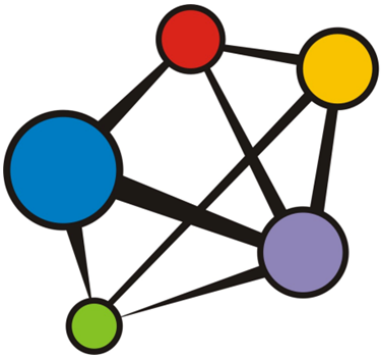


5ª Maratona de Programação

Facens 2009

PROBLEMA A: CONVERSOR DE BASE UNIVERSAL

Cor

Verde

Arquivo

`conversor.[c/cpp/pas/java/cs/vb]`

Descrição do problema

Um grupo de alienígenas viaja pela galáxia fazendo contato com outras civilizações. Eles trocam experiências e informações, muitas baseadas em números. O problema é que as civilizações dos planetas que eles visitam não utilizam as mesmas bases de numeração. Cada base de numeração utiliza um conjunto pré-determinado de símbolos para representar um determinado número. Por exemplo, na terra, a base de numeração mais utilizada é a base 10. Isso significa que há 10 algarismos (ou dígitos) diferentes, denotados pelos símbolos 0 a 9. Porém, em bases menores, utilizam-se menos algarismos, mas, em bases maiores, utilizam-se letras para completar os algarismos faltantes.

Na base 10, da direita para a esquerda: um algarismo na primeira posição vale por si próprio, na segunda, vale pela base, na terceira pela base ao quadrado, na quarta pela base ao cubo e assim sucessivamente. Por exemplo:

$8317 \rightarrow 8x10^3 + 3x10^2 + 1x10 + 7$

Uma vez que 10^0 é 1, podemos uniformizar a representação:

$8317 \rightarrow 8x10^3 + 3x10^2 + 1x10^1 + 7x10^0$

Em concordância com esta notação designamos as posições, da direita para esquerda, por posição 0, 1, 2 e assim sucessivamente. E designamos a potência respectiva por peso. Assim, dizemos que um algarismo na posição 0 tem peso 10^0, ou seja 1; na posição 1 tem peso 10^1, ou seja 10; na posição 2 tem peso 10^2 ou seja 100, etc. Genericamente, um dígito d_i na posição i tem peso 10^i, ou seja, vale

di_x10^i

Em geral, dado um número com os dígitos:

$$d_n\ d_{n-1}\ ...\ d_2\ d_1\ d_0$$

o seu valor numérico é:

$$d_n\ x10^n + d_{n-1}\ x10^{n-1} + ... + d_2\ x10^2 + d_1\ x10^1 + d_0\ x10^0$$

O conceito de representação numérica apresentado para a base decimal generaliza-se facilmente para outras bases. Em qualquer base um número representa-se como uma sequência de dígitos. Sendo B a base, um dígito d_i na posição i tem peso B^i ou seja vale:

$$d_i x B^i$$

Em geral, dado um número com os dígitos:

$$d_n\ d_{n-1}\ ...\ d_2\ d_1\ d_0$$

o seu valor numérico, dada a base B, é:

$$d_n\ xB^n + d_{n-1}\ xB^{n-1} + ... + d_2\ xB^2 + d_1\ xB^1 + d_0\ xB^0$$

Exemplos de conversão

10	2	3	5	8	13	16	20
0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1
2	10	2	2	2	2	2	2
3	11	10	3	3	3	3	3
4	100	11	4	4	4	4	4
5	101	12	10	5	5	5	5
6	110	20	11	6	6	6	6
7	111	21	12	7	7	7	7
8	1000	22	13	10	8	8	8
9	1001	100	14	11	9	9	9
10	1010	101	20	12	A	A	A
11	1011	102	21	13	B	B	B
12	1100	110	22	14	C	C	C
13	1101	111	23	15	10	D	D
14	1110	112	24	16	11	E	E
15	1111	120	30	17	12	F	F
16	10000	121	31	20	13	10	G
17	10001	122	32	21	14	11	H
18	10010	200	33	22	15	12	I
19	10011	201	34	23	16	13	J
20	10100	202	40	24	17	14	10
30	11110	1010	110	36	24	1E	1A
40	101000	1111	130	50	31	28	20
50	110010	1212	200	62	3B	32	2A
60	111100	2020	220	74	48	3C	30
70	1000110	2121	240	106	55	46	3A
80	1010000	2222	310	120	62	50	40
90	1011010	10100	330	132	6C	5A	4A
100	1100100	10201	400	144	79	64	50

