acm International Collegiate Programming Contest

2012/7/6 Tokyo

Links

A B C D E F G

Problem A

ミレニアム

賢王が新しい暦を定めた．「明日を1年1月1日と定める．1年には1月から10月までの10ヶ月があり，*大の月*から始まる．ふつうの年は，1月が大の月，2月は*小の月*，3月は大の月，そして最後の10月は小の月というように，大の月と小の月が交互に訪れる．大の月の日数は20日，小の月の日数は19日である．しかし，3年，6年，9年，…のように，3で割り切れる年には，小の月は訪れず，10すべての月を大の月とする．」

幾年月が流れた．やがて1000年1月1日のミレニアム記念日の祝賀式において，誕生してからの日数が王立のくじ引きで選ばれた数と一致する国民に景品を与えることになった．国民らを手助けして，生まれた日からミレニアム記念日までの日数を計算するプログラムを奉ぜよ．

Input

入力全体は以下のような形式で表される．

n
$Y_1\ M_1\ D_1$
$Y_2\ M_2\ D_2$
...
$Y_n\ M_n\ D_n$

ここで，最初の行にはデータセット数を表す100以下の正整数nが与えられる．この行に続いてn個のデータセットが与えられる．各データセットはそれぞれ1行であり，そのなかに三つの正整数Y_i (< 1000)，M_i (≤ 10)，D_i (≤ 20)を含む．これらの数は空白で区切られ，それぞれ，人物が生まれた日の王暦の年，月，日である．

Output

各データセットに与えられる誕生日について，その誕生日からミレニアム記念日までの日数をそれぞれ1行に出力しなさい．ただし，この日数に誕生日は勘定し，ミレニアム記念日は含まないものとする．出力行はこの数値以外の文字を含んではならない．

Sample Input

```
8
1 1 1
344 3 1
696 5 1
182 9 5
998 8 7
344 2 19
696 4 19
999 10 20
```

Output for the Sample Input

```
196470
128976
59710
160715
252
128977
59712
1
```

(End of Problem A) A B C D E F G

Problem B

繰り返す10進数

数の10進表現は数字を並べ替えることで別の数になる． このことを使って数列を作ってみよう．

最初に，非負の整数a_0と桁数Lを与える． 以下の規則を適用してa_iからa_{i+1}を得る．

1. 整数a_iを，L桁の10進数で表記する． 必要であれば上位の桁に 0 を付け加える． たとえば 2012 という数は 6 桁の10進数で表記する場合は 002012 となる．
2. 各桁の数字を並べ替えて，最も大きい整数と最も小さい整数を作る． 上の例では，最も大きい整数は 221000 であり， 最も小さい整数は 000122 = 122となる．
3. 最も大きい整数から最も小さな整数を引いて，整数a_{i+1}を得る． 上の例では 221000 − 122 を計算して 220878 を得る．

この計算を繰り返すと，数列 a_0 , a_1 , a_2 , ... が得られる．

例えば 83268 という整数と桁数6が与えられた時， この計算を繰り返すと次のような数列 a_0 , a_1 ,

a_2 , ... が得られる．

$a_0 = 083268$
$a_1 = 886320 - 023688 = 862632$
$a_2 = 866322 - 223668 = 642654$
$a_3 = 665442 - 244566 = 420876$
$a_4 = 876420 - 024678 = 851742$

$a_5 = 875421 - 124578 = 750843$
$a_6 = 875430 - 034578 = 840852$
$a_7 = 885420 - 024588 = 860832$
$a_8 = 886320 - 023688 = 862632$
…

整数を表すための桁数が指定されているので，a_0 , a_1 , a_2 , ... と順に計算していくといずれ同じ数が現れる．すなわち，条件 $a_i = a_j$ を満たすような i と j (ただし $i > j$) の組が必ず存在する．上の例では，$a_8 = a_1 = 862632$ なので $(i = 8, j = 1)$ の組が条件を満たす．

最初の整数 a_0 と桁数 L が与えられた場合に，条件 $a_i = a_j$ (ただし $i > j$) を満たす最小の i を求めるプログラムを作成せよ．

Input

入力は複数のデータセットからなる．各データセットは2つの整数 a_0 と L が1個のスペースで区切られた1行であり，a_0 が最初の整数を，L が桁数を表す．ただし，$1 \leq L \leq 6$ かつ $0 \leq a_0 < 10^L$ である．

入力の終わりは2個の0を含む行で示される．この行はデータセットではない．

Output

各データセットに対して，条件 $a_i = a_j$ $(i > j)$ を満たす最小の i を求め，そのときの j の値，a_i の値，$i - j$ の値をスペース1個ずつで区切り，1行で出力せよ．数値を出力する時は，余分な上位のゼロは有ってはならない．出力に余分な文字は含んではならない．

