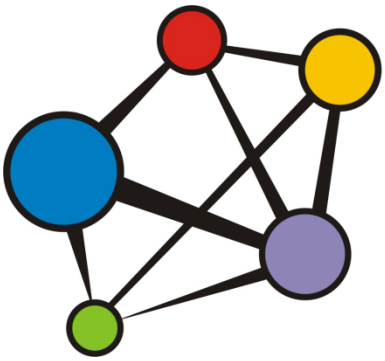

4ª Maratona de Programação

Facens 2008

PROBLEMA A: CONTADOR DE OBJETOS

Cor

Azul claro.

Arquivo

contador.[c/cpp/pas/java/cs/vb]

Descrição do problema

Uma fábrica de peças verifica sua produção diária contando, no final do turno, quantos itens saíram da linha de montagem. Como esse é um processo demorado e falho, a diretoria resolveu automatizar essa verificação através de uma técnica de processamento de imagens. Uma foto da esteira é tirada a cada suposto grupo de peças que passa e um programa de computador faz a contagem desses objetos.

A câmera utilizada já possui um algoritmo capaz transformar a imagem em uma matriz binária, ou seja, colocar 0 em cada *pixel* preto, que representa um *pixel* da esteira da linha de montagem, e 1 em cada *pixel* com cor, que representa um *pixel* da peça.

Considerando que uma peça é composta por um ou mais *pixels* em contato direto e que uma peça nunca toca a outra, ou seja, existe sempre um *pixel* da esteira entre quaisquer *pixels* de duas peças diferentes, desenvolva um algoritmo capaz de contar o número de peças de qualquer formato ao receber uma entrada que representa uma imagem binária de 5 por 5 *pixels*.

São apresentados dois exemplos nas figuras 1 e 2.

0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

Figura 1 – Imagem com 1 objeto

1	1	1	0	1
1	1	0	0	0
1	0	0	0	1
0	0	0	1	1
1	0	1	1	1

Figura 2 – Imagem com 4 objetos

Entrada

A entrada é constituída de vários casos de testes em que é passada uma matriz de 5 linhas e 5 colunas representando a imagem, composta por 0 e 1. Os 5 *pixels* de cada linha devem ser informados de uma vez, por exemplo, 00100. A execução deve ser interrompida quando for inserida uma linha contendo apenas um *pixel* igual a 0.

Saída

A saída, para cada imagem, é constituída do número de objetos encontrados nela.

Exemplos de testes

Entrada	**Saída**
00100 01110 11111 01110 00100 11111 00000 10101 00000 11111 0	1 5

PROBLEMA B: NÚMEROS AMIGÁVEIS

Cor

Vermelho.

Arquivo

numeros.[c/cpp/pas/java/cs/vb]

Descrição do problema

Para Pitágoras “O número é a causa e o princípio de tudo”. Esta afirmação sugere a existência de um princípio unificador do Universo, idéia que desempenhou um papel importante na filosofia grega. A mesma frase pode ainda servir para caracterizar a cultura Ocidental na sua relação com o número, ou melhor, dizendo, na sua obsessiva quantificação das qualidades. De fato, na ciência moderna, desde o Renascimento até a atualidade, é possível encontrar manifestações do espírito Pitagórico, das mais conscientes às mais ingênuas.

Os seguidores de Pitágoras não se limitaram a especular acerca da natureza e significado dos números e a estabelecer as suas propriedades místicas, eles produziram resultados matemáticos importantes perfeitamente integrados no conjunto da ciência grega. Pitágoras e os seus discípulos são mesmo considerados os iniciadores de uma área matemática, a Aritmética, hoje designada por Teoria de Números.

Os Pitagóricos ocupavam-se a descobrir as propriedades dos números, sem se preocupar com as suas aplicações, tal como faz hoje um investigador em teoria dos números.

Jamblico, um dos matemáticos pitagóricos influenciado pelo neo-platonismo, atribui a Pitágoras a descoberta dos números amigáveis.