Ao lado temos uma tabela de conversão exemplificando a representação de números em algumas bases. A primeira linha da tabela contém as bases, e as outras contêm números em cada base, correspondentes à base 10 (primeira coluna).

Sua função é construir um programa para os alienígenas que, dado um número em uma base informada, consiga convertê-lo em outra base especificada.

Entrada

A base de entrada (entre 2 e 30), o número na base de entrada e a base de saída (entre 2 e 30). Zero para terminar.

Saída

O número na base de saída.

Exemplos de testes

Entrada	**Saída**
10 10 2 3 222 20 16 FFF 5 3 10100 13 0	1010 16 112340 6C

PROBLEMA B: PROCESSAMENTO DE STRINGS

Cor

Vermelho

Arquivo

`pstrings.[c/cpp/pas/java/cs/vb]`

Descrição do problema

Uma empresa famosa por criar computadores utilizados para fins específicos, lançará seu novo modelo chamado SProc, o qual, segundo a companhia, será o principal computador para processamento de strings utilizados em criptografia e área afins.

Esse computador basicamente recebe strings de entrada e de acordo com programa carregado, fornece strings de saída. Para realizar esse processamento, o computador dispõe de apenas três instruções:

- Inserir caractere em uma posição específica;
- Substituir o caractere de uma posição específica por um caractere;
- Excluir um caractere de uma posição específica.

Os programas escritos para esse computador, utilizam instruções no formato CxiiE, onde:

- C é o código para instrução que será executada: I (Inserir), C (Substituir), ou D (Excluir)
- x é o caractere
- ii dois dígitos numéricos que representam a posição
- E é o caractere finalizador da transformação

Para entender como esse computador funciona, vejamos o exemplo a seguir que transforma a string ‘abcde’ na string ‘bchdt’.

```
      Abcde
Da01  bcde
Ih03  bchde
Ct05 bchdt
E     bchdt
```

Para comprovar a eficiência desse novo equipamento, seu programa deverá gerar o número mínimo de instruções para transformar uma string em outra, considerando as especificações desse computador.

Entrada

As entradas são constituídas de vários casos de teste, representados por duas strings de no máximo 55 caracteres cada, separadas por espaço, sendo a primeira a string de entrada e a segunda a string transformada.

Os casos de teste terminam quando existir somente um # na string de entrada.

Saída

Para cada entrada, o programa deverá fornece um inteiro como saída correspondente ao menor número possível de instruções necessárias para realizar a transformação.

Exemplo de Teste

Entrada	**Saída**
abcdefg bcdefg	1
qwertyu asdfgyu	5
plkueju ddhfjeu	6
scomp comps	2
#	

PROBLEMA C: A SOCIEDADE DO ANEL

Cor

Laranja

Arquivo

`umanel.[c/cpp/pas/java/cs/vb]`

Descrição do Problema

"*Um Anel para todos governar, Um Anel para encontrá-los, Um Anel para todos trazer e na escuridão aprisioná-los." – escritura no Um Anel revelado pelo fogo.*

Mestre Elrond reuniu os grandes líderes das raças da terra média para decidir o destino do terrível Um Anel, criado pelo malévolo Sauron, que o destino lançara na mão de um pequeno hobbit, Frodo. Após várias sugestões que não levaram a nenhuma conclusão, inclusive uma envolvendo uma catapulta do nobre Capitão Boromir, foi Gandalf The Coder, que dentre outras coisas já havida sido programador, que sugeriu:

"O Um Anel deve ser destruído nas Montanhas da Perdição em Mordor. Mas ninguém pode carregá-lo por muito tempo sem ficar louco. Devemos estabelecer um rodízio para carregar o Um Anel para Mordor."