上記の i は20を超えないと仮定して構わない．

Sample Input

```
2012 4
83268 6
1112 4
0 1
99 2
0 0
```

Output for the Sample Input

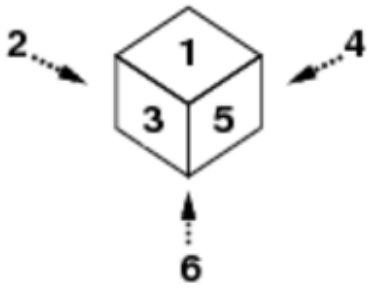
```
3 6174 1
1 862632 7
5 6174 1
0 0 1
1 0 1
```

(End of Problem B) A B C D E F G

Problem C

偏りのあるサイコロ

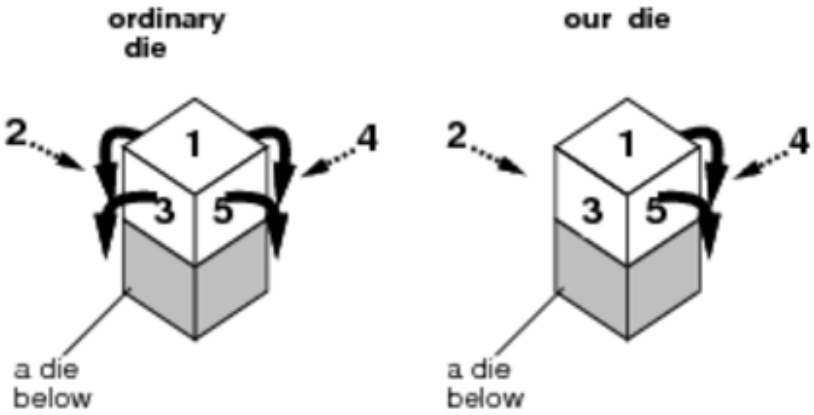
確率的アルゴリズムの大家であるランダム教授は現在，偏りのあるサイコロの実験を行っている．実験というのは，平面の上方の決まった位置から次々にサイコロを落とすものである．サイコロは平面またはすでにそこにあるサイコロの上に回転せずに落ち，さらに転がり落ちることもある．結果としては，積み上がった山の状態，具体的には上から見たときにどの目がいくつ見えるかを記録する．すべてのサイコロは同じ大きさで，数の振り方はすべて図C-1のようになっている．



図C-1: サイコロの面の配置

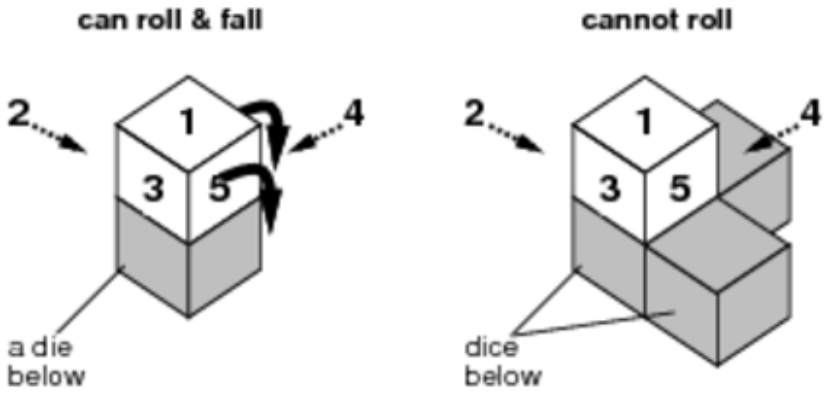
ここで用いるサイコロは非常に特殊な性質を持っている．

(1)通常のサイコロは4方向に転がることができるが，ここで用いるサイコロは4，5，6の面の方向にしか転がらない（1，2，3の面の方向には転がれない）．図C-2に示す状況では，ここで用いるサイコロ(our die)は2方向にしか転がらない．



