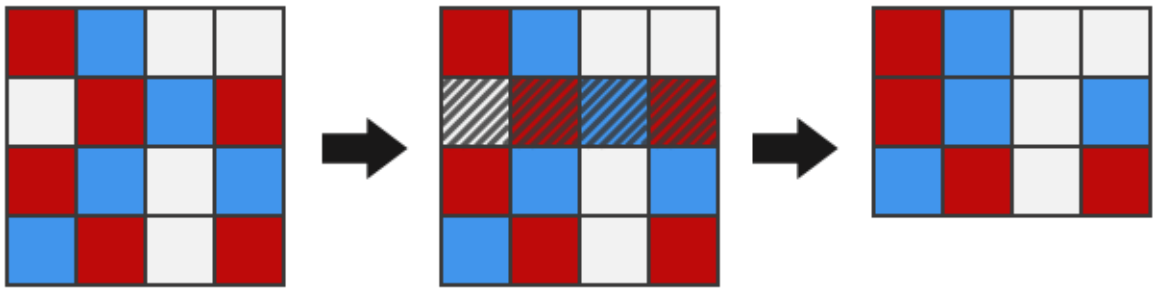
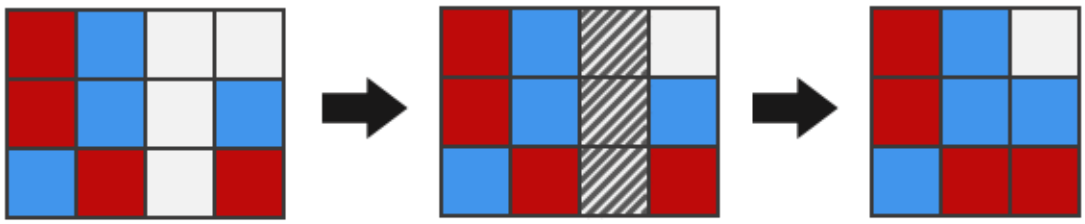
#	Problem Name	Time Limit	Memory Limit
A	boards	1 sec.	64 MB
B	buses	1 sec.	64 MB
C	crosses	1 sec.	64 MB
D	ghosts	1 sec.	64 MB
E	grail	1 sec.	64 MB
F	greyscale	1 sec.	64 MB
G	history	1 sec.	64 MB
H	rooks	1 sec.	64 MB
I	squares	1 sec.	64 MB

ამოცანა A. “გაფერადებული დაფა”

მოცემულია N სტრიქონისგან და M სვეტისგან შემდგარი დაფა, რომლის თითოეული კვადრატი შეღებილია ერთ-ერთში K ფერიდან. დაფაზე ყოველი ფერის უჯრედების რაოდენობა ერთიდაიგივეა. მანაომ უნდა წაშალოს ამ დაფის რომელიღაც სტრიქონები და სვეტები. ყოველი ასეთი წაშლის შემდეგ დაფა ისევ ერთიანი ხდება, ანუ სტრიქონის წაშლისას მის ქვევით მყოფი სტრიქონები ჩამოიჩოჩება ზემოთ, ხოლო სვეტის წაშლისას მის მარჯვნივ მყოფი სვეტები ჩამოიჩოჩება მარცხნივ.





მანაო ცდილობს სტრიქონების და სვეტების ისე წაშლას, რომ საბოლოო ჯამში მიიღოს კვადრატული ფორმის ერთსადაიმავე ფერის უჯრედებით შედგენილი დაფა. იპოვეთ ასეთი დაფის მაქსიმალური შესაძლო ზომა.

შეზღუდვები

$1 \leq N, M \leq 30$

$1 \leq K \leq 26$

$N * M$ იყოფა K-ზე.

შემომავალი ფაილის ფორმატი

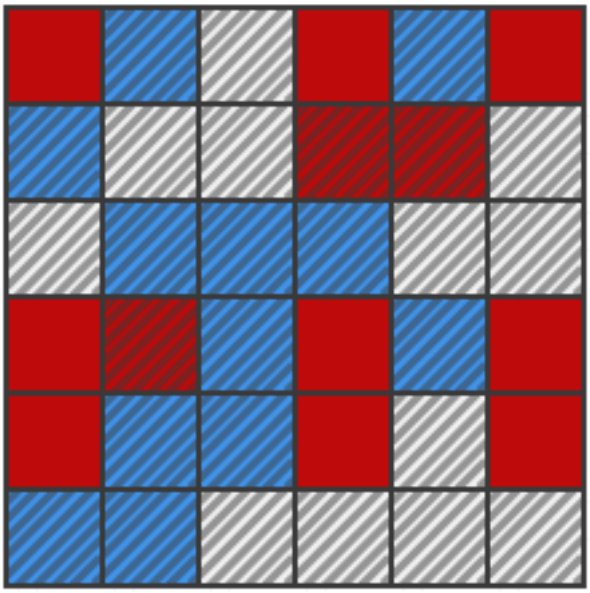
შესატან მონაცემთა boards.in ფაილის პირველი ხაზი შეიცავს სამ მთელ რიცხვს N, M, K. შემდეგი N ხაზიდან თითოეული შეიცავს M ცალ ზედა რეგისტრის ლათინურ ასოს. i-ური სტრიქონის j-ური სიმბოლო დაფის (i, j) უჯრედის ფერს განსაზღვრავს. ერთი ფერით შეღებილ უჯრედებს ერთიდაიგივე ასო შეესაბამება, განსხვავებულებს კი განსხვავებული.

