53

Problemas de Programação

Adonai Estrela Medrado
(adonaimedrado@hotmail.com)

Edição 2009
(Versão Preliminar – 20091209)

Sumário

Prefácio

A ideia inicial deste material foi construída quando eu tive o prazer de lecionar a disciplina *Laboratório de Programação II* para os alunos do curso de Ciências da Computação da Universidade Federal da Bahia. O desafio naquele momento era trabalhar problemas de programação típicos de competição.

Os frutos do trabalho me fizeram perceber que o modelo adotado poderia ser utilizado em outros cursos que objetivassem desenvolver a lógica ou aproximar o estudante de uma linguagem de programação. Percebi que os alunos gostavam da proposta e se sentiam desafiados e motivados.

A compilação apresentada aqui é um pouco daquilo que consegui construir na docência de dois semestres em cursos de programação e linguagem. Tenho em muito a agradecer aos alunos que participaram de todo o processo, elogiando, criticando, reportando problemas e sugerindo esclarecimentos.

Organizei os problemas em nível de dificuldade conforme a minha avaliação e julgamento. A sugestão é que o leitor leia primeiro o problema por inteiro e depois decida se está pronto ou não para desenvolver uma solução.

Salvador, 09 de novembro de 2009.

Nível iniciante

1 Problema da soma

Faça um programa capaz de solicitar um número N (1<N<1000) do usuário e ler N números inteiros. Após a leitura do último número deve-se informar:

- Na primeira linha a soma dos N números em decimal.
- Na segunda linha a soma em hexadecimal dos números pares informados.
- Na terceira linha a soma em octal dos números impares informados.

Assuma que todos os números fornecidos pelo usuário serão inteiros válidos e que as somas nunca serão superiores a um número inteiro de 32 bits.

Exemplo 1

Entrada

```
3
1
2
3
```

Saída

```
6
2
4
```

Exemplo 2

Entrada

```
10
1
2
3
4
5
6
7
8
9
0
```

Saída

```
45
14
31
```

2 Problema da média

Faça um programa capaz de solicitar um número inteiro N (1<N<1000) do usuário e ler N números inteiros. Ao final da leitura do último número, o programa deverá informar, com uma casa decimal, a média aritmética entre os N números que estejam contidos no intervalo fechado entre -1000 e 1000.

Assuma que todos os números fornecidos pelo usuário serão inteiros válidos e que a soma de todos os N nunca será superior a um número inteiro de 32 bits.

Exemplo 1

Entrada

```
2
3
4
```

Saída

```
3.5
```

Exemplo 2

Entrada

```
3
3
1001
4
```

Saída

```
3.5
```

Exemplo 3

Entrada

```
3
3
-1999
3
```

Saída

```
3.0
```

3 Problema do número espelho

Faça um programa que recebendo um número inteiro positivo N em hexadecimal seja capaz de verificar se este número em decimal é lido exatamente da mesma forma seja de frente para trás ou de trás para frente.

Caso positivo o programa deverá retornar uma linha com o caractere S, caso contrário o caractere informado deverá ser o N.

Exemplo 1

Entrada

```
1E1B9
```

Saída

```
S
```

Exemplo 2

Entrada

```
1E240
```

Saída

```
N
```

4 Problema da sequência de Fibonacci

A sequência de Fibonacci é construída de forma que cada termo é obtido pela soma dos dois termos anteriores. Por exemplo: 0, 1, 1, 2, 3, 5, 8, 13.

Faça um programa capaz de solicitar um número inteiro N (1<=N<=40) e informar os N primeiros elementos (um elemento por linha) da sequência de Fibonacci a partir do zero e com o segundo elemento 1.

Exemplo 1

Entrada

```
20
```

Saída

```
0
1
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
```

Exemplo 2

Entrada

```
1
```

Saída

```
0
```

5 Problema da conjectura de Goldbach

A conjectura de Goldbach (ainda não provada) diz que qualquer número par maior ou igual a 4 é a soma de dois números primos.

Faça um programa que, recebendo um número P par (2<=P<=4294967294), seja capaz de retornar dois número inteiros correspondentes aos dois números primos cuja soma seja igual ao número par P.

Considere que:

- Os valores de saída devem ser ordenados em ordem crescente.
- Existindo mais de uma combinação possível, retorna-se aquela cujo primeiro valor seja o menor.
- Não existindo valores (parabéns! você foi o primeiro no mundo que provou que a conjectura é falsa!) retorne -1.

Lembre-se: número primo é todo número inteiro maior que 1 que somente é divisível por si próprio e pela unidade.