Mestre Elrond imediatamente concordou, e para tal formou a Sociedade do Anel. Em sua infinita sabedoria, ele era capaz de dizer apenas olhando para um membro da sociedade quanto tempo ele poderia carregar o Um Anel sem perder completamente a sanidade. Assim ele fez um acordo com Gandalf The Coder.

"Eu irei lhe dizer quanto tempo cada membro da sociedade do anel pode carregar o anel sem ficar louco, e quanto tempo de descanso ele precisa para carregar novamente. Direi-lhe também o tempo total que levará para chegar as Montanhas da Perdição. Você deverá criar um programa usando magia negra que liste a ordem dos portadores do anel de forma que o Um Anel seja carregado pelo MENOR número de membros da sociedade que seja possível!"

E assim sua equipe interpreta, hoje, o papel do lendário Gandalf The Coder. Não perca tempo, o olho de Sauron está sobre vocês!

"*My preciouuussss!!!!" – ???*

Entrada

A entrada é constituída de vários casos de teste, pois Gandalf quer testar seu programa antes de apresentá-lo para Mestre Elrond. Cada entrada começa com um inteiro N (3<=N<=10) que representa o número de membros da comitiva e um inteiro H (100<=H<=2000) que representa o número de horas necessárias para chegar na Montanha da Perdição. Uma entrada com N=0 representa o final dos casos de teste. As N linhas seguintes possuem os nomes (nomes simples apelidos, sem espaços ou sobrenomes) seguidos de dois números X e Y (10<=X<=50 e 20<=Y<=100) que representam respectivamente quantas horas o portador pode ficar com o anel sem perder a sanidade e quantas horas ele precisa ficar sem o anel antes de poder carregá-lo novamente. O anel sempre deve começar com Frodo, que é quem está carregando-o atualmente. Sabe-se que ele está carregando o anel há 30 horas.

Saída

Cada caso de teste gera uma saída que é uma sequência separada por vírgula com a ordem em que o anel deve ser carregado de forma que o menor número possível de membros da comitiva tenha que carregar esse terrível fardo. No caso de mais de uma possibilidade de resposta, utilize a ordem em que os nomes foram apresentados na entrada. Caso seja impossível carregar o Um Anel para as Montanhas da Perdição com os membros informados, o programa deverá apresentar: “Precisamos de mais membros!”.

Exemplos de Testes

Entrada	**Saída**
3 100 Frodo 50 80 Boromir 10 50 Gandalf 30 120 5 500 Frodo 50 80 Boromir 10 50 Gandalf 30 120 Aragorn 20 130 Legolas 40 60 8 800 Frodo 50 80 Boromir 10 50 Gandalf 30 120 Aragorn 20 130 Legolas 40 60 Gimli 20 200 Samwise 45 90 Merryadoc 35 70 0	Precisamos de mais membros! Frodo, Aragorn, Legolas, Gandalf, Frodo, Legolas, Aragorn, Boromir, Gandalf, Frodo, Legolas, Aragorn, Boromir, Gandalf, Frodo, Legolas Frodo, Legolas, Samwise, Frodo, Legolas, Samwise, Frodo, Legolas, Samwise, Frodo, Legolas, Samwise, Frodo, Legolas, Samwise

PROBLEMA D: LABIRINTO DE CARACTERES

Cor

Azul Escuro

Arquivo

`labchar.[c/cpp/pas/java/cs/vb]`

Descrição do problema

Você faz parte de uma equipe de desenvolvimento de jogos casuais e tem como tarefa implementar um jogo chamado Labirinto de Caracteres, onde um tabuleiro de 4 por 4 é preenchido com caracteres e deve-se montar palavras utilizando as letras do tabuleiro de acordo com a seguinte regra: a partir de qualquer posição, pode-se escrever uma palavra de qualquer tamanho utilizando as letras do tabuleiro em seqüência tal que, a partir de uma letra, a próxima deve estar à esquerda, direita, acima ou abaixo dela, podendo inclusive passar por uma mesma letra mais de uma vez.