გამომავალი ფაილის ფორმატი

გამოსატან მონაცემთა boards.out ფაილში დაბეჭდეთ "C X" ფორმის ტექსტი. C არის იმ ფერის შესაბამისი ასო, რომელიც ექნება საბოლოო დაფის უჯრედებს, ხოლო X კი დაფის გვერდის მაქსიმალური შესაძლო სიგრძე. თუ მაქსიმალური ზომის დაფის მიღება სხვადასხვა ფერით შეიძლება, გამოიტანეთ ამ ფერების შესაბამის ასოებს შორის უმცირესი.

შემომავალი ფაილის მაგალითი (boards.in)	გამომავალი ფაილის მაგალითი (boards.out)
6 6 3 RBWRBR BWWRRW WBBBWW RRBRBR RBBRWR BBWWWW	R 3

განმარტება.



ამოცანა B. “ავტობუსები”

ქალაქში, სადაც მანაო ცხოვრობს, ავტობუსები საზოგადოებრივი ტრანსპორტის ყველაზე პოპულარული საშუალებაა. ქალაქში სულ არის N რაოდენობის ავტობუსების გაჩერება, რომელთა M წყვილი ორმხრივი გზით არის შეერთებული. გაჩერებები გადანომრილია მთელი რიცხვებით 1-დან N-მდე. ავტობუსების მარშრუტი იმდენად ბევრია, რომ ყოველი გზისთვის მოიძებნება მასზე გამავალი ერთი მაინც მარშრუტი. ავტობუსები გაჩერებებზე ყოველთვის ჩერდებიან.

ავტობუსით მოგზაურობისას საჭიროა სამგზავრო ბილეთის შეძენა. იმისთვის, რომ მოსახლეობის მიერ ბილეთების შეძენა გაკონტროლდეს, გაჩერებებს მეთვალყურეობენ მონიტორინგის სამსახურის თანამშრომლები. როდესაც მათ გაჩერებაზე ავტობუსი მოდის, ისინი უმოწმებენ ბილეთს ყველა იმ მგზავრს, რომელიც ავტობუსში ზის, ჩამოდის ან ამოდის.

მანაომ დაამუღამა, რომ მონიტორინგის სამსახურის თანამშრომლები მხოლოდ S რაოდენობის სტრატეგიულად მნიშვნელოვან გაჩერებაზე დგანან. ამიტომ, თუ ისე იმოგზაურებ, რომ ასეთ გაჩერებებზე არ გაიარო, ბილეთის ყიდვა სულაც არ მოგიწევს. მანაომ შეადგინა ქალაქში თავისი გადაადგილების გრაფიკი მომავალი ათი წლისთვის და აინტერესებს, რამდენი ბილეთის ეკონომიას მოახერხებს (იგი აპირებს ექსკლუზიურად ავტობუსებით იაროს). კონკრეტულად, მას აქვს K გაჩერებათა წყვილი და ყოველი მათგანისთვის უნდა იცოდეს, შეძლებს თუ არა ერთიდან მეორემდე მიღწევას ბილეთის შეძენის გარეშე.

შეზღუდვები

$2 \leq N \leq 10,000$

$1 \leq M \leq 20,000$

$1 \leq K \leq 30,000$

$0 \leq S \leq N$

ნებისმიერი გაჩერებიდან ნებისმიერ სხვამდე მიღწევა შესაძლებელი იქნება პირდაპირი გზის საშუალებით ან სხვა გაჩერებების გავლით.

შემომავალი ფაილის ფორმატი

შემოსატანი მონაცემების buses.in ფაილის პირველი ხაზი შეიცავს ოთხ მთელ რიცხს N, M, K, S. შემდეგი ხაზი შეიცავს S ცალ განსხვავებულ რიცხვს - სტრატეგიულად მნიშვნელოვან გაჩერებათა ნომრებს. შემდეგი M ხაზიდან თითოეული ერთ გზას აღწერს და ორ განსხვავებულ რიცხვს შეიცავს - იმ გაჩერებების ნომრებს, რომლებსაც შესაბამისი გზა აკავშირებს. შემდეგი K ხაზიდან თითოეული კი მანაოს გრაფიკის მორიგ ჩანაწერს შეიცავს ორი განსხვავებული გაჩერების ნომრის სახით.

