<= 20). Then N distinct integer keys are given in the next line. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, insert the keys one by one into an initially empty AVL tree. Then first print in a line the level-order traversal sequence of the resulting AVL tree. All the numbers in a line must be separated by a space, and there must be no extra space at the end of the line. Then in the next line, print "YES" if the tree is complete, or "NO" if not.

**Sample Input 1:**

5

88 70 61 63 65

**Sample Output 1:**

70 63 88 61 65

YES

**Sample Input 2:**

8

88 70 61 96 120 90 65 68

**Sample Output 2:**

88 65 96 61 70 90 120 68

NO

# 1124. Raffle for Weibo Followers (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

John got a full mark on PAT. He was so happy that he decided to hold a raffle（抽奖） for his followers on Weibo -- that is, he would select winners from every N followers who forwarded his post, and give away gifts. Now you are supposed to help him generate the list of winners.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives three positive integers M (<= 1000), N and S, being the total number of forwards, the skip number of winners, and the index of the first winner (the indices start from 1). Then M lines follow, each gives the nickname (a nonempty string of no more than 20 characters, with no white space or return) of a follower who has forwarded John's post.

Note: it is possible that someone would forward more than once, but no one can win more than once. Hence if the current candidate of a winner has won before, we must skip him/her and consider the next one.

**Output Specification:**

For each case, print the list of winners in the same order as in the input, each nickname occupies a line. If there is no winner yet, print "Keep going..." instead.

**Sample Input 1:**

9 3 2

Imgonnawin!

PickMe

PickMeMeMeee

LookHere

Imgonnawin!

TryAgainAgain

TryAgainAgain

Imgonnawin!

TryAgainAgain

**Sample Output 1:**

PickMe

Imgonnawin!

TryAgainAgain

**Sample Input 2:**

2 3 5

Imgonnawin!

PickMe

**Sample Output 2:**

Keep going...

# 1125. Chain the Ropes (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given some segments of rope, you are supposed to chain them into one rope. Each time you may only fold two segments into loops and chain them into one piece, as shown by the figure. The resulting chain will be treated as another segment of rope and can be folded again. After each chaining, the lengths of the original two segments will be halved.



Your job is to make the longest possible rope out of N given segments.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (2 <= N <= 104). Then N positive integer lengths of the segments are given in the next line, separated by spaces. All the integers are no more than 104.

**Output Specification:**

For each case, print in a line the length of the longest possible rope that can be made by the given segments. The result must be rounded to the nearest integer that is no greater than the maximum length.

**Sample Input:**

8

10 15 12 3 4 13 1 15

**Sample Output:**

14

# 1126. Eulerian Path (25)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

In graph theory, an Eulerian path is a path in a graph which visits every edge exactly once. Similarly, an Eulerian circuit is an Eulerian path which starts and ends on the same vertex. They were first discussed by Leonhard Euler while solving the famous Seven Bridges of Konigsberg problem in 1736. It has been proven that connected graphs with all vertices of even degree have an Eulerian circuit, and such graphs are called **Eulerian**. If there are exactly two vertices of odd degree, all Eulerian paths start at one of them and end at the other. A graph that has an Eulerian path but not an Eulerian circuit is called **semi-Eulerian**. (Cited from https://en.wikipedia.org/wiki/Eulerian\_path)

Given an undirected graph, you are supposed to tell if it is Eulerian, semi-Eulerian, or non-Eulerian.

**Input Specification:**

Each input file contains one test case. Each case starts with a line containing 2 numbers N (<= 500), and M, which are the total number of vertices, and the number of edges, respectively. Then M lines follow, each describes an edge by giving the two ends of the edge (the vertices are numbered from 1 to N).

**Output Specification:**

For each test case, first print in a line the degrees of the vertices in ascending order of their indices. Then in the next line print your conclusion about the graph -- either "Eulerian", "Semi-Eulerian", or "Non-Eulerian". Note that all the numbers in the first line must be separated by exactly 1 space, and there must be no extra space at the beginning or the end of the line.

**Sample Input 1:**

7 12

5 7

1 2

1 3

2 3

2 4

3 4

5 2

7 6

6 3

4 5

6 4

5 6

**Sample Output 1:**

2 4 4 4 4 4 2

Eulerian

**Sample Input 2:**

6 10

1 2

1 3

2 3

2 4

3 4

5 2

6 3

4 5

6 4

5 6

**Sample Output 2:**

2 4 4 4 3 3

Semi-Eulerian

**Sample Input 3:**

5 8

1 2

2 5

5 4

4 1

1 3

3 2

3 4

5 3

**Sample Output 3:**

3 3 4 3 3

Non-Eulerian

# 1127. ZigZagging on a Tree (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

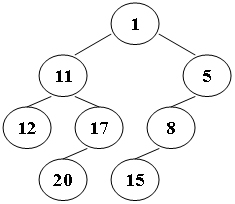
判题程序

Standard

作者

CHEN, Yue

Suppose that all the keys in a binary tree are distinct positive integers. A unique binary tree can be determined by a given pair of postorder and inorder traversal sequences. And it is a simple standard routine to print the numbers in level-order. However, if you think the problem is too simple, then you are too naive. This time you are supposed to print the numbers in "zigzagging order" -- that is, starting from the root, print the numbers level-by-level, alternating between left to right and right to left. For example, for the following tree you must output: 1 11 5 8 17 12 20 15.



**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<= 30), the total number of nodes in the binary tree. The second line gives the inorder sequence and the third line gives the postorder sequence. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print the zigzagging sequence of the tree in a line. All the numbers in a line must be separated by exactly one space, and there must be no extra space at the end of the line.

**Sample Input:**

8

12 11 20 17 1 15 8 5

12 20 17 11 15 8 5 1

**Sample Output:**

1 11 5 8 17 12 20 15

# 1128. N Queens Puzzle (20)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

The "eight queens puzzle" is the problem of placing eight chess queens on an 8×8 chessboard so that no two queens threaten each other. Thus, a solution requires that no two queens share the same row, column, or diagonal. The eight queens puzzle is an example of the more general N queens problem of placing N non-attacking queens on an N×N chessboard. (From Wikipedia - "Eight queens puzzle".)