Exemplo 1

Entrada

```
720
```

Saída

```
11
709
```

Exemplo 2

Entrada

```
666
```

Saída

```
5
661
```

6 Problema do quadrado gêmeo das partes

Existem alguns números que têm uma propriedade bastante interessante, observe:

100
    10+0=10 10*10=100
2025
    20+25=45 45*45=2025
3025
    30+25=55 55*55=3025
9801
    98+1=99 99*99=9801
10000
    100+0=100 100*100=10000
88209
    88+209=297 297*297=88209
494209
    494+209=703 703*703=494209
998001
    998+1=999 999*999=998001

Os números que têm esta propriedade são conhecidos como número de Kaprekar.

Cada um dos números apresentados tiveram seus algarismos decompostos de tal forma que a soma das partes elevado ao quadrado era igual ao número original.

Faça um programa capaz de ler e identificar se um determinado número N (1<=N<=100.000.000) possui ou não esta propriedade. Caso positivo, o programa deverá retornar uma única linha com o valor 1, caso contrário deve-se retornar uma linha com valor 0.

Exemplo 1

Entrada

```
60481729
```

Saída

```
1
```

Exemplo 2

Entrada

```
60481728
```

Saída

```
0
```

Exemplo 3

Entrada

```
300814336
```

Saída

```
1
```

Exemplo 4

Entrada

```
88200
```

Saída

```
0
```

7 Problema da soma reservada[1]

Para este problema considere que a soma reservada de N + M é dada por Rev(Rev(N) + Rev(M)), sendo que Rev(X) é uma função que retorna X com os algarismos na ordem inversa. Por exemplo, Rev(5789) = 9875.

Faça um programa capaz de receber dois números inteiros e informar a soma reservada do mesmo.

Exemplo 1

Entrada

```
1000
1200
```

Saída

```
22
```

Exemplo 2

Entrada

```
5
130000
```

Saída

```
63
```

Exemplo 3

Entrada

```
5
130000
```

Saída

```
63
```

1 Problema adaptado de http://www.topcoder.com/stat?c=problem_statement&pm=9925&rd=13508.

8 Problema dos casais

Em um bar, os homens recebem cartão de identificação com números ímpares e as mulheres cartões com números pares.

Um animador contratado para animar a programação do dia deseja saber se existe uma proporção de um-para-um entre homens e mulheres.

Você deve fazer um programa para fazer esta verificação.

O programa receberá um número N (1<=N<=1000) com o número de cartões distribuídos e uma lista com N números inteiros positivos (todos maior ou igual a 1 e menor ou igual a 500) cada um representando o número de um cartão. A saída deverá ser uma única linha com o caractere **S** caso exista a proporção ou com o caractere **N** caso contrário.

Exemplo 1

Entrada

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

Saída

```
S
```

Exemplo 2

Entrada

```
8
1 3 5 7 9 11 13 15
```

Saída

```
N
```

Exemplo 3

Entrada

```
4
1 2 3 4
```

Saída

```
S
```

Exemplo 4

Entrada

```
6
1 2 3 4 8 7
```

Saída

```
S
```

9 Problema dos sucessores

Faça um programa que solicite números inteiros I (-4000<=I<=4000) enquanto I for diferente de zero. Quando I for zero o programa deve imprimir todos os sucessores inteiros imediatos de cada I informado.

Observe que neste problema não há um limite para a quantidade de números I informados.

Exemplo

Entrada

```
1
2
3
4
5
6
7
8
9
10
0
```

Saída

```
2
3
4
5
6
7
8
9
10
11
```

10 Problema do quadrado perfeito

Um número inteiro positivo N é um quadrado perfeito se existe um número K tal que K*K=N.

Faça um programa que receberá uma quantidade indefinida de números inteiros positivos J (-10000<=J<=10000) até que J seja zero. A saída do programa deverá ser a quantidade de quadrados perfeitos informados.

Exemplo 1

Entrada

```
2
3
4
5
6
7
8
9
0
```

Saída

```
2
```

Exemplo 2

Entrada

```
4
9
16
144
0
```

Saída

```
4
```

11 Problema da competição alien

Os aliens do planeta Yks fazem uma competição para ver quem consegue escreve um texto com o maior número de palavras com N (1<=N<=10) caracteres.

Faça um programa para automatizar o cálculo e contar quantas palavras com N caracteres existem no texto digitado. O programa deverá receber o número N e um texto com até 250 caracteres e deverá dar como saída a quantidade de palavras com N sílabas que o texto contém.

Atenção: para resolver este problema você necessita saber que na escrita do povo de Yks não é utilizado espaço nem mesmo para separar palavras, ao invés disto, qualquer caractere que não seja uma letra *minúscula* entre a e z será considerado separador de palavra.