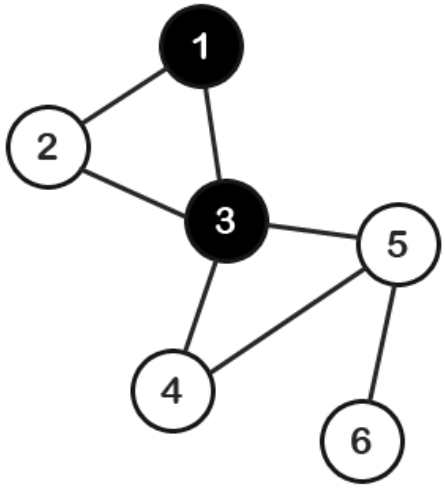
გამომავალი ფაილის ფორმატი

გამოსატან მონაცემთა buses.out ფაილში დაბეჭდეთ K სიგრძის სტრიქონი. მისი i-ური სიმბოლო მანაოს გრაფიკის i-ური ჩანაწერის პასუხს შეესაბამება და არის '1', თუ იგი მოახერხებს i-ურ გაჩერებათა წყვილს შორის ბილეთის შეუძენლად გადაადგილებას.

შემომავალი ფაილის მაგალითი (buses.in)	გამომავალი ფაილის მაგალითი (buses.out)
6 7 6 2 1 3 3 2 3 1 1 2 4 3 3 5 5 4 5 6 1 3 1 2 4 5 6 2 6 4 3 6	001010

განმარტება.

ავტობუსების გაჩერებათა სისტემა შემდეგნაირად გამოიყურება (შავი ფერით სტრატეგიულად მნიშვნელოვანი გაჩერებებია მონიშნული):



ამოცანა C. "ჯვრები"

მოცემულია N სტრიქონისგან და M სვეტისგან შემდგარი დაფა. ამ დაფის ყოველი სტრიქონის და სვეტის გადაკვეთაზე მყოფ უჯრედში წერია ან '.', ან 'X'. (i, j) აღნიშნავს i-ური სტრიქონისა და j-ური სვეტის გადაკვეთაზე მყოფ უჯრედს.

ამ დაფიდან გვინდა ამოვჭრათ ფიგურები, რომლებსაც ჯვრებს დავუძახებთ. ჯვარი შედგება 4k+1 ცალი 'X' უჯრედისგან, სადაც k მთელი დადებითი რიცხვია. ჯვარს აქვს შუაწერტილი (x, y) და მისგან ჰორიზონტალურად და ვერტიკალურად ორივე მიმართულებით მიმავალი k სიგრძის ტოტები. ფორმალურად რომ ვთქვათ, 4k+1 უჯრედისგან შემდგარი ფიგურა არის ჯვარი, თუ მის ყოველ უჯრედში წერია 'X' და მოიძებნება ისეთი (x, y) წყვილი, რომ ეს ფიგურა შეიცავს ყველა შემდეგ უჯრედს:

1) (x, y) უჯრედს

2) (x - i, y) უჯრედებს 1 <= i <= k -სთვის.

3) (x + i, y) უჯრედებს 1 <= i <= k -სთვის.

4) (x, y - i) უჯრედებს 1 <= i <= k -სთვის.

5) (x, y + i) უჯრედებს 1 <= i <= k -სთვის.

ორი ჯვარი განსხვავებულია, თუ მათი შემადგენელი უჯრედების სიმრავლეები ტოლი არაა. დაადგინეთ, რამდენი სხვადასხვა ჯვრის ამოჭრა შეიძლება მოცემული დაფიდან.

შეზღუდვები

$1 \leq N, M \leq 300$

შემომავალი ფაილის ფორმატი

შემოსატანი მონაცემების crosses.in ფაილის პირველი სტრიქონი შეიცავს ერთი ჰარით გამოყოფილ ორ მთელ რიცხვს N და M-ს. შემდეგ წერია N სტრიქონი, თითოეულ რომელთაგანზეც განლაგებულია M სიმბოლო '.' ან 'X' - მოცემული დაფა.

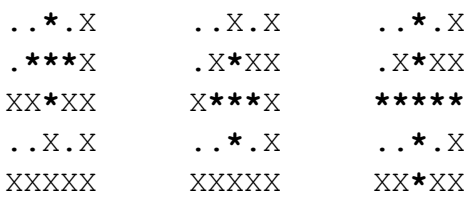
გამომავალი ფაილის ფორმატი

გამოსატან მონაცემთა crosses.out ფაილში გამოიტანეთ დაფიდან ამოჭრადი ჯვრების რაოდენობა.

შემომავალი ფაილის მაგალითი (crosses.in)	გამომავალი ფაილის მაგალითი (crosses.out)
5 5 ..X.X .XXXX XXXXX ..X.X XXXXX	3

მინიშნება.

ამ დაფიდან შემდეგი 3 ჯვრის ამოჭრა შეიძლება:



ამოცანა D. "HMMII Ghosts"

არის ერთი ძველი სტრატეგიული თამაშში სახელად Heroes of Might and Magic II. მასში ბრძოლის დროს მოწინააღმდეგეების ჯარი დაყოფილია რამდენიმე რაზმად. ყოველი რაზმი ერთსადაიმავე ჯიშის მონსტრებით კომპლექტდება. ყოველ მონსტრის ჯიშს გააჩნია თავისი Health Points (HP) - რამდენი ერთეული ჯანი აქვს და Damage Points (DP) - რამდენ ერთეულ ჯანს აკლებს მოწინააღმდეგეს იერიშის დროს. როდესაც მონსტრები ერთ N ზომის რაზმში არიან გაერთიანებული, იერიშის დროს ისინი DP * N ერთეულ ჯანს აკლებენ მოწინააღმდეგე რაზმს.