図C-2: 通常のサイコロと偏りのあるサイコロ

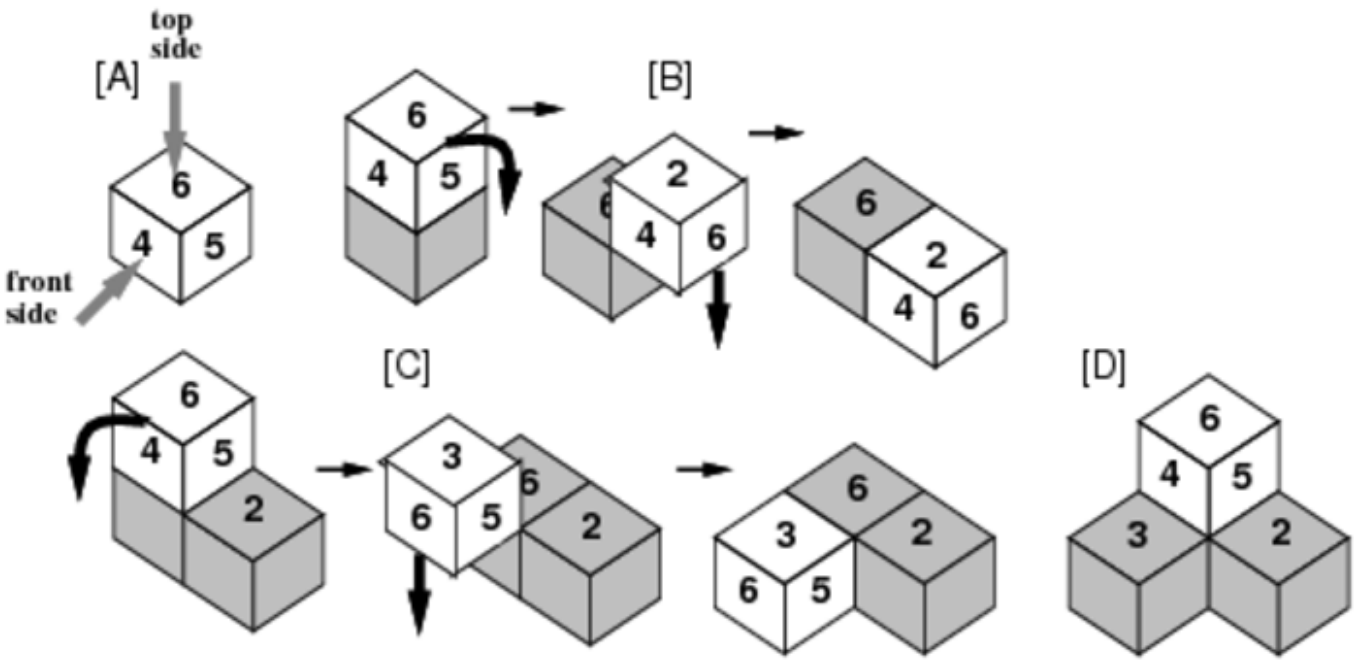
(2)サイコロが転がるのは，転がった後に落ちる時のみである．この様子を図C-3に示す．転がりうる方向が複数ある場合は，最も大きな目の面の方向に転がる．



図C-3: サイコロは落ちられる時だけ転がれる

(3)転がる場合には，図C-4の[B]，[C]のように，ちょうど90度回転した後，まっすぐ下に落ち，平面か他のサイコロの上に着地する．

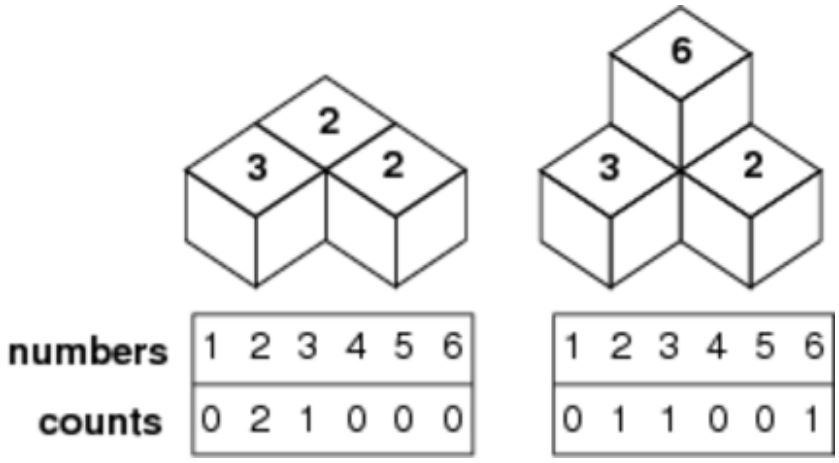
(4)サイコロは一度転がり落ちた後も，(1)〜(3)の規則に従い繰り返し転がり落ちる．



図C-4: 偏りのあるサイコロの積みあがり方の例

たとえば，同じ向き(上面が6，前面が4)で4つのサイコロを落下させた場合，図C-4のように山ができる．

こる.



図C-5: 記録の例

山が完成したら，1から6までの面が上からいくつ見えるかを数えて記録する．たとえば図C-5の左の場合は記録は「０２１０００」となり，右の場合は「０１１００１」となる.

Input

入力は複数のデータセットからなり，各データセットは次の形式で表される.

n
t_1 f_1
t_2 f_2
...
t_n f_n

ここでn ($1 \leq n \leq 100$) は整数であり，落とすサイコロの個数である．t_i とf_i ($1 \leq t_i, f_i \leq 6$) は1つの空白で区切られた整数であり，i番目のサイコロから手を放す瞬間の上面と前面の目である.

入力の終わりは1つのゼロだけからなる行によって示される.