Um par de números é amigável quando cada um deles é igual à soma dos divisores próprios do outro (divisores próprios de um número são todos os divisores inteiros positivos, exceto ele mesmo. Por exemplo, os divisores de 6 são 1, 2 e 3). A tarefa de sua equipe é escrever um programa onde, dado um par de números inteiros positivos, diga se é ou não amigável.

Entrada

A entrada é constituída de vários casos de testes, onde é dado um par de números inteiros positivos. Os números informados devem ser diferentes entre si e

nenhum deles poderá ser negativo. Caso seja dado como entrada os números -1 -1 o programa deverá ser encerrado.

Saída

Para cada teste de entrada, o programa deverá imprimir “numeros amigaveis" ou “numeros nao amigaveis" ou “numero invalido”.

Exemplos de testes

Entrada	**Saída**
10 14	numeros nao amigaveis
4 4	numero invalido
120 60	numeros nao amigaveis
284 220	numeros amigaveis
320 -1	numero invalido
-4 57	numero invalido
-1 -1	

PROBLEMA C: KNIGHT'S TOUR

Cor

Laranja.

Arquivo

knight.[c/cpp/pas/java/cs/vb]

Descrição do problema

O problema conhecido como Knight's Tour (percurso do cavalo) é um antigo quebra-cabeça definido sobre jogos de xadrez. Um percurso do cavalo em uma matriz qualquer é uma seqüência de movimentos da peça de xadrez conhecida como cavalo de tal forma que cada casa seja visitada exatamente uma vez.

Um cavalo no xadrez pode realizar um movimento de tal maneira que um dos eixos é alterado em duas posições e o outro eixo em uma posição, ao mesmo tempo.

A figura 1 mostra alguns percursos possíveis de um cavalo em um tabuleiro de xadrez. Com o objetivo de reduzir o tempo de processamento, neste problema consideraremos no máximo matrizes de 5 por 5.

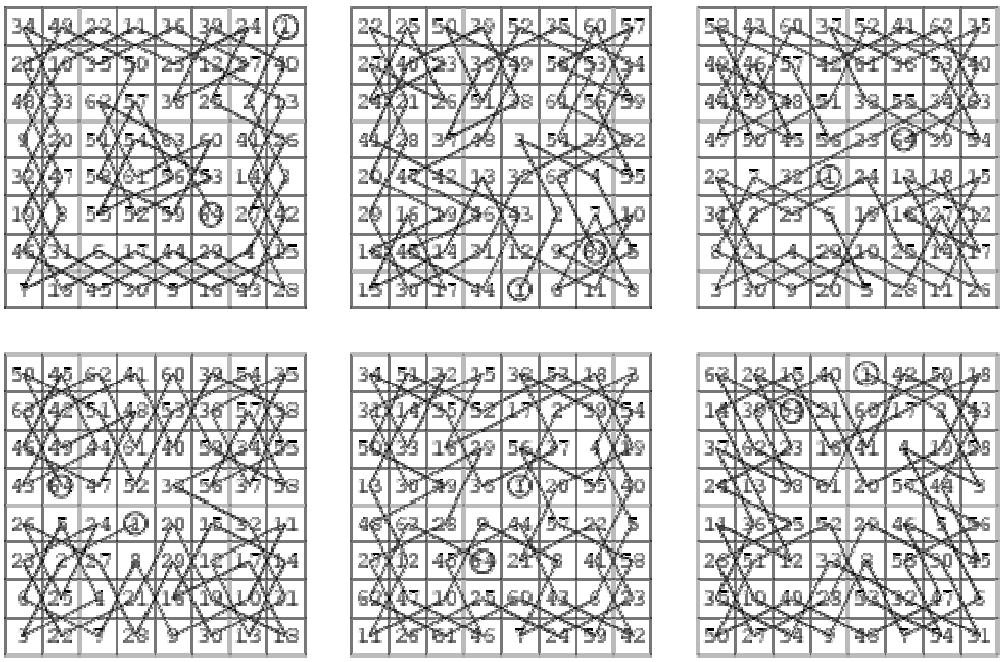

Figura 1 - exemplos de percursos de cavalo em um tabuleiro de xadrez

Entrada

A entrada é formada por vários casos de teste, sendo quatro números inteiros em cada teste. Os dois primeiros números *m* e *n* representam o tamanho do tabuleiro