A figura 1 mostra um tabuleiro preenchido com caracteres.

a	b	r	a
a	d	a	c
b	r	r	d
r	a	a	s

Figura 1 – Tabuleiro

No tabuleiro da figura 1, podemos compor, por exemplo, a palavra “abracadabra” de quatro formas diferentes, pegando os caracteres na ordem (0 0), (0 1), (0 2), (0 3), (1 3), (1 2), (1 1), (1 0), (2 0), (2 1) e (3 1), ou (0 0), (0 1), (0 2), (0 3), (1 3), (1 2), (1 1), (1 0), (2 0), (3 0) e (3 1), ou (0 0), (0 1), (0 2), (1 2), (1 3), (1 2), (1 1), (1 0), (2 0), (2 1) e (3 1), ou ainda (0 0), (0 1), (0 2), (1 2), (1 3), (1 2), (1 1), (1 0), (2 0), (3 0) e (3 1). A palavra “radar” também pode ser composta de quatro formas diferentes, pegando os caracteres na ordem (0 2), (1 2), (1 1), (1 2) e (0 2), ou (0 2), (1 2), (1 1), (1 2) e (2 2), ou (2 2), (1 2), (1 1), (1 2) e (2 2), ou ainda (2 2), (1 2), (1 1), (1 2) e (0 2).

O que você deve fazer é desenvolver um programa que recebe os caracteres de um tabuleiro e palavras, para as quais deve dizer “sim” caso seja possível formá-las no tabuleiro seguindo a regra, ou “nao”, caso não seja possível. Você deve receber palavras até que a entrada seja 0.

Entrada

O programa tem como entrada as 4 linhas do tabuleiro e em seguida as palavras, até receber 0 para encerrar. Serão utilizadas apenas letras minúsculas (de ‘a’ à ‘z’).

Saída

Para cada palavra, o programa deve informar “sim” caso seja possível formá-la no tabuleiro, ou “nao” caso não seja.

Exemplos de teste

Entrada	**Saída**
Abra adac brrd raas abracadabra radar casa cara 0	sim sim nao sim

PROBLEMA E: PAZ VERDE

Cor

Azul Claro

Arquivo

`paz.[c/cpp/pas/java/cs/vb]`

Descrição do problema

Uma empresa X especializada em processamento de imagens de satélite procurou por sua equipe, composta de proeminentes engenheiros, para auxiliá-los na resolução de um problema. A empresa X possui uma rede de satélites que fotografam várias regiões do planeta, enviando informações a órgãos governamentais e algumas grandes corporações.

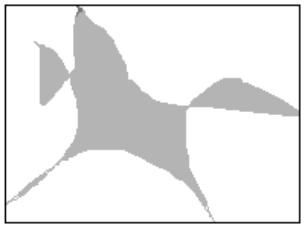
Recentemente ativistas de organizações verdes requisitaram algumas informações, mas a empresa X não quer comprometer sua clientela que obviamente não está interessada no meio ambiente. Sendo assim, ela resolveu lhe encaminhar alguns dados para que vocês processem e encaminhem para os ativistas.

Os ativistas, para realizar protestos ao redor do mundo, querem obter números de áreas verdes que, segundo eles, estão diminuindo drasticamente. A empresa X captará imagens de áreas verdes ao redor do mundo e lhe enviará um conjunto de pontos onde existem árvores.

Embora estes pontos possam ser usados para calcular uma área, os ativistas querem aumentar o drama sobre o problema, e decidiram transformar a área em um retângulo. Assim, sua tarefa é definir os pontos superior direito e inferior esquerdo do menor retângulo que contenha todos os pontos onde existem árvores. A figura a seguir mostra exemplos de áreas e dos retângulos correspondentes.

Área 1

Área 2



Área 3

Entrada

A entrada é constituída de vários casos de testes, sendo o primeiro número N de pontos que possuem árvores no mapa (10 <= N <=10000). Um valor de N igual a zero indica o final dos casos de teste. Em seguida, os N linhas representam estes N pontos, sendo que cada linha possui um par de coordenadas (X,Y) separados por espaço (0<=X<=100000, 0<=Y<=100000).