Output

各データセットごとに，6個の整数を1個の空白文字で区切って出力せよ．それぞれの整数は1～6それぞれの目が上方向から何個見えるかを表す．出力にはこれら以外の文字があってはならない.

Sample Input

```
4
6 4
6 4
6 4
6 4
2
5 3
5 3
```

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

Output for the Sample Input

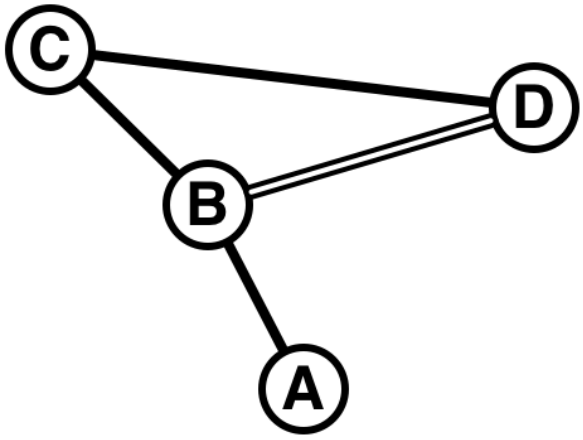
```
0 1 1 0 0 1
1 0 0 0 1 0
1 2 2 1 0 0
2 3 0 0 0 0
```

(End of Problem C) A B C D E F G

Problem D

鉄道乗り継ぎ

東京の鉄道ネットワークは非常に複雑である． たとえば，図D-1のような路線図が含まれている．



図D-1: 路線図の例

この路線図の上で，A駅からD駅に行きたいとする． 最短距離の経路を選ぶなら，A→B→Dと行けばよいことは明らかである． しかし，最短距離の経路が必ず最小コストの経路になるとは限らない．図で，A-B，B-C，C-Dが同じ鉄道会社の線で，B-Dだけ別の会社の線だったとする． この場合，A→B→C→Dの経路の方がA→B→Dの経路より安いことがありうる． その要因の一つは，運賃が距離に比例していないことである． 距離が長いほど，単位距離当たりの運賃は安くなるように設定されていることが多い． 複数の会社の線を乗り継ぐと，それぞれの会社の運賃を単純に合計するので， 単一の会社の線だけを使った経路に比べて，距離は短くとも， 運賃は割高になってしまうことがあるのだ．

この問題では，複数の鉄道会社からなる路線図が与えられる． また，それぞれの会社の運賃表（距離から運賃を計算するルール）が与えられる． 出発地と目的地が与えられたとき，最も安い運賃の経路を求めるプログラムを書くことがあなたの仕事である．

Input

入力は複数のデータセットから構成される． 各データセットの形式は次に示すとおりである．

> n m c s g
> x_1 y_1 d_1 c_1
>
> ...
> x_m y_m d_m c_m
> p_1 ... p_c
> $q_{1,1}$ ... q_{1,p_1-1}
> $r_{1,1}$ ... r_{1,p_1}
> ...
> $q_{c,1}$ ... q_{c,p_c-1}
> $r_{c,1}$ ... r_{c,p_c}

データセットの中の入力項目は，すべて非負の整数である． 行中の入力項目の区切りは空白1個である．

最初の行は，鉄道ネットワークの大きさと実現したい旅行を規定する． n は駅の数である($2 \leq n \leq 100$)． m は駅と駅をつなぐ路線の数である($0 \leq m \leq 10000$)． c は鉄道会社の数である($1 \leq c \leq 20$)． s は出発地の駅番号である($1 \leq s \leq n$)． g は目的地の駅番号である($1 \leq g \leq n, g \neq s$)．

続いて，m 本の路線のデータが順次与えられる． それぞれの行は，二つの駅x_i とy_i を結ぶ路線の記述である($1 \leq x_i \leq n, 1 \leq y_i \leq n, x_i \neq y_i$)． 各路線は，どちらの方向の移動にも使える． 同じ二つの駅を結ぶ路線が2本以上存在することもあり得る． d_i は，その路線の長さ（移動距離）である($1 \leq d_i \leq 200$)． c_i は，その路線を運営する鉄道会社の番号である($1 \leq c_i \leq c$)．

各鉄道会社の運賃表（距離と運賃の関係）は折れ線グラフで与えられる． 鉄道会社j 番に対して， 折れ線の区間の数（折れ目の数プラス 1）がp_j である ($1 \leq p_j \leq 50$)． $q_{j,k}$ ($1 \leq k \leq p_j-1$)が折れ目の位置を示す距離の値である ($1 \leq q_{j,k} \leq 10000$)． $r_{j,k}$ ($1 \leq k \leq p_j$)が該当する距離の範囲に対して距離 1 当たりの運賃の増分を与える ($1 \leq r_{j,k} \leq 100$)． より詳しく言えば，距離z のときの運賃を $f_j(z)$ と表すと， $q_{j,k-1}+1 \leq z \leq q_{j,k}$ を満たす距離z に対して $f_j(z) = f_j(z-1)+r_{j,k}$ である． ただし， $q_{j,0}$ と$f_j(0)$はゼロ， q_{j,p_j} は無限大と仮定する．