აღვწეროთ რა ხდება რაზმზე თავდასხმის დროს. ვთქვათ, რაზმში N მონსტრია, რომელსაც H ჯანის ერთეული გააჩნია, მოწინააღმდეგემ მას პირველად შეუტია და X ჯანი დააკლო. მაშინ რაზმში დაიღუპება X div H (მთელი გაყოფის შედეგი იგულისხმება) მონსტრი, ხოლო ერთ-ერთ გახდარჩენილს დააკლდება X mod H (გაყოფისგან ნაშთი იგულისხმება) ჯანის ერთეული. თუ რაზმში ყველა მონსტრი დაიღუპა, რაზმი იღუპება.

განვიხილოთ, რა ხდება როდესაც რაზმზე მეორედ ხდება იერიში. დავუშვათ, რაზმის N მონსტრისგან ერთ-ერთს უკვე აქვს დაკლებული P ჯანი. გამოვთვალოთ რაზმის ჯამური ჯანი S = N * H - P. მაშინ იერიშის შემდეგ რაზმში დარჩება (S - X) div H სრულად ჯანმრთელი მონსტრი, თუ (S - X) იყოფა H-ზე, წინააღმდეგ შემთხვევაში კი (S - X) div H ჯანმრთელი მონსტრი და ერთი მონსტრი (S - X) mod H ჯანის ერთეულით. ცხადია, თუ S არ აღემატება X-ს, რაზმი იღუპება.

ზოგიერთ ჯიშს გააჩნია განსაკუთრებული შესაძლებლობები. თამაშის ერთ-ერთი საშინელი მონსტრების ჯიშია Ghost - მოჩვენება. როდესაც მოჩვენებები უტევენ მოწინააღმდეგის რაზმს, იერიშის შედეგად დაღუპული ყველა მონსტრი მოჩვენებად გადაიქცევა და მათ რიგებს ავსებს. რა ჯიშისაც არ უნდა ყოფილიყო ეს მონსტრი, მის მოჩვენებას ყოველთვის მოჩვენების HP და DP ექნება.

ჩვენ თავს დავესხით მოჩვენებათა რაზმს, რომელშიც G მონსტრია GHP ჯანის და GDP იერიშის ერთეულით. ჩვენ გვაქვს ერთი რაზმი, რომლის მონსტრებს აქვთ MHP ჯანის და MDP იერიშის ერთეული. ბრძოლის პროცესი შემდეგნაირად იმართება: ჯერ ჩვენი რაზმი უტევს, შემდეგ მოჩვენებები გვიტევენ, შემდეგ ჩვენი რაზმის დაღუპულები მოჩვენებებს უერთდებიან. ეს პროცესი მანამ გრძელდება, სანამ ერთ-ერთი რაზმი არ დაიღუპება. გამოთვალეთ, მინიმუმ რამდენი მონსტრი უნდა იყოს ჩვენს რაზმში, რომ მოვახერხოთ მოჩვენებათა დამარცხება.

შეზღუდვები

$1 \le G \le 1{,}000{,}000$

$1 \le GHP \le 100$

$1 \le GDP \le 100$

$1 \le MHP \le 500$

$1 \le MDP \le 100$

შემომავალი ფაილის ფორმატი

შემოსატან მონაცემთა ghosts.in ფაილის ერთადერთი ხაზი შეიცავს 5 მთელ რიცხვს *G*, *GHP*, *GDP*, *MHP*, *MDP*.

გამომავალი ფაილის ფორმატი

გამოსატანი მონაცემების ghosts.out ფაილში ჩაწერეთ ამოცანის პასუხი.

შემომავალი ფაილის მაგალითი (ghosts.in)	გამომავალი ფაილის მაგალითი (ghosts.out)
5 10 3 6 4	6

განმარტება.

თუ რაზმში 6 მონსტრი გვეყოლება, ბრძოლა შემდეგნაირად წარიმართება. პირველი იერიშით ჩვენ მოჩვენებათა რაზმს 6*4=24 ჯანის ერთეულს დავაკლებთ, რის შედეგადაც იქ დარჩება 3 მოჩვენება, რომელთაგან ერთს ექნება დარჩენილი მხოლოდ 6 ერთეული HP. პასუხად მოჩვენებები 3*3=9 ერთეულს დაგვაკლებენ, ანუ მოკლავენ ჩვენს ერთ-ერთ მონსტრს და მეორეს 3 HP-ს დააკლებენ. მოჩვენებათა რიცხვი 4-მდე გაიზრდება. ჩვენ მეორე იერიშით მოჩვენებებს 5*4=20 ერთეულს დავაკლებთ, ანუ დარჩება 2 მოჩვენება, რომელთაგან ერთს მხოლოდ 6 HP აქვს. პასუხად მოჩვენებები 2*3=6 ერთეულს დაგვაკლებენ, 1 მონსტრს მოკლავენ და შემოიერთებენ თავიანთ რიგებში. შემდეგი იერიშით 4*4=16 ერთეულს დავაკლებთ და ერთ ჯანსაღ მოჩვენებას დავტოვებთ და ასე შემდეგ. საბოლოო გამარჯვებას მე-5 თავდასხმისას ვიზეიმებთ.