Exemplo 1

Entrada

```
3
sss.aasd.sss...
```

Saída

```
2
```

Exemplo 2

Entrada

```
2
123123kj244141hm22242420+-
xx....sadf...dfadfa..sd.
```

Saída

```
4
```

Exemplo 3

Entrada

```
5
asdfg..lk..ujjmjh...lkium
```

Saída

```
2
```

Nível intermediário

1 Problema das sequências alternadas

Faça um programa que

1. Leia um número N.
2. Leia N seqüências de 5 números inteiros.
3. Imprima as N seqüências alternando a ordem de crescente para decrescente.

Se desejar pode-se assumir que o N máximo será 1000.

A entrada e a saída devem ser conforme o exemplo abaixo.

Exemplo

Entrada

```
4
87
55
34
0
53
21
8
61
82
9
80
78
1
10
99
4
78
43
71
23
```

Saída

```
0 34 53 55 87
82 61 21 9 8
1 10 78 80 99
78 71 43 23 4
```

Variação 1

Resolva o mesmo problema só que assuma que o número de elementos em cada seqüência pode variar.

- Antes de cada sequência o usuário informá qual o número de elementos da sequência.

Variação 2

Resolva o mesmo problema só que desta vez imprima na saída a sequência:

- Em ordem crescente, quando nenhum elemento da sequência for primo.

- Em ordem decrescente, quando um elemento da sequência for primo.
- Na mesma ordem em que foi digitada, quando mais de um elemento da sequência for primo.

Variação 3

Resolva o mesmo problema alternando a ordem crescente e decrescente, só que faça a ordenação de acordo com o peso P definido abaixo.

- O peso P de cada elemento será a quantidade de divisores inteiros positivos que ele possui.

2 Problema da letra mais frequente[2]

Faça um programa capaz de identificar a(s) letra(s) mais frequente(s) em uma cadeia de caracteres.

A entrada será uma cadeia de caracteres sem espaços de no máximo 1000 caracteres. A saída deverá ser a(s) letra(s) mais frequente(s) seguida por sua porcentagem com duas casas decimais.

Deve-se desconsiderar diferenças de maiúsculas e minúsculas.

Qualquer outro caractere que não seja uma letra de A a Z deverá ser desconsiderado no cálculo da porcentagem e na contagem. A saída deve ser dada em letras minúsculas.

Exemplo 1

Entrada

```
aabc
```

Saída

```
a 50.00%
```

Exemplo 2

Entrada

```
aabcc
```

Saída

```
a 40.00%
c 40.00%
```

Exemplo 3

Entrada

```
aabcc
```

Saída

```
a 40.00%
c 40.00%
```

Exemplo 4

Entrada

```
asl;dzc]ewa;d]sd.vcxhkjasdfa]]bkjolnn
opuibuiopjl;
```

Saída

```
a 9.76%
```

2 Problema adaptado de: http://www.topcoder.com/stat?c=problem_statement&pm=9905&rd=13507.

```
d 9.76%
```

3 Problema do giro da palavra

Para este problema, considere que uma cadeia é rotacionada quando uma quantidade N de caracteres é movida do final para o início da cadeia. Por exemplo, as rotações possível da palavra **linguagem** são:

```
inguageml
nguagemli
guagemlin
uagemling
agemlingu
gemlingua
emlinguag
mlinguage
linguagem
```

Observe que a própria palavra **linguagem** também é considerada como uma rotação.

Faça um programa para verificar se uma cadeia S2 pode ser obtida pela rotação da cadeia S1.

O programa deve ler a cadeia S1 e a cadeia S2 e retornar 1 caso S2 possa ser obtida através da rotação de S1. Caso não seja possível, deve-se retornar 0.

Exemplo 1

Entrada

```
programacao
programacao
```

Saída

```
1
```

Exemplo 2

Entrada

```
programacao
rogramacaop
```

Saída

```
1
```

O mesmo resultado é esperado para: ogramacaopr, gramacaopro, ramacaoprog, amacaoprogr, macaoprogra, acaoprogram, caoprograma, aoprogramac, oprogramaca e macaoprogra.

Exemplo 3

Entrada

```
papagaio
opapagai
```

Saída

```
1
```

O mesmo resultado é esperado para: apagaiop, pagaiopa, agaiopap, gaiopapa, aiopapag, iopapaga e papagaio.

Exemplo 4

Entrada

```
papagaio
opapagia
```

Saída

```
0
```

4 Problema da palavra mágica

Uma palavra P de tamanho K (2<=K<=100) é mágica se ela tem um número par de letras e se ao ordenarmos em ordem alfabética as K/2 primeiras letras obtém-se as K/2 letras finais.

Exemplo de palavras mágicas:

- asas.
- gogo.
- gluglu.
- chocho.
- adfsadfs.
- aaaa.