たとえば，p_j が 3 で， $q_{j,1} = 3$， $q_{j,2} = 6$， $r_{j,1} = 10$， $r_{j,2} = 5$， $r_{j,3} = 3$ だったとする． この場合の距離と運賃の対応関係は次のようになる．

距離	1	2	3	4	5	6	7	8	9
運賃	10	20	30	35	40	45	48	51	54

$q_{j,k}$ は，k に関して単調増加である． $r_{j,k}$ は，k に関して単調減少である．

最後のデータセットの直後に，空白で区切られた五つのゼロからなる行がある．

Output

入力の各データセットに対して，合計運賃が最小になるような経路を選んだときの所要運賃を答えなさい． ただし，出発地から目的地に到達できない場合は "-1" と出力すること． 出力行の中に，結果を表す文字以外のもの（たとえば空白）が含まれていてはならない．

出発地から目的地までの経路を定めたときの運賃の計算ルールは次のとおりとする． 同じ会社の複数の路線を連続して経由する場合は，これらの路線の合計距離に基づいて運賃を決める． 全体の運賃は，こうして決めた同一会社連続区間の運賃の総和である． 同じ会社の路線を複数利用していても，間にほかの会社の路線がはさまっている場合は，前後の区間の運賃を独立に計算する． 乗り継ぎ割引のような制度は存在しない．

Sample Input

```
4 4 2 1 4
1 2 2 1
2 3 2 1
3 4 5 1
2 4 4 2
3 1
3 6
10 5 3

10
2 0 1 1 2
1

1
4 5 2 4 1
4 3 10 1
3 2 2 1
3 2 1 2
3 2 5 2
2 1 10 1
3 3
20 30
3 2 1
5 10
3 2 1
5 5 2 1 5
1 2 10 2
1 3 20 2
2 4 20 1
3 4 10 1
4 5 20 1
2 2
20
4 1
20
3 1
0 0 0 0 0
```