ამოცანა E. "გრაალის თასი"

კვლავ ვუბრუნდებით დიად თამაშს Heroes of Might and Magic II. მისი ზოგადი სათამაშო რუკა H x W უჯრედისგან შედგება, ყოველი რომელთაგანი ან გავლადია, ან დაბრკოლებას შეიცავს და მასზე გავლა შეუძლებელია. ასევე ზოგიერთ გავლად უჯრედში დგას მონსტრების რაზმი, რომლის ერთხელ დამარცხების შემდეგ ამ უჯრედზე შესაძლებელი ხდება თავისუფლად გადაადგილება ერთხელ ან ბევრჯერ. უჯრედს, რომელიც რუკის ზემო მარცხენა კუთხეში იმყოფება, (1, 1)-ად აღვნიშნავთ, ხოლო მისგან ჰორიზონტალურად x-1 უჯრედით და ვერტიკალურად y-1 უჯრედით დაშორებულ უჯრედს (x, y)-ით.

სათამაშო რუკაზე გმირები გადაადგილდებიან, რომლებსაც თავიანთი ჯარი ჰყავთ. ჩვენი გმირი თავიდან $(x1, y1)$ უჯრედში იმყოფება. მისი მისია უძვირფასესი არტეფაქტის, გრაალის თასის პოვნა და სამშობლოს დედაქალაქში გადატანაა, რომელიც $(x2, y2)$ უჯრედშია მოთავსებული. გმირმა ახლახანს შეიტყო, რომ გრაალის თასი $(x3, y3)$ უჯრედშია ჩამარხული. ამრიგად, მან უნდა მიაღწიოს გრაალის თასის შემცველ უჯრედამდე, ამოთხაროს იგი და შემდეგ დედაქალაქში მივიდეს. ერთი უჯრედიდან მეორეში გადასვლა მაშინ შეიძლება, როდესაც მათ საერთო გვერდი აქვთ.

ყოველ უჯრედში მოთავსებული მონსტრების რაზმისთვის ცნობილია, რამდენ ჯარისკაცს დააკარგვინებს იგი ჩვენს გმირს. ვინაიდან წინ კიდევ ბევრი ბრძოლაა, გმირს უნდა, მინიმალური რაოდენობის ჯარისკაცი დაკარგოს თავისი საწყისი პოზიციიდან გრაალამდე და შემდეგ დედაქალაქამდე გზაში. დაადგინეთ ის მინიმალური რაოდენობა.

შეზღუდვები

$1 \le H, W \le 300$

$1 \le x1, x2, x3 \le W$

$1 \le y1, y2, y3 \le H$

(x1, y1), (x2, y2), (x3, y3) უჯრედები განსხვავებულია.

(x1, y1), (x2, y2), (x3, y3) უჯრედები გავლადია და იქ არ დგას მონსტრების რაზმი.

გარანტირებულია, რომ გმირის საწყისი პოზიციიდან გრაალის უჯრედი მიღწევადია, ხოლო მისგან მიღწევადია დედაქალაქის უჯრედი.

ერთი მონსტრების რაზმი გმირს შეიძლება აკარგვინებდეს 1-დან 1000-მდე ჩათვლით ჯარისკაცს.

შემომავალი ფაილის ფორმატი

შესატან მონაცემთა grail.in ფაილის პირველი ხაზი შეიცავს ორ მთელ რიცხვს H და W. მეორე ხაზში ჩაწერილია ექვსი მთელი რიცხვი *x1, y1, x2, y2, x3, y3*. შემდეგ მოდის H ხაზი, თითოეული მათგანი კი W რიცხვს შეიცავს. i-ური ხაზის j-ური რიცხვი სათამაშო

რუკის (i, j) უჯრედს შეესაბამება და არის -1, თუ ამ უჯრედზე გავლა შეუძლებელია, 0, თუ ეს უჯრედი გავლადია და არ შეიცავს მონსტრების რაზმს და არის დადებითი რიცხვი X, თუ ეს უჯრედი გავლადია და მასში X ჯარისკაცის დამაკარგვინებელი მონსტრების რაზმი დგას.

გამომავალი ფაილის ფორმატი

გამოსატან მონაცემთა grail.out ფაილში დაბეჭდეთ ერთი მთელი რიცხვი - მინიმალური შესაძლო ჯარისკაცების რაოდენობა, რომლებიც დაიღუპებიან გმირის მისიის შესრულებისას.