do caso de teste, sendo 1 ≤ m ≤ 5 e 1 ≤ n ≤ 5. Os dois últimos números representam a coordenada (x,y) em que o cavalo iniciará o seu percurso. As coordenadas começam em (0,0). Nunca serão informadas coordenadas que extrapolem o tamanho do tabuleiro. Considere a origem e orientação do sistema de coordenadas como apresentado na figura 2.

Os casos de teste terminam quando os quatro números informados são zero.

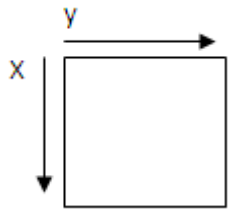

Figura 2 - eixo de coordenadas para a entrada

Saída

Para cada caso de teste será gerada uma saída com o número de percursos do cavalo possíveis para a entrada correspondente.

Exemplos de teste

Entrada	**Saída**
3 3 0 0	0
3 3 1 1	0
3 4 0 0	2
5 4 0 0	32
1 5 0 2	0
5 5 0 0	304
0 0 0 0	

PROBLEMA D: COMPRESSÃO DE DADOS

Cor

Verde.

Arquivo

compress.[c/cpp/pas/java/cs/vb]

Descrição do problema

A compressão de dados é o ato de reduzir o espaço ocupado por dados num determinado dispositivo. Essa operação é realizada através de diversos algoritmos de compressão, reduzindo a quantidade de bits para representar um dado, sendo esse dado uma imagem, um texto, ou um arquivo qualquer.

Comprimir dados destina-se também a retirar a redundância, baseando-se que muitos dados contêm informações redundantes que podem ou precisam ser eliminadas de alguma forma. Essa forma é através de uma regra, chamada de código ou protocolo, que, quando seguida, elimina os bits redundantes de informações, de modo a diminuir seu tamanho nos ficheiros. Por exemplo, a seqüência “AAAAAA” que ocupa 6 bytes, poderia ser representada pela seqüência “6A”, que ocupa 2 bytes, economizando 67% de espaço.

Além da eliminação da redundância, os dados são comprimidos pelos mais diversos motivos. Entre os mais conhecidos estão economizar espaço em dispositivos de armazenamento, como discos rígidos, ou ganhar desempenho (diminuir tempo) em transmissões.

Todo algoritmo de compressão de dados deve necessariamente ter um algoritmo relacionado capaz de descomprimi-los. Por exemplo, a seqüência compactada “6A” deve ser transformada de volta na seqüência “AAAAAA” para que possa ser utilizada.

Faça um programa capaz de comprimir e descomprimir dessa forma uma seqüência de caracteres, onde o primeiro caracter indica se é esperada a compressão ou descompressão dos dados.

Entrada

A entrada é constituída de vários casos de testes recebendo uma seqüência de até 100 caracteres onde o primeiro será necessariamente ‘C’ ou ‘D’, que indica se a seqüência deverá ser comprimida ou descomprimida, respectivamente, e deve ser

descartado ao realizar o processo de compressão ou descompressão. Não existem casos de teste com mais de 9 caracteres iguais em seqüência. A execução deve ser interrompida quando for inserida uma linha contendo apenas ‘C’.

Saída

Para cada seqüência recebida, exibir o resultado da compressão ou descompressão.

Exemplos de teste

Entrada	**Saída**
CAAAAAAXXXXX D6A5X CABCCCCDDDD D1A1B4C4D C	6A5X AAAAAAXXXXX 1A1B4C4D ABCCCCDDDD

PROBLEMA E: CERCA

Cor

Azul escuro.