Output for the Sample Input

Output for the Sample Input

```
54
-1
63

130
```

(End of Problem D) A B C D E F G

Problem E

鎖中経路

平面上の複数の円環から構成される鎖がある． 最初（最後）の円は，次（一つ前）の円だけと交差し， 中間の各円は隣の二つの円だけと交差する．

あなたの仕事は，以下の条件を満たす最短経路を見つけることである．

- 経路は，最初と最後の円の中心をつなぐ．
- 経路は鎖中にある．すなわち，経路上のすべての点は，少なくとも一つの円の内側もしくは円周上にある．

図 E-1 は，そのような鎖の例と対応する最短経路を示したものである．

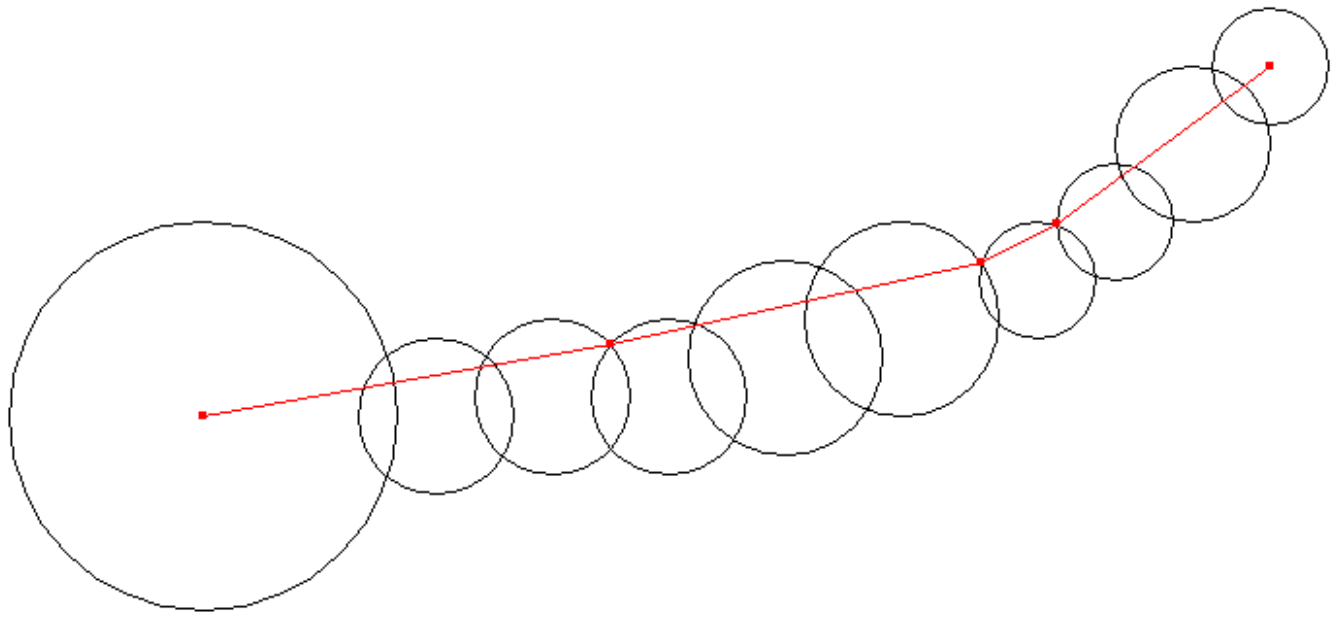


図 E-1: 鎖の例と対応する最短経路

Input

入力は複数のデータセットからなる． 各データセットは一つの鎖の形状を表し，その形式は以下のとおりである．

n

x_1 y_1 r_1
x_2 y_2 r_2
...
x_n y_n r_n

データセットの最初の行は，円の個数を表す整数 n ($3 \leq n \leq 100$) のみからなる． 続く n 行のそれぞれは，空白文字 1 個で区切られた 3 個の整数からなる． (x_i, y_i) と r_i は， i 番目の円 C_i の中心の位置と半径を表す． $0 \leq x_i \leq 1000$, $0 \leq y_i \leq 1000$, $1 \leq r_i \leq 25$ であると仮定してよい．

C_i と C_{i+1} ($1 \leq i \leq n-1$) は，離れた 2 点で交差すると仮定してよい． $j \geq i+2$ のとき，C_i と C_j は離れており，一方が他方を含むこともない． 加えて，それぞれの円は他の円の中心を内部に含むことがないと仮定してもよい．

入力の終わりは，ゼロを一つだけ含む行で表される．

なお，図 E-1 は，後に示す Sample Input 中の最初のデータセットに対応している． 図 E-2 は，Sample Input 中の引き続くデータセットに対する最短経路を示している．