შემომავალი ფაილის მაგალითი (grail.in)	გამომავალი ფაილის მაგალითი (grail.out)
5 5 1 1 1 5 5 1 0 10 10 10 0 1 10 10 10 10 1 10 10 10 10 100 -1 0 0 0 0 10 10 10 10	62

ამოცანა F. "ნაცრისფერის შეკუმშვა"

განვიხილოთ H x W ზომის შავ-თეთრი სურათი, რომლის თითოეული პიქსელის სიკაშკაშე კოდირებულია [0, 255] შუალედიდან ერთი მთელი რიცხვით. ამიტომ პიქსელს შეიძლება შევუსაბამოთ ორი 16-ობითი ციფრი, მაგალითად 10 ჩაიწერება როგორც 0A, 178 როგორც B2, ხოლო 190 როგორც BE.

ჩვენ გვჭირდება ამ ბიტმაპის შეკუმშვა. ამას ვაპირებთ ყოველი პიქსელისთვის მხოლოდ ერთი 16-ობითი ციფრის შენახვით ორის ნაცვლად. ამიტომ 256 ტონის ნაცვლად გვექნება მხოლოდ 16 ტონი და ის პიქსელები, რომლების სიკაშკაშე განსხვავებულია, უნდა გადავღებოთ რომელიმე ტონში ამ თექვსმეტიდან.

პიქსელის საწყის სიკაშკაშესა და გადაღებვის შემდეგ სიკაშკაშეს შორის სხვაობის აბსოლუტურ მნიშვნელობას დავარქვათ ამ პიქსელის გადაკოდირების გადახრა. ჩვენი ამოცანაა, ისე შევარჩიოთ ის 16 ტონი და მათი განაწილება პიქსელებზე, რომ ბიტმაპის გადაკოდირების ჯამური გადახრა იყოს მინიმალური. ბიტმაპის გადაკოდირების ჯამური გადახრა მისი ყოველი პიქსელისთვის გადახრების ჯამია. თუ არსებობს მინიმალური ჯამური გადახრის მიღების რამდენიმე ვარიანტი, ნებისმიერი გამოიტანეთ.

შეზღუდვები

$1 \leq H, W \leq 500$

ყოველი პიქსელის სიკაშკაშე მთელი რიცხვია [0, 255] შუალედიდან.

შემომავალი ფაილის ფორმატი

შესატან მონაცემთა greyscale.in ფაილის პირველი ხაზი შეიცავს ორ მთელ რიცხვს H და W. შემდეგი H ხაზიდან თითოეული შეიცავს $2W$ ცალ 16-ობით ციფრს ('0'-'9', 'A'-'F') რაიმე გამყოფის გარეშე. i-ური სტრიქონის 2j და 2j+1 სიმბოლოები (j, i) პიქსელის სიკაშკაშეს განსაზღვრავენ, მათში პირველი უფროს თანრიგს, მეორე კი უმცროს თანრიგს შეესაბამება.

გამომავალი ფაილის ფორმატი

გამოსატან მონაცემთა greyscale.out ფაილის პირველ ხაზზე დაბეჭდეთ 16 რიცხვი [0, 255] შუალედიდან. ამათგან i-ური (0-დან გადათვლით) რიცხვი შეესაბამება 16-ობითი თვლის სისტემის i ციფრისთვის მინიჭებულ სიკაშკაშეს. შემდეგ დაბეჭდეთ H ხაზი და თითოეულ მათგანზე W 16-ობითი ციფრი გამყოფების გარეშე. i-ურ ხაზში j-ური სიმბოლო (j, i) პიქსელის სიკაშკაშეა კოდირების შემდეგ.

შემომავალი ფაილის მაგალითი (greyscale.in)	გამომავალი ფაილის მაგალითი (greyscale.out)
5 5 0001020304 0506070809 AABACADAEA A0B0C0D0E0 2030405060 7080909192	160 192 170 218 202 2 176 80 48 234 7 96 186 64 208 32 55555 AAAAA 2C439 061E3 F8D7B

განმარტება.

ამ მაგალითში მინიმალური ჯამური გადახრა 18-ს შეადგენს.

ამოცანა G. "ისტორია"

ერთხელ მანაო გავიდა გასაუბრებაზე მსოფლიო ისტორიისა და კულტურის ინსტიტუტში. მას შესთავაზეს შემდეგი დავალების ამოხსნა.

არის N ისტორიული პიროვნება და ყოველისთვის ცნობილია მისი დაბადების წელი - რიცხვი [1, 2000] დიაპაზონიდან. როგორც მოგეხსენებათ, ზოგიერთი ისტორიული პიროვნება ჩვენს წელთაღრიცხვამდე დაიბადა. მანაოს ამოცანა ზუსტად იმის დადგენაშია, თუ რომელია ასეთი.

მანაოს აქვს დამატებითი ინფორმაცია - პიროვნებათა M წყვილისთვის მან იცის, ამ წყვილში რომელი დაიბადა უფრო ადრე.