Exemplo de palavras que não são mágicas:

- xixi (ordenando as letras xi em ordem alfabética ficaria ix que não é igual a xi).
- xoxo (ordenando as letras xo em ordem alfabética ficaria ox que não é igual a xo).
- muximuxi (ordenando as letras muxi em ordem alfabética ficaria imux que não é igual a muxi).
- asdffdsa (ordenando as letras asdf em ordem alfabética ficaria adfs que não é igual a fdsa).
- aaaaa (não é mágica, pois tem um número ímpar de caracteres).

Faça um programa que recebendo uma linha com uma palavra seja capaz de identificar se ela é ou não uma palavra mágica. Caso seja, mostre uma linha com o caractere **S**, caso contrário mostre uma linha com o caractere **N**.

Exemplo 1

Entrada

```
popo
```

Saída

```
N
```

Exemplo 2

Entrada

```
gugu
```

Saída

```
S
```

Exemplo 3

Entrada

```
tutu
```

Saída

```
S
```

Exemplo 4

Entrada

```
aaaaa
```

Saída

```
N
```

Exemplo 5

Entrada

```
tatatt
```

Saída

```
S
```

Exemplo 6

Entrada

```
gramados
```

Saída

```
N
```

5 Problema do conjunto e seus elementos únicos

Por definição, um conjunto não pode ter elementos repetidos.

Faça um programa capaz de ler um número inteiro N (1<=N<=1000) e N inteiros K (-1000<=K<=1000).

A saída deverá ser um conjunto formado pelos K inteiros. Os elementos deverão ser exibidos em ordem crescente.

Exemplo 1

Entrada

```
10
10
10
9
9
8
8
7
7
6
6
```

Saída

```
6
7
8
9
10
```

Exemplo 2

Entrada

```
5
-1
-1
-1
-1
-1
```

Saída

```
-1
```

Exemplo 3

Entrada

```
5
-123
123
999
1000
```

```
-1000
```

Saída

```
-1000
-123
123
999
1000
```

6 Problema da moda

A moda é o conjunto formado pelos elementos com a maior frequência em uma amostra. Por exemplo, na amostra (1,2,3,3,3,4,4,4,4,5) a moda é {4}, na amostra (1,1,1,2,2,2,3,4,5,5,6) a moda é {1,2}.

Faça um programa capaz de calcular a moda de uma amostra de dados K.

A entrada será um número indefinido de elementos de K um por linha.

A saída deverá ser os elementos K pertencentes à moda em ordem crescente um em cada linha.

K terá como elementos números inteiros positivos entre 0 e 256, sendo que o número zero identifica o fim da entrada de dados.

Exemplo

Entrada

```
256
1
5
91
83
256
4
5
60
7
8
29
42
8
42
24
5
42
256
0
```

Saída

```
5
42
256
```

Variação 1

Considere que K terá como elementos números inteiros entre -2147483648 e 2147483647, sendo que o número zero identifica o fim da entrada de dados.

Exemplo

Entrada

```
1
5
```

```
91
83
2
467
4
467
5
60
7
8
467
29
42
8
42
24
5
42
0
```

Saída

```
5
42
467
```

Variação 2

K terá como elementos cadeias de caracteres sem espaço com no máximo 50 caracteres, sendo que a cadeia **"0"** (sem as aspas) identifica o fim da entrada de dados.

7 Problema da simplificação das frações

Considere uma fração no formato N/M sendo N inteiro (0<=N<=10000) e M inteiro (1<=M<=10000).

Faça um programa capaz de simplificar uma fração neste formato tal que o novo numerador e o novo denominador sejam primos entre si.

O programa receberá um conjunto de testes composto por um número indefinido de linhas. Cada linha conterá os valores de N e M separados por um espaço. O final da entrada será indicado por N=M=0.

A saída do programa deverá ser a fração simplificada na mesma ordem da entrada uma em cada linha separando-se o numerador e o denominador por espaço.

Exemplo

Entrada

```
0 1
1 1
2 2
2 4
9 3
25 625
0 0
```

Saída

```
0 1
1 1
1 1
1 2
3 1
1 25
```

8 Problema do colecionador de moedas[3]

Um colecionador de moedas tem o objetivo de colecionar o maior número possível de moedas diferentes (qualidade ou valor não são importantes).

Faça um programa que, recebendo um vetor V com o valor das moedas e um vetor C com as moedas que o colecionar possui, informe qual o número máximo de moedas diferentes que o colecionador pode conseguir.

Considere que o colecionador só consegue dinheiro para comprar outras moedas se vender as moedas que possui.