Here you are NOT asked to solve the puzzles. Instead, you are supposed to judge whether or not a given configuration of the chessboard is a solution. To simplify the representation of a chessboard, let us assume that no two queens will be placed in the same column. Then a configuration can be represented by a simple integer sequence (Q1, Q2, ..., QN), where Qi is the row number of the queen in the i-th column. For example, Figure 1 can be represented by (4, 6, 8, 2, 7, 1, 3, 5) and it is indeed a solution to the 8 queens puzzle; while Figure 2 can be represented by (4, 6, 7, 2, 8, 1, 9, 5, 3) and is NOT a 9 queens' solution.

|  |  |  |
| --- | --- | --- |
| http://nos.patest.cn/nf_ol5ygnynwww.jpg |  | http://nos.patest.cn/nf_ol5yhbyve1t.jpg |
| Figure 1 |  | Figure 2 |

**Input Specification:**

Each input file contains several test cases. The first line gives an integer K (1 < K <= 200). Then K lines follow, each gives a configuration in the format "N Q1 Q2 ... QN", where 4 <= N <= 1000 and it is guaranteed that 1 <= Qi <= N for all i=1, ..., N. The numbers are separated by spaces.

**Output Specification:**

For each configuration, if it is a solution to the N queens problem, print "YES" in a line; or "NO" if not.

**Sample Input:**

4

8 4 6 8 2 7 1 3 5

9 4 6 7 2 8 1 9 5 3

6 1 5 2 6 4 3

5 1 3 5 2 4

**Sample Output:**

YES

NO

NO

YES

# 1129. Recommendation System (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Recommendation system predicts the preference that a user would give to an item. Now you are asked to program a very simple recommendation system that rates the user's preference by the number of times that an item has been accessed by this user.

**Input Specification:**

Each input file contains one test case. For each test case, the first line contains two positive integers: N (<= 50000), the total number of queries, and K (<= 10), the maximum number of recommendations the system must show to the user. Then given in the second line are the indices of items that the user is accessing -- for the sake of simplicity, all the items are indexed from 1 to N. All the numbers in a line are separated by a space.

**Output Specification:**

For each case, process the queries one by one. Output the recommendations for each query in a line in the format:

query: rec[1] rec[2] ... rec[K]

where **query** is the item that the user is accessing, and **rec[i]** (i = 1, ... K) is the i-th item that the system recommends to the user. The first K items that have been accessed most frequently are supposed to be recommended in non-increasing order of their frequencies. If there is a tie, the items will be ordered by their indices in increasing order.

Note: there is no output for the first item since it is impossible to give any recommendation at the time. It is guaranteed to have the output for at least one query.

**Sample Input:**

12 3

3 5 7 5 5 3 2 1 8 3 8 12

**Sample Output:**

5: 3

7: 3 5

5: 3 5 7

5: 5 3 7

3: 5 3 7

2: 5 3 7

1: 5 3 2

8: 5 3 1

3: 5 3 1

8: 3 5 1

12: 3 5 8

# 1130. Infix Expression (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a syntax tree (binary), you are supposed to output the corresponding infix expression, with parentheses reflecting the precedences of the operators.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N ( <= 20 ) which is the total number of nodes in the syntax tree. Then N lines follow, each gives the information of a node (the i-th line corresponds to the i-th node) in the format:

**data left\_child right\_child**

where **data** is a string of no more than 10 characters, **left\_child** and **right\_child** are the indices of this node's left and right children, respectively. The nodes are indexed from 1 to N. The NULL link is represented by -1. The figures 1 and 2 correspond to the samples 1 and 2, respectively.

|  |  |
| --- | --- |
| http://nos.patest.cn/nh_ol5yyz6cr02.JPG | http://nos.patest.cn/nh_ol5yyackpi3.JPG |
| Figure 1 | Figure 2 |

**Output Specification:**

For each case, print in a line the infix expression, with parentheses reflecting the precedences of the operators. Note that there must be no extra parentheses for the final expression, as is shown by the samples. There must be no space between any symbols.

**Sample Input 1:**

8

\* 8 7

a -1 -1

\* 4 1

+ 2 5

b -1 -1

d -1 -1

- -1 6

c -1 -1

**Sample Output 1:**

(a+b)\*(c\*(-d))

**Sample Input 2:**

8

2.35 -1 -1

\* 6 1

- -1 4

% 7 8

+ 2 3

a -1 -1

str -1 -1

871 -1 -1

**Sample Output 2:**

(a\*2.35)+(-(str%871))

# 1131. Subway Map (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

In the big cities, the subway systems always look so complex to the visitors. To give you some sense, the following figure shows the map of Beijing subway. Now you are supposed to help people with your computer skills! Given the starting position of your user, your task is to find the quickest way to his/her destination.



**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (< =100), the number of subway lines. Then N lines follow, with the i-th (i = 1, ..., N) line describes the i-th subway line in the format:

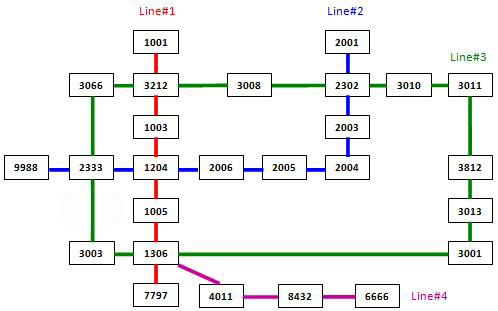
M S[1] S[2] ... S[M]

where **M** (<= 100) is the number of stops, and **S[i]**'s (i = 1, ... M) are the indices of the stations (the indices are 4-digit numbers from 0000 to 9999) along the line. It is guaranteed that the stations are given in the correct order -- that is, the train travels between **S[i]** and **S[i+1]** (i = 1, ..., M-1) without any stop.

Note: It is possible to have loops, but not self-loop (no train starts from S and stops at S without passing through another station). Each station interval belongs to a unique subway line. Although the lines may cross each other at some stations (so called "transfer stations"), no station can be the conjunction of more than 5 lines.

After the description of the subway, another positive integer K (<= 10) is given. Then K lines follow, each gives a query from your user: the two indices as the starting station and the destination, respectively.

The following figure shows the sample map.



Note: It is guaranteed that all the stations are reachable, and all the queries consist of legal station numbers.

**Output Specification:**

For each query, first print in a line the minimum number of stops. Then you are supposed to show the optimal path in a friendly format as the following:

Take Line#X1 from S1 to S2.