Saída

Para cada teste de entrada, o programa deverá mostrar as coordenadas X,Y do canto superior direito e inferior esquerdo do menor quadrado que contém todos os pontos informados.

Exemplos de Testes

Entrada	**Saída**
10 17879 24040 79125 52049 5973 80914 83125 17011 22586 98198 84568 97529 42698 41247 50646 38005 79360 89789 56233 94061 11 12406 26190 74508 25737 23200 24599 52351 52513 70964 58987 47985 28986 54445 79044 91500 25860 22321 66346 79218 6605 20479 42089 0	5973, 17011 84568, 98198 12406, 6605 91500, 79044

PROBLEMA F: PONTO FLUTUANTE

Cor

Amarelo

Arquivo

`pontof.[c/cpp/pas/java/cs/vb]`

Descrição do Problema

Entre 1970 e 1980 um grupo formado por cientistas e engenheiros de diferentes empresas de computação realizou um trabalho intenso na tentativa de encontrar um padrão de representação dos números, que deveria ser adotado por todas as indústrias na construção de seus computadores. A necessidade desta padronização tinha como principal objetivo uniformizar os resultados obtidos por um mesmo programa computacional executado em diferentes máquinas.

Esta discussão teve início em 1976, e este grupo de trabalho ficou conhecido como IEEEp754, pois foi organizado pelo *Institute for Electrical and Electronics Engineers* - IEEE. Entre os fabricantes estavam Apple, Zilog, DEC, Intel, Hewlett-Packard, Motorola e National Semiconductor. O prof. William Kahan liderava o grupo de cientistas e pelo trabalho desenvolvido neste projeto, recebeu o prêmio Turing Prize em 1989.

Este projeto tinha como metas principais:

- Especificar como representar os números em precisão simples e dupla;

- Padronizar o arredondamento nas operações neste sistema;

- Estabelecer critérios para padronizar situações como divisão por zero e operações envolvendo infinito.

Em 1985 o resultado deste trabalho foi publicado e ficou conhecido oficialmente como ANSI/IEEE Std 754-1985.

A base numérica no padrão IEEE754 é a binária. Nesse padrão em precisão simples, um número real é representado por 32 bits (4 bytes), sendo que:

- 1 bit é reservado para o sinal do número (0 - positivo ou 1- negativo);

- 8 bits são reservados para armazenar informações do expoente, que é um número inteiro;

- 23 bits são reservados para a mantissa;

No padrão IEEE754, a sequência de 8 bits armazena o número s = e + 127, onde e é o expoente.

Para a representação da mantissa, utiliza-se o sistema normalizado, no qual o primeiro dígito é sempre igual a 1 e, por esta razão não é armazenado, sendo denominado bit escondido. Esta normalização permite um ganho na precisão, pois podemos considerar que a mantissa é armazenada em 24 bits.

Exemplo:

Exemplo:

$(2,75)_{10} = (10,11)_2$ → (conversão não normalizada)

$(2,75)_{10} = (1,011 \times 2^1)_2$ → (conversão normalizada)

$s = 1 + 127 = 128_{10} = 10000000_2$

no padrão IEEE754: 0 10000000 01100000000000000000000

$(-0,25)_{10} = (0,001 \times 2^{-2})_2$ → (conversão não normalizada)

$(-0,25)_{10} = (1,0 \times 2^{-2})_2$ → (conversão normalizada)

$s = - 2 + 127 = 125_{10} = 01111101_2$

no padrão IEEE754: 1 01111101 00000000000000000000000

Entrada

A entrada é constituída de vários casos de testes, onde é dado um número na base 10. Caso seja dado como entrada o número 0, o programa deverá ser encerrado.

Saída

Para cada teste de entrada, o programa deverá imprimir a representação em binário no padrão IEEE 754 normalizado.

Exemplos de Testes

Entrada	**Saída**
50.8	0 10000100 10010110011001100110011
78.9	0 10000101 00111011100110011001101
-204.3	1 10000110 10011000100110011001101
-0.678	1 01111110 01011011001000101101000
-32.3	1 10000100 00000010011001100110011
0	