A entrada do vetor V será dada da seguinte forma:

- Uma primeira linha com um Nv inteiro (1<=Nv<=100) identificando a quantidade de elementos de V.
- Nv linhas com números inteiros Kv (1<=Kv<=50). Cada elemento Kv representará o valor de mercado de cada moeda cujo nome é a posição de K a partir do zero (veja exemplo).

A entrada do vetor C se dará logo após à entrada do vetor V e será da seguinte forma:

- Uma primeira linha com um Nc inteiro (1<=Nc<=Nv).
- Nc linhas com números inteiros Kc (0<=Kc<=Nv-1). Cada elemento Kc representará o nome de uma moeda que o colecionador possui.

Exemplo

Entrada

```
5
9
4
7
8
17
2
4
0
```

Neste exemplo:

- Nv=5.
- Nc=2.
- Conforme o vetor V, o valor da moeda "0" é $9, da moeda "1" é $4, da moeda "2" é $7, da moeda "3" é $8, da moeda "4" é $17.
- Conforme o vetor C, o colecionador possui a moeda "0" e "4", que custam respectivamente $9 e $17 conforme consta no vetor V.

Saída

```
3
```

3 Problema adaptado de http://www.topcoder.com/stat?c=problem_statement&pm=9933&rd=13506.

9 Problema da transmissão de rádio

Deseja-se criar um programa capaz de identificar uma mensagem inimiga que está sendo transmitida em ondas de rádio acima de 100Mhz. O programa de computador espião Kni já captou a transmissão e é necessário que seja construído outro software capaz de interpretar e extrair a mensagem.

O Kni dá como saída uma cadeia como a seguinte:

```
90c87esd67uj,./';*&^120lin87uj101gu87km102a77jh150gem..&
```

Onde, da esquerda para direita:

- 90 é a frequência em Mhz.
- c é o código lido na frequência de 90Mhz.
- 87 é a frequência do próximo código.
- esd é o código lido na frequência de 87Mhz
- 67 é a frequência do próximo código.
- uj é o código lido na frequência de 67Mhz
- ,./';&^ foi uma interferência que ocorreu quando lia-se o código da frequência de 67Mhz.
- ...

Assim, no fragmento acima, a mensagem transmitida acima de 100Mhz foi: linguagem. Pois, **lin** foi transmitido a 120Mhz **gu** a 101Mhz, **a** a 102Mhz e **gem** a 150Mhz.

Construa um programa capaz de recebendo uma cadeia de no máximo 250 caracteres retornar a mensagem transmitida acima de 100Mhz.

Considere que:

- a freqüência estará sempre entre 1 e 200Mhz.
- toda a interferência deverá ser ignorada. Deve-se considerar interferência todo caractere diferente de uma letra ou um número.
- não existirá espaços na cadeia de entrada (produzida pelo Kni).
- o tamanho máximo da mensagem será de 100 caracteres.

Exemplo 1

Entrada

```
90c87esd67uj,./';*&^120lin87uj101gu87km102a77jh150gem..&
```

Saída

```
linguagem
```

Exemplo 2

Entrada

```
*(12*23qualquer130i120n87j102t87ejh104er*&^_)(105n7k122e33kw140t**
```

Saída

```
internet
```

10 Problema da idade em dias

Faça um programa capaz de informar a idade em dias com base na data de nascimento de um ser.

Para resolver este problema você precisa saber que:

- Setembro, abril, junho e novembro têm 30 dias, todos os outros meses tem 31 exceto fevereiro que tem 28, exceto nos anos bissextos nos quais ele tem 29.
- Todo ano par divisível por 4 é um ano bissexto (1992 = 4*468 então 1992 será um ano bissexto, mas 1990 não é um ano bissexto).
- A regra acima não é válida para anos de virada de século. Estes anos devem ser divisíveis por 400 para serem anos bissextos, todos os outro não são. Assim, o ano 1700, 1800, 1900 e 2100 não são bissextos, mas 2000 é bissexto.

Não é permitido o uso de funções de data da linguagem, exceto as necessárias para obter a data atual.

O programa deverá ler um dado por linha correspondente ao dia, mês e ano de nascimento (inteiros positivos válidos para seus respectivos valores). A saída será constituída de uma única linha com a quantidade de dias que o ser viveu.