Take Line#X2 from S2 to S3.

......

where **Xi**'s are the line numbers and **Si**'s are the station indices. Note: Besides the starting and ending stations, only the transfer stations shall be printed.

If the quickest path is not unique, output the one with the minimum number of transfers, which is guaranteed to be unique.

**Sample Input:**

4

7 1001 3212 1003 1204 1005 1306 7797

9 9988 2333 1204 2006 2005 2004 2003 2302 2001

13 3011 3812 3013 3001 1306 3003 2333 3066 3212 3008 2302 3010 3011

4 6666 8432 4011 1306

3

3011 3013

6666 2001

2004 3001

**Sample Output:**

2

Take Line#3 from 3011 to 3013.

10

Take Line#4 from 6666 to 1306.

Take Line#3 from 1306 to 2302.

Take Line#2 from 2302 to 2001.

6

Take Line#2 from 2004 to 1204.

Take Line#1 from 1204 to 1306.

Take Line#3 from 1306 to 3001.

# 1132. Cut Integer (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Cutting an integer means to cut a K digits long integer Z into two integers of (K/2) digits long integers A and B. For example, after cutting Z = 167334, we have A = 167 and B = 334. It is interesting to see that Z can be devided by the product of A and B, as 167334 / (167 x 334) = 3. Given an integer Z, you are supposed to test if it is such an integer.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<= 20). Then N lines follow, each gives an integer Z (10<=Z<=231). It is guaranteed that the number of digits of Z is an even number.

**Output Specification:**

For each case, print a single line "Yes" if it is such a number, or "No" if not.

**Sample Input:**

3

167334

2333

12345678

**Sample Output:**

Yes

No

No

# 1133. Splitting A Linked List (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a singly linked list, you are supposed to rearrange its elements so that all the negative values appear before all of the non-negatives, and all the values in [0, K] appear before all those greater than K. The order of the elements inside each class must not be changed. For example, given the list being 18→7→-4→0→5→-6→10→11→-2 and K being 10, you must output -4→-6→-2→7→0→5→10→18→11.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains the address of the first node, a positive N (<= 105) which is the total number of nodes, and a positive K (<=1000). The address of a node is a 5-digit nonnegative integer, and NULL is represented by -1.

Then N lines follow, each describes a node in the format:

*Address Data Next*

where *Address* is the position of the node, *Data* is an integer in [-105, 105], and *Next* is the position of the next node. It is guaranteed that the list is not empty.

**Output Specification:**

For each case, output in order (from beginning to the end of the list) the resulting linked list. Each node occupies a line, and is printed in the same format as in the input.

**Sample Input:**

00100 9 10

23333 10 27777

00000 0 99999

00100 18 12309

68237 -6 23333

33218 -4 00000

48652 -2 -1

99999 5 68237

27777 11 48652

12309 7 33218

**Sample Output:**

33218 -4 68237

68237 -6 48652

48652 -2 12309

12309 7 00000

00000 0 99999

99999 5 23333

23333 10 00100

00100 18 27777

27777 11 -1

# 1134. Vertex Cover (25)

时间限制

600 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A **vertex cover** of a graph is a set of vertices such that each edge of the graph is incident to at least one vertex of the set. Now given a graph with several vertex sets, you are supposed to tell if each of them is a vertex cover or not.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives two positive integers N and M (both no more than 104), being the total numbers of vertices and the edges, respectively. Then M lines follow, each describes an edge by giving the indices (from 0 to N-1) of the two ends of the edge.

After the graph, a positive integer K (<= 100) is given, which is the number of queries. Then K lines of queries follow, each in the format:

Nv v[1] v[2] ... v[Nv]

where **Nv** is the number of vertices in the set, and **v[i]**'s are the indices of the vertices.

**Output Specification:**

For each query, print in a line "Yes" if the set is a vertex cover, or "No" if not.

**Sample Input:**

10 11

8 7

6 8

4 5

8 4

8 1

1 2

1 4

9 8

9 1

1 0

2 4

5

4 0 3 8 4

6 6 1 7 5 4 9

3 1 8 4

2 2 8

7 9 8 7 6 5 4 2

**Sample Output:**

No

Yes

Yes

No

No

# 1135. Is It A Red-Black Tree (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

There is a kind of balanced binary search tree named **red-black tree** in the data structure. It has the following 5 properties:

(1) Every node is either red or black.  
(2) The root is black.  
(3) Every leaf (NULL) is black.  
(4) If a node is red, then both its children are black.  
(5) For each node, all simple paths from the node to descendant leaves contain the same number of black nodes.

For example, the tree in Figure 1 is a red-black tree, while the ones in Figure 2 and 3 are not.

|  |  |  |
| --- | --- | --- |
| http://nos.patest.cn/pc_ovw0hp2a4xp.jpg | http://nos.patest.cn/pc_ovw0huh8oas.jpg | http://nos.patest.cn/pc_ovw0jb719kh.jpg |
| Figure 1 | Figure 2 | Figure 3 |

For each given binary search tree, you are supposed to tell if it is a legal red-black tree.

**Input Specification:**

Each input file contains several test cases. The first line gives a positive integer K (<=30) which is the total number of cases. For each case, the first line gives a positive integer N (<=30), the total number of nodes in the binary tree. The second line gives the preorder traversal sequence of the tree. While all the keys in a tree are positive integers, we use negative signs to represent red nodes. All the numbers in a line are separated by a space. The sample input cases correspond to the trees shown in Figure 1, 2 and 3.

**Output Specification:**

For each test case, print in a line "Yes" if the given tree is a red-black tree, or "No" if not.

**Sample Input:**

3

9

7 -2 1 5 -4 -11 8 14 -15

9

11 -2 1 -7 5 -4 8 14 -15

8

10 -7 5 -6 8 15 -11 17

**Sample Output:**

Yes

No

No

# 1136. A Delayed Palindrome (20)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Consider a positive integer N written in standard notation with k+1 digits ai as ak...a1a0 with 0 <= ai < 10 for all i and ak > 0. Then N is **palindromic** if and only if ai = ak-i for all i. Zero is written 0 and is also palindromic by definition.