მანაომ მაშინ სამსახური ვერ მიიღო. ვნახოთ, თქვენ თუ უშველიდით. ყოველი პიროვნებისთვის გადაწყვიტეთ, როდის დაიბადა იგი - ჩვენს წელთაღრიცხვამდე თუ მის მერე. თქვენი განაწილება არ უნდა ეწინააღმდეგებოდეს მოცემულ ინფორმაციას. თუ არსებობს რამდენიმე შესაძლო განაწილება, გამოიტანეთ ნებისმიერი. თუ მოცემული ინფორმაცია თვითონვე წინააღმდეგობრივია, გამოიტანეთ "Data inconsistent".

შეზღუდვები

$2 \le N \le 1000$

დაბადების წლები [1, 2000] დიაპაზონიდან მთელი რიცხვებია.

$1 \le M \le 10,000$

შემომავალი ფაილის ფორმატი

შემოსატან მონაცემთა history.in ფაილის პირველი ხაზი შეიცავს ორ მთელ რიცხვს N და M. მეორე ხაზში წერია N ცალი რიცხვი - ისტორიულ პიროვნებათა დაბადების წლები. შემდეგი M ხაზიდან თითოეული არის "X < Y" სახის, სადაც X და Y არიან ისტორიულ პიროვნებათა ნომრები. ნომრები მინიჭებულია 1-დან იმ მიმდევრობით, რომლითაც ამ პიროვნებათა დაბადების წლები ეწერა ფაილის მეორე ხაზში. ეს ჩანაწერი ნიშნავს, რომ პიროვნება X დაიბადა პიროვნება Y-ზე უფრო ადრე.

გამომავალი ფაილის ფორმატი

გამოსატან მონაცემთა history.out ფაილში დაბეჭდეთ "Data inconsistent" (ბრჭყალების გარეშე), თუ მოცემული ინფორმაცია წინააღმდეგობრივია. წინააღმდეგ შემთხვევაში გამოიტანეთ N ხაზი. მათგან i-ური i-ურ ისტორიულ პიროვნებას შეესაბამება და შეიცავს ტექსტს "BC", თუ პიროვნება ჩვენს წელთაღრიცხვამდე დაიბადა ან "AD", თუ ჩვენს წელთაღრიცხვაში.

შემომავალი ფაილის მაგალითი (history.in)	გამომავალი ფაილის მაგალითი (history.out)
4 3 1500 1600 14 66 1 < 2 1 < 3 4 < 3	BC AD AD BC
3 3 1 2 3 1 < 2 2 < 3 3 < 1	Data inconsistent

ამოცანა H. “ეტლების ომი”

მანაო იმდენად მაგარია ჭადრაკში, რომ ყველას უგებს. წლების მანძილზე მას ეს საკმაოდ მობეზრდა და ამიტომ მან გადაწყვიტა ჭადრაკის მონათესავე თამაშები გამოიგონოს და ისინი ითამაშოს. მისი ერთ-ერთი ქმნილებაა "ეტლების ომი". მანაო ამ თამაშის წესებს დიდი ხნის მანძილზე ხვეწავდა და საბოლოოდ შემდეგი რამ მიიღო.

დაფაზე, რომელიც N სტრიქონისგან და M სვეტისგან შედგება, ყოველ სვეტში დგას თეთრი და შავი ეტლი. პირველი მოთამაშე თეთრებით, ხოლო მეორე შავებით თამაშობს. სვლა მდგომარეობს რომელიმე ერთი ეტლის თავის სვეტში გადაადგილებაში წინ ან უკან. რაც მთავარია, ეტლს არ შეუძლია იმ უჯრედში დადგომა ან გავლა, რომელშიც მოწინააღმდეგეს ეტლია მოთავსებული, ანუ ფიგურების აღება არ ხდება. თამაში მაშინ მთავრდება, როდესაც მოთამაშეს აღარ შეუძლია არც ერთი თავისი ფიგურის გადაადგილება, რა შემთხვევაშიც იგი წაგებულად ცხადდება.

მანაო გენიალურად თამაშობს ამ თამაშს და ყველა პარტიას იგებს, რომლის მოგებაც თეორიულად შესაძლებელია. ამიტომ იგი მოწინააღმდეგეს თეთრ ფიგურებს უთმობს. თქვენ მოცემული პოზიციით უნდა დაადგინოთ, შესაძლებელია თუ არა მისი დამარცხება და დადებით შემთხვევაში გამოიტანოთ ისეთი სვლა, რომლის შემდეგაც ეს ისევ შესაძლებელი დარჩება. თუ ასეთი სვლა რამდენიმეა, გამოიტანეთ მათგან ისეთი, რომელიც ყველაზე მარცხენა სვეტში მდგომი ეტლით ხორციელდება. თუ ასეთი ეტლით რამდენიმე ოპტიმალური სვლა არსებობს, გამოიტანეთ ისეთი, რომლის შედეგადაც ეტლი ყველაზე მცირე ნომრის მქონე სტრიქონში აღმოჩნდება.