Na linguagem C, pode-se obter o dia, mês e ano corrente através do código a seguir (necessário a inclusão de **time.h**):

```
time_t agora_t = time(NULL);
struct tm *agora =
localtime(&agora_t);
int dia = agora->tm_mday;
int mes = agora->tm_mon + 1;
int ano = agora->tm_year + 1900;
```

Exemplo 1

Entrada

```
21
9
1981
```

Saída

```
10006
```

(Considerando dia 12/02/2009)

Exemplo 2

Entrada

```
31
7
1981
```

Saída

```
10058
```

(Considerando dia 12/02/2009)

Exemplo 3

Entrada

```
1
3
2004
```

Saída

```
1809
```

(Considerando dia 12/02/2009)

Outros exemplos

Você pode gerar mais casos de teste para a data atual em: http://www.peterrussell.com/age.php

11 Problema da separação das sílabas (versão light)

Geralmente um processador de textos utiliza algum algoritmo para fazer a hifenização das palavras. Neste algoritmo são consideradas posições onde a palavra pode ser divida. Por exemplo, a palavra **programação** têm as seguintes possibilidades para a divisão silábica:

```
pro-gramação
progra-mação
programa-ção
```

Faça um programa que, recebendo uma cadeia de caracteres (máximo de 50 caracteres) no formato abaixo, mostre todas as divisões silábicas possíveis (uma por linha) em ordem de preferência da palavra representada.

Formato de entrada e instruções gerais

<letra$_0$><dígito$_0$><letra$_1$><dígito$_1$>...<letra$_n$><dígito$_n$>

Onde:

- <letra$_n$> é uma letra minúscula do alfabeto.
- <dígito$_n$> pode ser omitido e é um dígito de inteiro positivo no intervalo fechado entre 1 e 9.

Caso <dígito$_n$> seja um número par o ponto não pode sofre divisão silábica, caso seja impar poderá sofrer a divisão. Valores maiores são aqueles que indicam pontos onde há a preferência pela divisão.

Caso exista mais de um ponto com a mesma preferência, todas as opções devem ser exibidas, uma opção linha mostrando primeiro aquelas que o hífen aparece mais a esquerda.

Exemplo 1

Entrada

```
p2r4o5g2r4a5ma7ção
```

Saída

```
programa-ção
pro-gramação
progra-mação
```

Exemplo 2

Entrada

```
pro7gra9ma7ção
```

Saída

```
progra-mação
pro-gramação
programa-ção
```

Exemplo 3

Entrada

```
i2n3c4o8n2s7t6i9t8u7c2i4o5n6a4l
```

Saída

```
inconsti-tucional
incons-titucional
inconstitu-cional
inconstitucio-nal
in-constitucional
```

12 Problema da codificação da string

Um grupo de estudantes de Ciência da Computação de primeiro semestre decidiu criar um algoritmo para comprimir e codificar um textos. A técnica desenvolvida é bem simples.

Primeiro um texto de T caracteres (0<T<1000) contendo somente letras maiúsculas e números é convertido para uma cadeia contendo exclusivamente letras maiúsculas da seguinte forma:

- as letras diferentes de Z são copiadas na mesma ordem em que aparecerem.
- as letras Z são convertidas para a cadeia ZZ na mesma ordem em que aparecerem.
- caso um número seja encontrado ele é convertido em um par de letras sendo a primeira um Z e a segunda um A, caso o número seja 0, um B, caso o número seja 1, um C caso o número seja 2 e assim por diante até J caso o número seja 9.

A cadeia resultante desta primeira etapa é recodificada para eliminar repetições de caracteres. O procedimento consiste em uma variação da compressão RLE, assim:

- sendo A uma letra e K o número vezes sucessivas (sempre menor que 1000) que letra A é repetida, o resultado a ser exibido será KA, exceto se K for igual a 1 quando neste caso a saída será A.

Faça um programa capaz de codificar e de decodificar esta string.

Formato de entrada

- Uma linha contendo a letra C ou a letra D. Caso C então a próxima linha representará uma cadeia a ser codificada, caso D então a próxima linha deverá ser decodificada.
- Uma linha contendo a cadeia a ser processada.

Formato de saída

- Uma linha contendo o resultado do processamento ou, caso a cadeia de entrada não for válida, uma linha em branco.

Exemplo 1

Entrada

```
C
ABC
```

Saída

```
ABC
```

Exemplo 2

Entrada

```
C
AAAAAAAABC
```

Saída

```
8ABC
```

Exemplo 3

Entrada

```
C
ABCZ
```

Saída

```
ABC2Z
```

Exemplo 4

Entrada

```
C
ABCZ9Z
```

Saída

```
ABC3ZJ2Z
```

Exemplo 5

Entrada

```
D
2Z
```

Saída

```
Z
```

Exemplo 6

Entrada

```
C
1234567890
```

Saída

```
ZBZCZDZEZFZGZHZIZJZA
```

Exemplo 7

Entrada

```
D
Z
```

Saída

```

```

Cadeia inválida, deve-se exibir uma linha em branco.

13 Problema do professor de matemática caxias

O professor de matemática Saixac é considerado pelos colegas e alunos como bastante caxias e exigente. Ele estuda e ensina o uso do parênteses, colchetes e chaves.