Non-palindromic numbers can be paired with palindromic ones via a series of operations. First, the non-palindromic number is reversed and the result is added to the original number. If the result is not a palindromic number, this is repeated until it gives a palindromic number. Such number is called **a delayed palindrome**. (Quoted from https://en.wikipedia.org/wiki/Palindromic\_number)

Given any positive integer, you are supposed to find its paired palindromic number.

**Input Specification:**

Each input file contains one test case which gives a positive integer no more than 1000 digits.

**Output Specification:**

For each test case, print line by line the process of finding the palindromic number. The format of each line is the following:

A + B = C

where A is the original number, B is the reversed A, and C is their sum. A starts being the input number, and this process ends until C becomes a palindromic number -- in this case we print in the last line "C is a palindromic number."; or if a palindromic number cannot be found in 10 iterations, print "Not found in 10 iterations." instead.

**Sample Input 1:**

97152

**Sample Output 1:**

97152 + 25179 = 122331

122331 + 133221 = 255552

255552 is a palindromic number.

**Sample Input 2:**

196

**Sample Output 2:**

196 + 691 = 887

887 + 788 = 1675

1675 + 5761 = 7436

7436 + 6347 = 13783

13783 + 38731 = 52514

52514 + 41525 = 94039

94039 + 93049 = 187088

187088 + 880781 = 1067869

1067869 + 9687601 = 10755470

10755470 + 07455701 = 18211171

Not found in 10 iterations.

# 1137. Final Grading (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

For a student taking the online course "Data Structures" on China University MOOC (http://www.icourse163.org/), to be qualified for a certificate, he/she must first obtain no less than 200 points from the online programming assignments, and then receive a final grade no less than 60 out of 100. The final grade is calculated by G = (Gmid-termx 40% + Gfinalx 60%) if Gmid-term > Gfinal, or Gfinal will be taken as the final grade G. Here Gmid-term and Gfinalare the student's scores of the mid-term and the final exams, respectively.

The problem is that different exams have different grading sheets. Your job is to write a program to merge all the grading sheets into one.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives three positive integers: P , the number of students having done the online programming assignments; M, the number of students on the mid-term list; and N, the number of students on the final exam list. All the numbers are no more than 10,000.

Then three blocks follow. The first block contains P online programming scores Gp's; the second one contains M mid-term scores Gmid-term's; and the last one contains N final exam scores Gfinal's. Each score occupies a line with the format: StudentID Score, where StudentID is a string of no more than 20 English letters and digits, and Score is a nonnegative integer (the maximum score of the online programming is 900, and that of the mid-term and final exams is 100).

**Output Specification:**

For each case, print the list of students who are qualified for certificates. Each student occupies a line with the format:

StudentID Gp Gmid-term Gfinal G

If some score does not exist, output "-1" instead. The output must be sorted in descending order of their final grades (G must be rounded up to an integer). If there is a tie, output in ascending order of their StudentID's. It is guaranteed that the StudentID's are all distinct, and there is at least one qualified student.

**Sample Input:**

6 6 7

01234 880

a1903 199

ydjh2 200

wehu8 300

dx86w 220

missing 400

ydhfu77 99

wehu8 55

ydjh2 98

dx86w 88

a1903 86

01234 39

ydhfu77 88

a1903 66

01234 58

wehu8 84

ydjh2 82

missing 99

dx86w 81

**Sample Output:**

missing 400 -1 99 99

ydjh2 200 98 82 88

dx86w 220 88 81 84

wehu8 300 55 84 84

# 1138. Postorder Traversal (25)

时间限制

600 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Suppose that all the keys in a binary tree are distinct positive integers. Given the preorder and inorder traversal sequences, you are supposed to output the first number of the postorder traversal sequence of the corresponding binary tree.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives a positive integer N (<=50000), the total number of nodes in the binary tree. The second line gives the preorder sequence and the third line gives the inorder sequence. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print in one line the first number of the postorder traversal sequence of the corresponding binary tree.

**Sample Input:**

7

1 2 3 4 5 6 7

2 3 1 5 4 7 6

**Sample Output:**

3

# 1139. First Contact (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Unlike in nowadays, the way that boys and girls expressing their feelings of love was quite subtle in the early years. When a boy A had a crush on a girl B, he would usually not contact her directly in the first place. Instead, he might ask another boy C, one of his close friends, to ask another girl D, who was a friend of both B and C, to send a message to B -- quite a long shot, isn't it? Girls would do analogously.

Here given a network of friendship relations, you are supposed to help a boy or a girl to list all their friends who can possibly help them making the first contact.

**Input Specification:**

Each input file contains one test case. For each case, the first line gives two positive integers N (1 < N <= 300) and M, being the total number of people and the number of friendship relations, respectively. Then M lines follow, each gives a pair of friends. Here a person is represented by a 4-digit ID. To tell their genders, we use a negative sign to represent girls.

After the relations, a positive integer K (<= 100) is given, which is the number of queries. Then K lines of queries follow, each gives a pair of lovers, separated by a space. It is assumed that the first one is having a crush on the second one.

**Output Specification:**

For each query, first print in a line the number of different pairs of friends they can find to help them, then in each line print the IDs of a pair of friends.

If the lovers A and B are of opposite genders, you must first print the friend of A who is of the same gender of A, then the friend of B, who is of the same gender of B. If they are of the same gender, then both friends must be in the same gender as theirs. It is guaranteed that each person has only one gender.

The friends must be printed in non-decreasing order of the first IDs, and for the same first ones, in increasing order of the seconds ones.

**Sample Input:**

10 18

-2001 1001

-2002 -2001

1004 1001

-2004 -2001

-2003 1005

1005 -2001

1001 -2003

1002 1001

1002 -2004

-2004 1001

1003 -2002

-2003 1003

1004 -2002

-2001 -2003

1001 1003

1003 -2001

1002 -2001

-2002 -2003

5

1001 -2001

-2003 1001

1005 -2001

-2002 -2004

1111 -2003

**Sample Output:**

4

1002 2004

1003 2002

1003 2003

1004 2002

4

2001 1002

2001 1003

2002 1003

2002 1004

0

1

2003 2001

0

# 1140 Look-and-say Sequence（20分）

Look-and-say sequence is a sequence of integers as the following:

D, D1, D111, D113, D11231, D112213111, ...

where D is in [0, 9] except 1. The (n+1)st number is a kind of description of the nth number. For example, the 2nd number means that there is one D in the 1st number, and hence it is D1; the 2nd number consists of one D (corresponding to D1) and one 1 (corresponding to 11), therefore the 3rd number is D111; or since the 4th number is D113, it consists of one D, two 1's, and one 3, so the next number must be D11231. This definition works for D = 1 as well. Now you are supposed to calculate the Nth number in a look-and-say sequence of a given digit D.