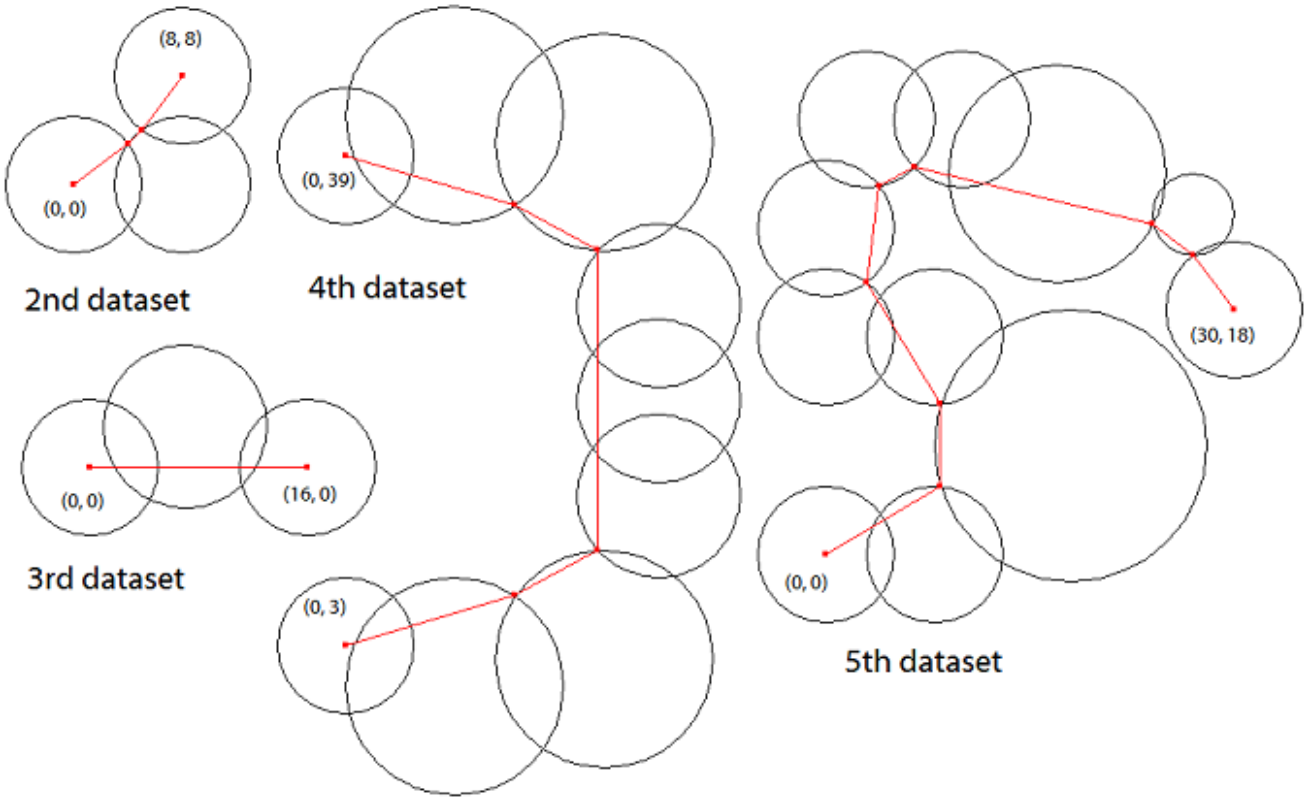


図 E-2: 鎖の例と対応する最短経路

Output

各データセットに対して，最初と最後の円の中心をつなぐ最短鎖中経路の長さを 1 行で出力せよ． 答えには 0.001 を超える誤差があってはいけない． それ以外の余計な文字を出力してはならない．

Sample Input

```
10
802 0 10
814 0 4
820 1 4
826 1 4
832 3 5
838 5 5
845 7 3
849 10 3
853 14 4
857 18 3
3
0 0 5
8 0 5
8 8 5
3
0 0 5
7 3 6
16 0 5
9
0 3 5
8 0 8
19 2 8
23 14 6
23 21 6
23 28 6
19 40 8
8 42 8
0 39 5
11
0 0 5
8 0 5
18 8 10
8 16 5
0 16 5
0 24 5
3 32 5
10 32 5
17 28 8
27 25 3
30 18 5
0
```

Output for the Sample Input

```
58.953437
11.414214
16.0
61.874812
63.195179
```

(End of Problem E) A B C D E F G

Problem F

一般化ポーカー

1からMのランクを持つ$N \times M$枚のカードのデッキがある．それぞれのランクごとにN枚のカードが含まれている．

ランクmのカードのことを以下ではmと記述する．

デッキからL枚の手札をランダムに引くことができる．その手札 が与えられたパターンにマッチしているとボーナスが与えられる．パターンは，以下のように記述される．

$$\text{hand\_pattern} = \text{card\_pattern}_1 \text{ ' ' } \text{card\_pattern}_2 \text{ ' ' ... ' ' } \text{card\_pattern}_L$$

$$\text{card\_pattern} = \text{'*'} \mid \text{var\_plus}$$

```
var_plus = variable | var_plus '+'
variable = 'a' | 'b' | 'c'
```

hand_pattern
: hand_pattern中の各card_patternがそれぞれ手札中の異なるカードにマッチすれば， その手札はhand_patternにマッチする．

card_pattern
: card_patternがアステリスク('*')の時はどのカードにもマッチする． 文字'a', 'b', 'c'は変数を表し，同じ変数は同じランクのカードとマッチする． 変数の後に1個以上のプラス('+')を続けたパターンは，変数に対応するランクに'+'の個数を足して得られるラ ンクのカードにマッチする．変数の後に'+'をいくつか続けたパターンがhand_patternに含まれる時は， 同じ変数の後に'+'をより少ない数(ゼロを含む)続けたパターンがすべてhand_patternに含まれると 仮定してよい．たとえば，'a+++'がhand_patternに現れる時は，'a', 'a+', 'a++' が hand_patternに現れる．