Na sua prova, ele solicitou aos alunos que escrevessem e lhe enviassem por e-mail expressões de até 200 caracteres utilizando-se das regras que ensinou em sala de aula.

Nesta avaliação ele não estava interessado em analisar as operações matemáticas em si, mas o uso dos símbolos modificadores de ordem da resolução.

A instrução foi para que os alunos escrevessem expressões utilizando-se de parênteses e/ou colchetes e/ou chaves. As exigências eram:

- Todo parênteses, colchetes ou chaves que for aberto deve ser fechado.
- Nenhum parênteses, colchetes ou chaves pode ser fechado sem antes ser aberto.
- Uma expressão agrupada por chaves só poderá existir se dentro dela existir pelo menos uma expressão agrupadas por colchetes ou pelo menos uma outra expressão agrupada por chaves que atenda ao pré-requisito anterior.
- Uma expressão agrupada por colchetes só poderá existir se dentro dela existir pelo menos uma expressão agrupadas por parênteses.

As expressões numéricas em si podem ou não existir.

Faça um programa para verificar as expressões imprimindo "1" caso for correta "0" caso errada.

As expressões terão até 200 caracteres, serão informadas uma por linha e não haverá espaço entre seus caracteres. O resultado poderá ser dada logo após um retorno de carro ("enter").

A entrada terminará quando uma expressão com somente um número zero for informada.

Exemplo 1

Entrada

```
(1+2)+(3+4)
[(1+2)+(3+4)]
{[(1+2)+(3+4)]}
{{[(1+2)+(3+4)]}}
0
```

Saída

```
1
1
1
1
```

Exemplo 2

Entrada

```
()
()()
[()()]
{[()()]}
```

```
{[()]}
0
```

Saída

```
1
1
1
1
1
```

Exemplo 3

Entrada

```
((
))()
[()()[
{[()))]}
{[()]{
{]()))]}
}[()]{
0
```

Saída

```
0
0
0
0
0
0
0
```

Exemplo 4

Entrada

```
(1+2)(
)1+2)+(3+4)
](1+2)+(3+4)[
}[(1+2)+(3+4)]{
0
```

Saída

```
0
0
0
0
```

Exemplo 5

Entrada

```
{(1+2)+(3+4)}
([3+4])
[3+4]
[({1+2}+{3+4})]
0
```

Saída

```
0
0
0
0
```

14 Problema da competição de ciclismo

Um sensor de velocidade para bicicletas pode funcionar através de um dispositivo no garfo que fecha o circuito na presença de um imã preso ao aro.

Desta forma, a cada giro completo da roda, um software pode calcular a distância percorrida pelo ciclista e sua velocidade tendo como base a circunferência da roda e no tempo gasto no giro.

O proprietário de uma academia deseja implementar uma maneira para que seus clientes possam simular uma competição.

Sua tarefa é fazer um protótipo deste sistema para demonstrar sua viabilidade.

Você receberá um número N (2<=N<=10) representando o número de competidores. Para cada competidor lhe serão fornecidas 3 linhas.

- Linha 1:
  - valor da circunferência da roda em milímetros.
- Linha 2:
  - quantidade G (1<=G<=100) de giros completos dados pela roda.
- Linha 3:
  - G valores V (1<=V<=5000) inteiros identificando o tempo em milissegundos que a roda demorou para executar a volta completa.

Cada competidor é numerado a partir do 1 na ordem em que suas informações foram inseridas.

As entradas fornecidas consideram que:

- no momento em que um vencedor atinge a linha de chegada todos os outros param de pedalar.
- não há empates; só existe um vencedor: aquele que atingir a maior distância.
- podem existir várias desistências durante o percurso por isto alguns competidores podem parar de pedalar antes de o vencedor atingir a linha de chegada.

O software deverá informar como saída duas linhas, a primeira com o número do competidor vencedor, a segunda com a velocidade média em km/h deste competidor arredondada para uma casa decimal.

Informações úteis:

- Um milissegundo é igual a 0.000000278 hora.
- Um milímetro é igual a 0.000001 kilômetro.

Você pode conferir seu cálculo de conversão de mm/ms para km/h em http://www.convertunits.com/from/millimeters+per+millisecond/to/kilometre/hour.

Exemplo

Entrada

```
5
2011
10
500 450 300 200 150 100 150 160 120 130
```

```
2121
12
440 400 330 300 250 240 250 150 120 100 120 100
2121
2
1440 900
2000
8
400 420 300 290 200 180 150 160
1800
10
520 400 390 290 220 280 150 160 100 120
```

Saída

```
2
32.7
```

Para clarificação e testes, segue abaixo, em ordem decrescente, os valores de distância e velocidade média de cada competidor no exemplo acima.