### Input Specification:

Each input file contains one test case, which gives D (in [0, 9]) and a positive integer N (≤ 40), separated by a space.

### Output Specification:

Print in a line the Nth number in a look-and-say sequence of D.

### Sample Input:

1 8

### Sample Output:

1123123111

# 1141 PAT Ranking of Institutions（25 分）

After each PAT, the PAT Center will announce the ranking of institutions based on their students' performances. Now you are asked to generate the ranklist.

### Input Specification:

Each input file contains one test case. For each case, the first line gives a positive integer N (≤10​5​​), which is the number of testees. Then N lines follow, each gives the information of a testee in the following format:

ID Score School

where ID is a string of 6 characters with the first one representing the test level: B stands for the basic level, A the advanced level and T the top level; Score is an integer in [0, 100]; and School is the institution code which is a string of no more than 6 English letters (case insensitive). Note: it is guaranteed that ID is unique for each testee.

### Output Specification:

For each case, first print in a line the total number of institutions. Then output the ranklist of institutions in nondecreasing order of their ranks in the following format:

Rank School TWS Ns

where Rank is the rank (start from 1) of the institution; School is the institution code (all in lower case); ; TWS is the **total weighted score** which is defined to be the integer part of ScoreB/1.5 + ScoreA + ScoreT\*1.5, where ScoreX is the total score of the testees belong to this institution on level X; and Ns is the total number of testees who belong to this institution.

The institutions are ranked according to their TWS. If there is a tie, the institutions are supposed to have the same rank, and they shall be printed in ascending order of Ns. If there is still a tie, they shall be printed in alphabetical order of their codes.

### Sample Input:

10

A57908 85 Au

B57908 54 LanX

A37487 60 au

T28374 67 CMU

T32486 24 hypu

A66734 92 cmu

B76378 71 AU

A47780 45 lanx

A72809 100 pku

A03274 45 hypu

### Sample Output:

5

1 cmu 192 2

1 au 192 3

3 pku 100 1

4 hypu 81 2

4 lanx 81 2

注意：对于每个学校的总分是要先把每个学校的各级总分算出来，再按比例合成

# 1142 Maximal Clique（25 分）

A **clique** is a subset of vertices of an undirected graph such that every two distinct vertices in the clique are adjacent. A **maximal clique**is a clique that cannot be extended by including one more adjacent vertex. (Quoted from <https://en.wikipedia.org/wiki/Clique_(graph_theory>))

Now it is your job to judge if a given subset of vertices can form a maximal clique.

### Input Specification:

Each input file contains one test case. For each case, the first line gives two positive integers Nv (≤ 200), the number of vertices in the graph, and Ne, the number of undirected edges. Then Ne lines follow, each gives a pair of vertices of an edge. The vertices are numbered from 1 to Nv.

After the graph, there is another positive integer M (≤ 100). Then M lines of query follow, each first gives a positive number K (≤ Nv), then followed by a sequence of K distinct vertices. All the numbers in a line are separated by a space.

### Output Specification:

For each of the M queries, print in a line Yes if the given subset of vertices can form a maximal clique; or if it is a clique but not a **maximal clique**, print Not Maximal; or if it is not a clique at all, print Not a Clique.

### Sample Input:

8 10

5 6

7 8

6 4

3 6

4 5

2 3

8 2

2 7

5 3

3 4

6

4 5 4 3 6

3 2 8 7

2 2 3

1 1

3 4 3 6

3 3 2 1

### Sample Output:

Yes

Yes

Yes

Yes

Not Maximal

Not a Clique

# 1143 Lowest Common Ancestor（30 分）

The lowest common ancestor (LCA) of two nodes U and V in a tree is the deepest node that has both U and V as descendants.

A binary search tree (BST) is recursively defined as a binary tree which has the following properties:

* The left subtree of a node contains only nodes with keys less than the node's key.
* The right subtree of a node contains only nodes with keys greater than or equal to the node's key.
* Both the left and right subtrees must also be binary search trees.

Given any two nodes in a BST, you are supposed to find their LCA.

### Input Specification:

Each input file contains one test case. For each case, the first line gives two positive integers: M (≤ 1,000), the number of pairs of nodes to be tested; and N (≤ 10,000), the number of keys in the BST, respectively. In the second line, N distinct integers are given as the preorder traversal sequence of the BST. Then M lines follow, each contains a pair of integer keys U and V. All the keys are in the range of **int**.

### Output Specification:

For each given pair of U and V, print in a line LCA of U and V is A. if the LCA is found and A is the key. But if A is one of U and V, print X is an ancestor of Y. where X is A and Y is the other node. If U or V is not found in the BST, print in a line ERROR: U is not found. or ERROR: V is not found. or ERROR: U and V are not found..

### Sample Input:

6 8

6 3 1 2 5 4 8 7

2 5

8 7

1 9

12 -3

0 8

99 99

### Sample Output:

LCA of 2 and 5 is 3.

8 is an ancestor of 7.

ERROR: 9 is not found.

ERROR: 12 and -3 are not found.

ERROR: 0 is not found.

ERROR: 99 and 99 are not found.

思路：在建树的时候用set记录每个结点，对于每一组u、v先判断是否在树中。然后再从根结点x开始查找，如果x=u则u是v的祖先，如果x介于u、v之间则LCA为x，如果u、v都大于x则去x的右子树中递归查找。

# 1144 The Missing Number（20 分）

Given N integers, you are supposed to find the smallest positive integer that is NOT in the given list.

### Input Specification:

Each input file contains one test case. For each case, the first line gives a positive integer N (≤10​5​​). Then N integers are given in the next line, separated by spaces. All the numbers are in the range of **int**.

### Output Specification:

Print in a line the smallest positive integer that is missing from the input list.