შეზღუდვები

$2 \le N \le 50$
$1 \le M \le 50$
ყოველ სვეტში თეთრი და შავი ეტლის სტრიქონის ნომერი განსხვავებულია და $[1, N]$ შუალედშია.

შემომავალი ფაილის ფორმატი

შესატან მონაცემთა rooks.in ფაილის პირველი ხაზი შეიცავს ორ მთელ რიცხვს N და M. შემდეგ წერია M ხაზი, მათგან i-ური i-ურ სვეტს აღწერს და ორ რიცხვს შეიცავს - ამ სვეტში თეთრი და შავი (ყოველთვის ამ მიმდევრობით) ეტლების სტრიქონების ნომრებს.

გამომავალი ფაილის ფორმატი

გამოსატან მონაცემთა rooks.out ფაილის პირველ ხაზში გამოიტანეთ "Impossible", თუ მანაოს დამარცხება შეუძლებელია. წინააღმდეგ შემთხვევაში დაბეჭდეთ "X Y" სახის სტრიქონი, სადაც X არის სვეტის ნომერი, რომელშიც უნდა გაკეთდეს სვლა, ხოლო Y არის იმ სტრიქონის ნომერი, სადაც უნდა გადაადგილოთ X-ური სვეტის ეტლი.

შემომავალი ფაილის მაგალითი (rooks.in)	გამომავალი ფაილის მაგალითი (rooks.out)
5 2 2 5 4 2	1 3

ამოცანა I. "კვადრატების გაჭრა"

სიბრტყეზე მოცემულია N სხვადასხვა ზომის და ორიენტაციის კვადრატი. ამ სიბრტყეს მოვდოთ წრფე და ყველა კვადრატი, რომელიც ამ წრფემ ორ ტოლი ფართობის მქონე ნაწილად გაჭრა, წავშალოთ. სხვა კვადრატებს, მიუხედავად იმისა, გადაკვეთა ისინი მოდებულმა წრფემ თუ არა, არაფერი მოსდით. შემდეგ სიბრტყეზე მოვდოთ მეორე წრფე და წავშალოთ ყველა ის კვადრატი, რომელიც ორ ტოლი ფართობის მქონე ნაწილად გაიჭრა ამ მეორე წრფით.

დაადგინეთ, რამდენნაირად შეგვიძლია წრფეების მოდება ისე, რომ საბოლოო ჯამში სიბრტყეზე არც ერთი კვადრატი არ დარჩეს. ორი ვარიანტი განსხვავებულია, თუ ერთში არსებობს ისეთი წრფე, რომელიც არ ემთხვევა მეორე ვარიანტის არც ერთ წრფეს.

ყოველი კვადრატი მოიცემა მისი ერთ-ერთი კუთხის $(x1, y1)$ კოორდინატებით და მოპირდაპირე კუთხის $(x2, y2)$ კოორდინატებით.

შეზღუდვები

$1 \le N \le 5000$

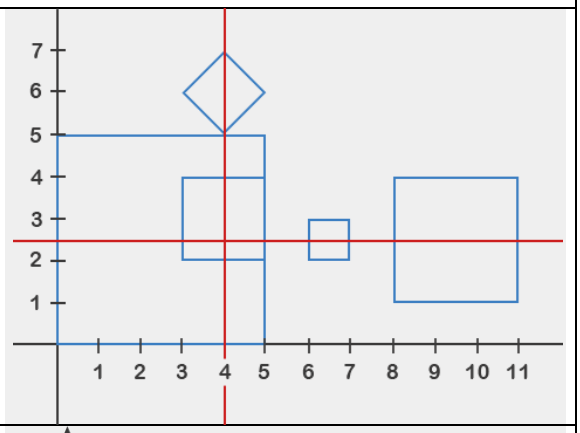
ყოველი კვადრატისთვის მისი კუთხეების კოორდინატები მთელი რიცხვებია და აკმაყოფილებენ $\{-10000 \le x1, y1, x2, y2 \le 10000\}$ უტოლობებს. კვადრატები არაგადაგვარებულია.

შემომავალი ფაილის ფორმატი

შესატანი მონაცემების squares.in ფაილის პირველი ხაზი შეიცავს ერთ მთელ რიცხვს N. შემდეგი N ხაზიდან თითოეულზე ოთხი მთელი რიცხვი $x1, y1, x2, y2$ წერია ამ მიმდევრობით.

გამომავალი ფაილის ფორმატი

გამოსატან მონაცემთა squares.out ფაილში გამოიტანეთ წრფეების მოდების ვარიანტების რაოდენობა. თუ იგი უსასრულოა, დაბეჭდეთ "-1" (ბრჭყალების გარეშე).

შემომავალი ფაილის მაგალითი (squares.in)	გამომავალი ფაილის მაგალითი (squares.out)	განმარტება
5 3 6 5 6 0 0 5 5 6 2 7 3 11 1 8 4 3 4 5 2	1	
5 2 1 5 6 6 1 11 0 9 -6 7 -4 6 -2 4 -4 1 -1 3 -3	0