Arquivo

cerca.[c/cpp/pas/java/cs/vb]

Descrição do problema

Seu avô possui uma fazenda e vai deixá-la de herança para você, mas com uma condição: como ele quer cercar toda a propriedade gastando o mínimo possível, você deve desenvolver um programa que permita a ele estudar diversas possibilidades de cercar a fazenda para ver em qual delas vai gastar o menos arame farpado.

A idéia, para economizar ao máximo, é contornar o tronco das árvores da fazenda com arame farpado ligando uma à outra, de forma a envolver toda a fazenda.

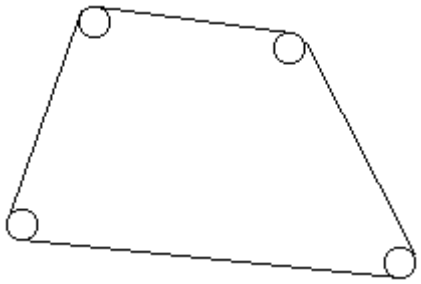

Figura 1 - exemplo

Desenvolva um programa que, para um número de árvores, seu raio e a posição do centro de cada uma delas, calcula o quanto de arame farpado será necessário para formar um polígono convexo envolvendo a fazenda. Considere que a posição das árvores é sempre informada na ordem em que elas são conectadas e, que da última árvore, o arame farpado será conectado de volta na primeira. O raio é igual para todas as árvores.

Entrada

A entrada é constituída de vários casos de testes recebendo primeiro o número n de árvores ($3 \leq n \leq 50$), depois o raio r em metros ($0.05 \leq r \leq 5$) e então a posição de cada árvore em seqüência no formato x y (x e y separados por espaço). O tamanho máximo do terreno é 10000 x 10000. A execução deve ser interrompida quando for inserido 0 no número de árvores.

Saída

O programa deve exibir o total de arame farpado que será utilizado para envolver o conjunto de árvores, mas, como só é possível comprar o arame farpado de metro em metro, o resultado deve ser exibido em número inteiro arredondado para cima se houver casa decimal (por exemplo, se o resultado for 14.28, exiba 15).

Exemplos de teste

Entrada	**Saída**
4 1.0 0.0 0.0 2.0 0.0 2.0 2.0 0.0 2.0 5 1.0 1.0 4.0 10.0 9.0 9.0 10.0 7.0 3.0 1.0 1.0 0	15 35

PROBLEMA F: GEOMETRICLÂNDIA

Problema adaptado do projeto eLane

Cor

Amarelo.

Arquivo

geometri.[c/cpp/pas/java/cs/vb]

Descrição do problema

O governo de um país resolveu criar uma cidade turística totalmente planejada chamada Geometriclândia, na qual as pessoas pudessem determinar o percurso a um determinado local de forma simples e rápida. Para isso, todas as ruas da cidade são orientadas na direção oeste-leste e norte-sul, e todos os quarteirões são do mesmo tamanho, formando uma *grid* regular. As interseções de ruas nessa cidade são identificadas pelo número da rua em cada direção, por exemplo, (2,4) representa a interseção da rua 2 na direção oeste-leste com a rua 4 na direção norte-sul.

Em 2008, um turista de sobrenome Descartes obcecado por geometria decide visitar Geometriclândia. Descartes deseja iniciar seu trajeto no ponto central da cidade, identificado pela interseção (0,0), depois quer caminhar uma quadra para norte, leste, sul ou oeste, seguindo essas prioridades. Se for para o norte, interseção (0,1), se for para sul (0,-1), se for para o leste (1,0) e se for para o oeste (-1,0). Ao chegar à cidade, o turista se anima ao ver a regularidade da cidade, e decide fazer seu percurso da seguinte forma: Partindo do ponto central (0,0), quer caminhar um quarteirão, seguindo a prioridade de direção, em seguida deseja caminhar mais dois quarteirões, só que não quer mais andar na mesma direção, nem quer voltar. No próximo segmento, ele caminha três quarteirões, depois quatro, cinco, e assim sucessivamente até chegar novamente ao ponto inicial, sempre mudando de direção a cada segmento.