### Sample Input:

10

5 -25 9 6 1 3 4 2 5 17

### Sample Output:

7

作者: CHEN, Yue

单位: 浙江大学

时间限制: 150ms

内存限制: 64MB

代码长度限制: 16KB

# 1145 Hashing - Average Search Time（25 分）

The task of this problem is simple: insert a sequence of distinct positive integers into a hash table first. Then try to find another sequence of integer keys from the table and output the average search time (the number of comparisons made to find whether or not the key is in the table). The hash function is defined to be *H*(*key*)=*key*%*TSize* where *TSize* is the maximum size of the hash table. Quadratic probing (with positive increments only) is used to solve the collisions.

Note that the table size is better to be prime. If the maximum size given by the user is not prime, you must re-define the table size to be the smallest prime number which is larger than the size given by the user.

### Input Specification:

Each input file contains one test case. For each case, the first line contains 3 positive numbers: MSize, N, and M, which are the user-defined table size, the number of input numbers, and the number of keys to be found, respectively. All the three numbers are no more than 10​4​​. Then N distinct positive integers are given in the next line, followed by M positive integer keys in the next line. All the numbers in a line are separated by a space and are no more than 10​5​​.

### Output Specification:

For each test case, in case it is impossible to insert some number, print in a line X cannot be inserted. where X is the input number. Finally print in a line the average search time for all the M keys, accurate up to 1 decimal place.

### Sample Input:

4 5 4

10 6 4 15 11

11 4 15 2

### Sample Output:

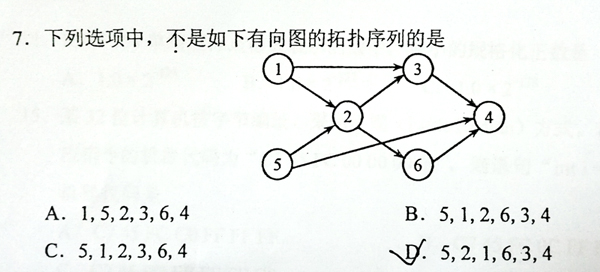
15 cannot be inserted.

2.8

思路：对于平方探测是指探测位置p=h+i\*i，i一直从0增加到maxSize，由于只有正向探测，故查找失败共探测了maxSize+1次

# 1146 Topological Order（25 分）

This is a problem given in the Graduate Entrance Exam in 2018: Which of the following is NOT a topological order obtained from the given directed graph? Now you are supposed to write a program to test each of the options.



### Input Specification:

Each input file contains one test case. For each case, the first line gives two positive integers N (≤ 1,000), the number of vertices in the graph, and M (≤ 10,000), the number of directed edges. Then M lines follow, each gives the start and the end vertices of an edge. The vertices are numbered from 1 to N. After the graph, there is another positive integer K (≤ 100). Then K lines of query follow, each gives a permutation of all the vertices. All the numbers in a line are separated by a space.

### Output Specification:

Print in a line all the indices of queries which correspond to "NOT a topological order". The indices start from zero. All the numbers are separated by a space, and there must no extra space at the beginning or the end of the line. It is graranteed that there is at least one answer.

### Sample Input:

6 8

1 2

1 3

5 2

5 4

2 3

2 6

3 4

6 4

5

1 5 2 3 6 4

5 1 2 6 3 4

5 1 2 3 6 4

5 2 1 6 3 4

1 2 3 4 5 6

### Sample Output:

3 4

# 1147 Heaps（30 分）

In computer science, a **heap** is a specialized tree-based data structure that satisfies the heap property: if P is a parent node of C, then the key (the value) of P is either greater than or equal to (in a max heap) or less than or equal to (in a min heap) the key of C. A common implementation of a heap is the binary heap, in which the tree is a complete binary tree. (Quoted from Wikipedia at <https://en.wikipedia.org/wiki/Heap_(data_structure>))

Your job is to tell if a given complete binary tree is a heap.

### Input Specification:

Each input file contains one test case. For each case, the first line gives two positive integers: M (≤ 100), the number of trees to be tested; and N (1 < N ≤ 1,000), the number of keys in each tree, respectively. Then M lines follow, each contains N distinct integer keys (all in the range of **int**), which gives the level order traversal sequence of a complete binary tree.

### Output Specification:

For each given tree, print in a line Max Heap if it is a max heap, or Min Heap for a min heap, or Not Heap if it is not a heap at all. Then in the next line print the tree's postorder traversal sequence. All the numbers are separated by a space, and there must no extra space at the beginning or the end of the line.

### Sample Input:

3 8

98 72 86 60 65 12 23 50

8 38 25 58 52 82 70 60

10 28 15 12 34 9 8 56

### Sample Output:

Max Heap

50 60 65 72 12 23 86 98

Min Heap

60 58 52 38 82 70 25 8

Not Heap

56 12 34 28 9 8 15 10

# 1148 Werewolf - Simple Version（20 分）

Werewolf（狼人杀） is a game in which the players are partitioned into two parties: the werewolves and the human beings. Suppose that in a game,

* player #1 said: "Player #2 is a werewolf.";
* player #2 said: "Player #3 is a human.";
* player #3 said: "Player #4 is a werewolf.";
* player #4 said: "Player #5 is a human."; and
* player #5 said: "Player #4 is a human.".

Given that there were 2 werewolves among them, at least one but not all the werewolves were lying, and there were exactly 2 liers. Can you point out the werewolves?

Now you are asked to solve a harder version of this problem: given that there were *N* players, with 2 werewolves among them, at least one but not all the werewolves were lying, and there were exactly 2 liers. You are supposed to point out the werewolves.

### Input Specification:

Each input file contains one test case. For each case, the first line gives a positive integer *N* (5≤*N*≤100). Then *N* lines follow and the *i*-th line gives the statement of the *i*-th player (1≤*i*≤*N*), which is represented by the index of the player with a positive sign for a human and a negative sign for a werewolf.

### Output Specification:

If a solution exists, print in a line in ascending order the indices of the two werewolves. The numbers must be separated by exactly one space with no extra spaces at the beginning or the end of the line. If there are more than one solution, you must output the smallest solution sequence -- that is, for two sequences *A*=*a*[1],...,*a*[*M*] and *B*=*b*[1],...,*b*[*M*], if there exists 0≤*k*<*M* such that *a*[*i*]=*b*[*i*] (*i*≤*k*) and *a*[*k*+1]<*b*[*k*+1], then *A* is said to be smaller than *B*. In case there is no solution, simply print No Solution.