異なる変数の間には，表すランクについての制約はない． たとえば，'a'に対応するランクと'b'に対応するランクは同じでもよいし，異なっていてもよい．

hand_patternの例を示す．パターン

```
a * b a b
```

は，'a'が3を, 'b'が10 を(または'a'が10を, 'b'が3を)表すとすると

```
3 3 10 10 9
```

とマッチする． 同じパターンは手札

```
3 3 3 3 9
```

ともマッチする．この時は，'a'も'b'も3を表す．パターン

```
a a+ a++ a+++ a++++
```

は手札

```
4 5 6 7 8
```

とマッチする．この場合，'a'は4になる．

デッキからランダムに取り出した手札が与えられたhand_patternにマッチする確率を求めるプログラムを作成しなさい．

Input

入力は複数のデータセットからなる．それぞれのデータセットは以下の形式からなる．

> *N M L*
> *card_pattern*$_1$ *card_pattern*$_2$ ... *card_pattern*$_L$

最初の行は空白で区切られた三つの正の整数*N*, *M*, *L*を含んでいる． *N*は同ランクのカードの数, *M*は

ランクの数，Lは手札の枚数を示す．N, M, Lは以下の制約を満たす．

$1 \leq N \leq 7$
$1 \leq M \leq 60$
$1 \leq L \leq 7$
$L \leq N \times M$

2行目はhand_patternを記述している．

入力の終わりは空白一つで区切られたゼロ3個からなる行によって示される．

Output

各データセットについて，そのhand_patternに手札がマッチする確率を小数として出力せよ．

出力には10^{-8}を超える誤差があってはならない．

それ以外の文字を含んでいてはならない．

Sample Input

```
1 1 1
a
3 3 4
a+ * a *
2 2 3
a a b
2 2 3
* * *
2 2 3
* b b
2 2 2
a a
2 3 3
a a+ a++
2 6 6
a a+ a++ b b+ b++
4 13 5
a a * * *
4 13 5
a a b b *
4 13 5
a a a * *
4 13 5
a a+ a++ a+++ a++++
4 13 5
* * * * *
4 13 5
a a a b b
4 13 5
a a a a *
7 60 7
a b a b c c *
7 60 7
* * * * * * *
7 60 7
a a+ a++ a+++ a++++ a+++++ a++++++
1 14 4
b a+ a a
0 0 0
```

Output for the Sample Input

```
1.0000000000
0.8809523810
1.0000000000
1.0000000000
1.0000000000
0.3333333333
0.4000000000
0.1212121212
0.4929171669
0.0492196879
0.0228091236
0.0035460338
1.0000000000
0.0014405762
0.0002400960
0.0002967709
1.0000000000
0.0000001022
0.0000000000
```

(End of Problem F) A B C D E F G

Problem G

ACM洋菓子店

先月自分の店をオープンしたお菓子職人のアンバー・C・マエスは，店の宣伝のために ICPC (International Chocolate Patissier Competition)に出品することにした． そのために，板チョコレートの試作を繰り返していたアンバーは， ついに本当においしいレシピを発見した．しかし，そのレシピではチョコレートをきれいな長方形に整形するために高度な技術が必要だった．つい今しがたも図G-1に示すような， おかしな形の板チョコレートを作ってしまったところだ．

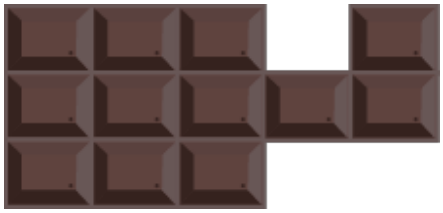


図 G-1: おかしな形の板チョコレート

板チョコレートはたくさんの小長方形のチョコレートでできている．隣接する小長方形の間には溝が掘られており，割りやすくなっている． アンバーは，おかしな形の板チョコレートをいくつかの長方形の断片に切り分けて店で売ろうと考えた．彼女は次のように板チョコレートを切ることにした．

- 板チョコレートは溝に沿って切る．
- 断片が全て長方形になるように切り分ける．
- できるだけ少ない数の断片に切り分ける．

このルールに従えば，図G-1は例えば図G-2のように切り分けられる． 図G-3には長方形でない断片が含まれており，図G-4は図G-2よりも多くの断片に切り分けているため，ルールに従っていない．

図 G-2: ルールに従って切り分けた例

図 G-3: 長方形でない断片を含むような切り分け方の例

図 G-4: 図G-2よりも多くの断片に切り分ける例

あなたの仕事は，このルールに従って板チョコレートを切り分けた時，いくつの断片に分かれるかを求めるプログラムを作ることである．

Input

入力は，複数のデータセットからなり，入力の終わりはスペースで区切られたゼロ2つからなる行である．各データセットは，次の形式をしている．

h w
$r_{(1,1)}$ … $r_{(1,w)}$
$r_{(2,1)}$ … $r_{(2,w)}$
…
$r_{(h,1)}$ … $r_{(h,w)}$

h と w は，板チョコレートの直交する2方向の長さを小長方形チョコレートの数で表した整数であり，$2 \leq h \leq 100$, $2 \leq w \leq 100$ と仮定してよい．続く h 行はそれぞれ，w 個の文字で構成されており，文字 $r_{(i,j)}$ は，場所 (i, j) の小長方形チョコレートの有無を表す．文字の意味は，次の通り．

- ".": 小長方形チョコレートなし
- "#": 小長方形チョコレートあり

連結していない板チョコレート(図G-5)や穴のある形の板チョコレート(図G-6や図G-7)を表すようなデータセットが無いことを仮定してよい．また，各データセットには少なくとも1つの"#"の文字が含まれていることを仮定してよい．

図 G-5: 連結していない板チョコレート

図 G-6: 穴のある形の板チョコレート

図 G-7: 穴のある形の板チョコレートのもう1つの例

Output

入力データセットごとに，ルールに従って切り分けた時の断片の数を表す整数のみからなる行を出力せよ．

Sample Input

3 5

```
3 5
###.#
#####
###..
4 5
.#.##
.####
####.
##.#.
8 8
.#.#.#.#
########
.######.
########
.######.
########
.######.
########
8 8
.#.#.#.#
########
.##.#.#.
##....##
.##.###.
##...###
.##.###.
###.#.##
4 4
####
####
####
####
0 0
```

Output for the Sample Input

```
3
5
11
19
1
```

(End of Problem G) A B C D E F G