Infelizmente, durante a visita a cidade, algumas interseções estão interrompidas devido a obras para melhoria das ruas. No entanto a prefeitura, através do seu secretário de transportes Gordnilys, sempre publica nos jornais locais as informações de quais interseções estão bloqueadas.

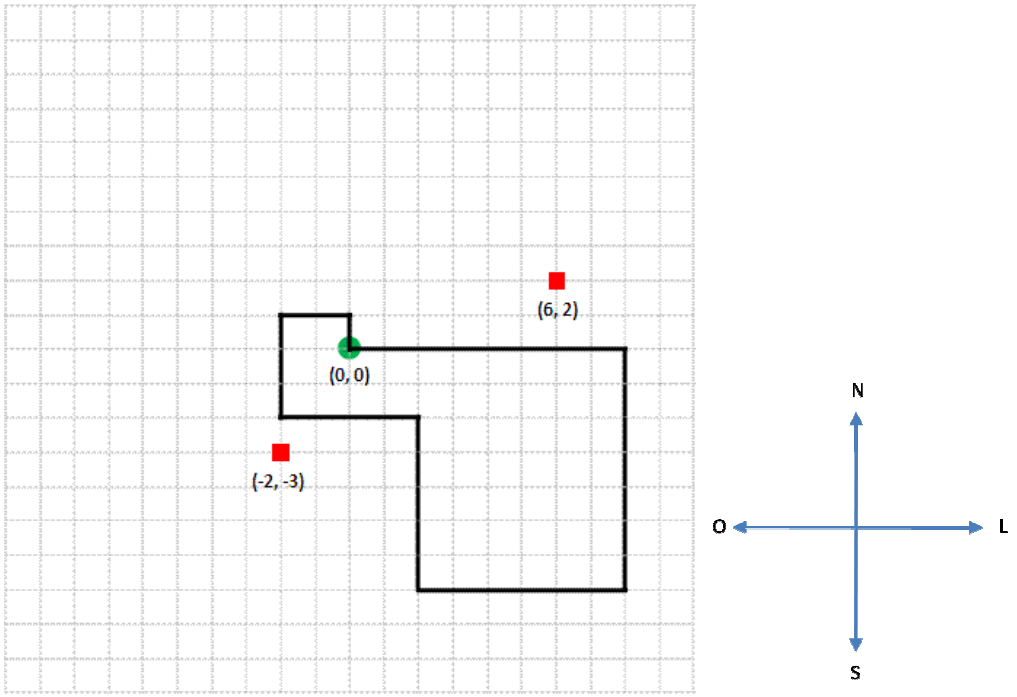

Figura 1 – entrada 1

Sua tarefa é fazer um programa que auxilie o turista a determinar uma rota, de modo que consiga fazer trajetos que o permitam partir do ponto central da cidade (0,0), percorrer uma determinada rota e retornar ao mesmo ponto central (0,0).

Entrada

A primeira entrada para o programa será um inteiro entre 1 e 25 indicando o comprimento do maior segmento que o turista percorrerá. Este é o comprimento do último segmento que leva o turista de volta ao ponto central. Na linha seguinte será informada a direção inicial do turista (N, S, L ou O). Em seguida será informado um inteiro entre 0 e 50 indicando o número de interseções que estão em obras. Nas demais linhas, serão informadas as coordenadas das interseções bloqueadas (x y), sendo x e y inteiros entre –50 e 50, mas diferente de (0, 0).

O Programa deverá ser encerrado quando o comprimento do último segmento for -1.

Saída

O programa deverá exibir na tela a seqüência das direções (N, S, L, O) que o turista deverá percorrer para sair do ponto central (0,0) e retornar ao mesmo. Caso não seja possível fazer o trajeto especificado, indicar que não é possível.

Exemplos de teste

Entrada	Saída
8 N 2 -2 -3 6 2 2 L 3 4 0 0 -5 2 1 7 O 1 1 0 -1	N O S L S L N O Nao existe rota possivel S L N L N O S