### Sample Input 1:

5

-2

+3

-4

+5

+4

### Sample Output 1:

1 4

### Sample Input 2:

6

+6

+3

+1

-5

-2

+4

### Sample Output 2 (the solution is not unique):

1 5

### Sample Input 3:

5

-2

-3

-4

-5

-1

### Sample Output 3:

No Solution

# 1149 Dangerous Goods Packaging（25 分）

When shipping goods with containers, we have to be careful not to pack some incompatible goods into the same container, or we might get ourselves in serious trouble. For example, oxidizing agent （氧化剂） must not be packed with flammable liquid （易燃液体）, or it can cause explosion.

Now you are given a long list of incompatible goods, and several lists of goods to be shipped. You are supposed to tell if all the goods in a list can be packed into the same container.

### Input Specification:

Each input file contains one test case. For each case, the first line gives two positive integers: *N* (≤10​4​​), the number of pairs of incompatible goods, and *M* (≤100), the number of lists of goods to be shipped.

Then two blocks follow. The first block contains N pairs of incompatible goods, each pair occupies a line; and the second one contains M lists of goods to be shipped, each list occupies a line in the following format:

K G[1] G[2] ... G[K]

where K (≤1,000) is the number of goods and G[i]'s are the IDs of the goods. To make it simple, each good is represented by a 5-digit ID number. All the numbers in a line are separated by spaces.

### Output Specification:

For each shipping list, print in a line Yes if there are no incompatible goods in the list, or No if not.

### Sample Input:

6 3

20001 20002

20003 20004

20005 20006

20003 20001

20005 20004

20004 20006

4 00001 20004 00002 20003

5 98823 20002 20003 20006 10010

3 12345 67890 23333

### Sample Output:

No

Yes

Yes

# 1150 Travelling Salesman Problem（25 分）

The "travelling salesman problem" asks the following question: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?" It is an NP-hard problem in combinatorial optimization, important in operations research and theoretical computer science. (Quoted from "<https://en.wikipedia.org/wiki/Travelling_salesman_problem>".)

In this problem, you are supposed to find, from a given list of cycles, the one that is the closest to the solution of a travelling salesman problem.

### Input Specification:

Each input file contains one test case. For each case, the first line contains 2 positive integers *N* (2<*N*≤200), the number of cities, and *M*, the number of edges in an undirected graph. Then *M* lines follow, each describes an edge in the format City1 City2 Dist, where the cities are numbered from 1 to *N* and the distance Dist is positive and is no more than 100. The next line gives a positive integer *K* which is the number of paths, followed by *K* lines of paths, each in the format:

*n* *C*​1​​ *C*​2​​ ... *C*​*n*​​

where *n* is the number of cities in the list, and *C*​*i*​​'s are the cities on a path.

### Output Specification:

For each path, print in a line Path X: TotalDist (Description) where X is the index (starting from 1) of that path, TotalDist its total distance (if this distance does not exist, output NA instead), and Description is one of the following:

* TS simple cycle if it is a simple cycle that visits every city;
* TS cycle if it is a cycle that visits every city, but not a simple cycle;
* Not a TS cycle if it is NOT a cycle that visits every city.

Finally print in a line Shortest Dist(X) = TotalDist where X is the index of the cycle that is the closest to the solution of a travelling salesman problem, and TotalDist is its total distance. It is guaranteed that such a solution is unique.

### Sample Input:

6 10

6 2 1

3 4 1

1 5 1

2 5 1

3 1 8

4 1 6

1 6 1

6 3 1

1 2 1

4 5 1

7

7 5 1 4 3 6 2 5

7 6 1 3 4 5 2 6

6 5 1 4 3 6 2

9 6 2 1 6 3 4 5 2 6

4 1 2 5 1

7 6 1 2 5 4 3 1

7 6 3 2 5 4 1 6

### Sample Output:

Path 1: 11 (TS simple cycle)

Path 2: 13 (TS simple cycle)

Path 3: 10 (Not a TS cycle)

Path 4: 8 (TS cycle)

Path 5: 3 (Not a TS cycle)

Path 6: 13 (Not a TS cycle)

Path 7: NA (Not a TS cycle)

Shortest Dist(4) = 8

# 1151 LCA in a Binary Tree（30 分）

The lowest common ancestor (LCA) of two nodes U and V in a tree is the deepest node that has both U and V as descendants.

Given any two nodes in a binary tree, you are supposed to find their LCA.

### Input Specification:

Each input file contains one test case. For each case, the first line gives two positive integers: M (≤ 1,000), the number of pairs of nodes to be tested; and N (≤ 10,000), the number of keys in the binary tree, respectively. In each of the following two lines, N distinct integers are given as the inorder and preorder traversal sequences of the binary tree, respectively. It is guaranteed that the binary tree can be uniquely determined by the input sequences. Then M lines follow, each contains a pair of integer keys U and V. All the keys are in the range of **int**.

### Output Specification:

For each given pair of U and V, print in a line LCA of U and V is A. if the LCA is found and A is the key. But if A is one of U and V, print X is an ancestor of Y. where X is A and Y is the other node. If U or V is not found in the binary tree, print in a line ERROR: U is not found. or ERROR: V is not found. or ERROR: U and V are not found..

### Sample Input:

6 8

7 2 3 4 6 5 1 8

5 3 7 2 6 4 8 1

2 6

8 1

7 9

12 -3

0 8

99 99

### Sample Output:

LCA of 2 and 6 is 3.

8 is an ancestor of 1.

ERROR: 9 is not found.

ERROR: 12 and -3 are not found.

ERROR: 0 is not found.

ERROR: 99 and 99 are not found.

# 1152 Google Recruitment （20 分）

In July 2004, Google posted on a giant billboard along Highway 101 in Silicon Valley (shown in the picture below) for recruitment. The content is super-simple, a URL consisting of the first 10-digit prime found in consecutive digits of the natural constant *e*. The person who could find this prime number could go to the next step in Google's hiring process by visiting this website.



The natural constant *e* is a well known transcendental number（超越数）. The first several digits are: *e* = 2.71828182845904523536028747135266249775724709369995957496696762772407663035354759457138217852516642**7427466391**932003059921... where the 10 digits in bold are the answer to Google's question.

Now you are asked to solve a more general problem: find the first K-digit prime in consecutive digits of any given L-digit number.

### Input Specification:

Each input file contains one test case. Each case first gives in a line two positive integers: L (≤ 1,000) and K (< 10), which are the numbers of digits of the given number and the prime to be found, respectively. Then the L-digit number N is given in the next line.

### Output Specification:

For each test case, print in a line the first K-digit prime in consecutive digits of N. If such a number does not exist, output 404 instead. Note: the leading zeroes must also be counted as part of the K digits. For example, to find the 4-digit prime in 200236, 0023 is a solution. However the first digit 2 must not be treated as a solution 0002 since the leading zeroes are not in the original number.

### Sample Input 1:

20 5

23654987725541023819

### Sample Output 1:

49877

### Sample Input 2:

10 3

2468024680

### Sample Output 2:

404

# 1153 Decode Registration Card of PAT （25 分）

A registration card number of PAT consists of 4 parts:

* the 1st letter represents the test level, namely, T for the top level, A for advance and B for basic;
* the 2nd - 4th digits are the test site number, ranged from 101 to 999;
* the 5th - 10th digits give the test date, in the form of yymmdd;
* finally the 11th - 13th digits are the testee's number, ranged from 000 to 999.

Now given a set of registration card numbers and the scores of the card owners, you are supposed to output the various statistics according to the given queries.

### Input Specification:

Each input file contains one test case. For each case, the first line gives two positive integers *N* (≤10​4​​) and *M* (≤100), the numbers of cards and the queries, respectively.

Then *N* lines follow, each gives a card number and the owner's score (integer in [0,100]), separated by a space.

After the info of testees, there are *M* lines, each gives a query in the format Type Term, where

* Type being 1 means to output all the testees on a given level, in non-increasing order of their scores. The corresponding Termwill be the letter which specifies the level;
* Type being 2 means to output the total number of testees together with their total scores in a given site. The corresponding Term will then be the site number;
* Type being 3 means to output the total number of testees of every site for a given test date. The corresponding Term will then be the date, given in the same format as in the registration card.

### Output Specification:

For each query, first print in a line Case #: input, where # is the index of the query case, starting from 1; and input is a copy of the corresponding input query. Then output as requested:

* for a type 1 query, the output format is the same as in input, that is, CardNumber Score. If there is a tie of the scores, output in increasing alphabetical order of their card numbers (uniqueness of the card numbers is guaranteed);
* for a type 2 query, output in the format Nt Ns where Nt is the total number of testees and Ns is their total score;
* for a type 3 query, output in the format Site Nt where Site is the site number and Nt is the total number of testees at Site. The output must be in non-increasing order of Nt's, or in increasing order of site numbers if there is a tie of Nt.

If the result of a query is empty, simply print NA.

### Sample Input:

8 4

B123180908127 99

B102180908003 86

A112180318002 98

T107150310127 62

A107180908108 100

T123180908010 78

B112160918035 88

A107180908021 98

1 A

2 107

3 180908

2 999

### Sample Output:

Case 1: 1 A

A107180908108 100

A107180908021 98

A112180318002 98

Case 2: 2 107

3 260

Case 3: 3 180908

107 2

123 2

102 1

Case 4: 2 999

NA

# 1154 Vertex Coloring （25 分）

A **proper vertex coloring** is a labeling of the graph's vertices with colors such that no two vertices sharing the same edge have the same color. A coloring using at most *k* colors is called a (proper) ***k*-coloring**.

Now you are supposed to tell if a given coloring is a proper *k*-coloring.

### Input Specification:

Each input file contains one test case. For each case, the first line gives two positive integers *N* and *M* (both no more than 10​4​​), being the total numbers of vertices and edges, respectively. Then *M* lines follow, each describes an edge by giving the indices (from 0 to *N*−1) of the two ends of the edge.

After the graph, a positive integer *K* (≤ 100) is given, which is the number of colorings you are supposed to check. Then *K* lines follow, each contains *N* colors which are represented by non-negative integers in the range of **int**. The *i*-th color is the color of the *i*-th vertex.

### Output Specification:

For each coloring, print in a line k-coloring if it is a proper k-coloring for some positive k, or No if not.

### Sample Input:

10 11

8 7

6 8

4 5

8 4

8 1

1 2

1 4

9 8

9 1

1 0

2 4

4

0 1 0 1 4 1 0 1 3 0

0 1 0 1 4 1 0 1 0 0

8 1 0 1 4 1 0 5 3 0

1 2 3 4 5 6 7 8 8 9

### Sample Output:

4-coloring

No

6-coloring

No

# 1155 Heap Paths （30 分）

In computer science, a **heap** is a specialized tree-based data structure that satisfies the heap property: if P is a parent node of C, then the key (the value) of P is either greater than or equal to (in a max heap) or less than or equal to (in a min heap) the key of C. A common implementation of a heap is the binary heap, in which the tree is a complete binary tree. (Quoted from Wikipedia at <https://en.wikipedia.org/wiki/Heap_(data_structure>))

One thing for sure is that all the keys along any path from the root to a leaf in a max/min heap must be in non-increasing/non-decreasing order.

Your job is to check every path in a given complete binary tree, in order to tell if it is a heap or not.

### Input Specification:

Each input file contains one test case. For each case, the first line gives a positive integer *N* (1<*N*≤1,000), the number of keys in the tree. Then the next line contains *N* distinct integer keys (all in the range of **int**), which gives the level order traversal sequence of a complete binary tree.

### Output Specification:

For each given tree, first print all the paths from the root to the leaves. Each path occupies a line, with all the numbers separated by a space, and no extra space at the beginning or the end of the line. The paths must be printed in the following order: for each node in the tree, all the paths in its right subtree must be printed before those in its left subtree.

Finally print in a line Max Heap if it is a max heap, or Min Heap for a min heap, or Not Heap if it is not a heap at all.

### Sample Input 1:

8

98 72 86 60 65 12 23 50

### Sample Output 1:

98 86 23

98 86 12

98 72 65

98 72 60 50

Max Heap

### Sample Input 2:

8

8 38 25 58 52 82 70 60

### Sample Output 2:

8 25 70

8 25 82

8 38 52

8 38 58 60

Min Heap

### Sample Input 3:

8

10 28 15 12 34 9 8 56

### Sample Output 3:

10 15 8

10 15 9

10 28 34

10 28 12 56

Not Heap