- Competidor 2) 25452 32.7
- Competidor 1) 20110 32.0
- Competidor 5) 18000 24.6
- Competidor 4) 16000 27.4
- Competidor 3) 4242 6.5

15 Problema do baile de casais

Um organizador de um baile descobriu que as mulheres que iriam frequentar à festa eram bastante exigentes e conservadoras.

Cada uma delas só aceita dançar com um homem que seja no mínimo um ano mais velho que ela. Por exemplo, suponha Maria de 54 anos, ela só aceita dançar com João se ele no mínimo tiver 55 anos.

Para piorar sua situação o organizador descobriu que se uma mulher ficar sem par de dança, todas as outras se recusam a dançar.

Ele resolveu que precisava de um programa para verificar a lista de convidados e verificar se a festa iria ou não ser um sucesso. Sucesso representaria o fato de todas as mulheres terem par (mesmo que algum homem fique sem dançar), a fracasso significaria que alguma mulher ficou sem companheiro de dança.

Verificando a lista de convidados ele percebeu uma coisa bem interessante: todos os homens tinham sempre idades impares e as mulheres idades pares.

Faça um programa que, recebendo um número N de convidados (2<=N<=1000) e N idades I (18<=I<=100), seja capaz de informar se a festa será um sucesso (**S**) ou um fracasso (**F**).

Exemplo 1

Entrada

```
2
18 21
```

Saída

```
S
```

Exemplo 2

Entrada

```
2
21 22
```

Saída

```
F
```

Exemplo 3

Entrada

```
7
21 22 24 55 23 32 57
```

Saída

```
S
```

Possível solução: (22,23),(24,55),(32,57). O homem com idade de 21 ficará sem par, porém não há problema nisto.

Exemplo 4

Entrada

```
8
21 31 33 35 37 39 41 43
```

Saída

```
S
```

Explicação: Não há mulheres para ficarem desacompanhadas. Pela regra descrita a festa seria um sucesso.

Exemplo 5

Entrada

```
9
21 31 33 35 37 39 41 43 58
```

Saída

```
F
```

Explicação: A única mulher de 58 anos não dançará com nenhum homem da festa já que todos são mais jovens que ela.

Exemplo 6

Entrada

```
4
22 32 36 48
```

Saída

```
F
```

16 Problema do TMA

O tempo médio de atendimento (TMA) de uma central de teleatendimento é calculado pela média dos tempos de todos os atendimentos realizados em um período.

O gerente de uma central deseja contratá-lo como analista chefe, porém, para testar suas habilidades de programador, lhe propôs o desafio de calcular o tempo médio de atendimento com base em um arquivo texto.

O formato do arquivo é bastante simples. Cada linha do arquivo contém dois valores inteiros. O primeiro representa o momento de início do atendimento, o segundo o momento de fim de atendimento.

Cada momento é medido em minutos a partir do início do horário do expediente.

Faça um programa que leia este arquivo (que estará no mesmo diretório do seu programa com o nome entrada.txt) e exiba na saída padrão o mínimo, o máximo, a moda e a média com uma casa decimal (um valor em cada linha, nesta ordem) do tempo de atendimento.

Algumas considerações:

- Cada momento está no intervalo fechado entre 0 e 1000.
- O arquivo não está ordenado e terá no mínimo uma linha.
- Se não existir moda ou se existir mais de um tempo de atendimento que seja a moda, imprima -1.
- O separador dos decimais da moda deve ser de acordo com as configurações regionais do computador.

Exemplo

Arquivo entrada.txt

```
5 12
6 20
7 8
6 98
11 14
8 25
98 100
56 79
45 98
12 55
1 3
4 6
7 10
10 13
13 16
```

Saída

```
1
92
3
17,9
```

17 Problema do tesouro real[4]

Era uma vez um reino onde a matemática era um grande problema. Quando um posto para trabalhar nas finanças reais necessitava ser preenchido apresentava-se aos candidatos o seguinte enunciado de problema:

Dado dois vetores de inteiros positivos A e B com N elementos e a função S abaixo, reordene os números em A de forma que o valor de S seja o menor possível. Não é permitido mudar a ordem dos elementos de B.

$S=A_0*B_0+A_1*B_1+...+A_n*B_n$

O reino precisa de um programa para conferir as respostas, assim, pede-se um programa que dado um N (2<=N<=100), um vetor A e um vetor B (com N elementos dentro do limite fechado de 1 e 1000) seja capaz de retornar o menor valor possível de S.

Exemplo 1

Entrada

```
3
1
10
8
7
56
13
```

Saída

```
230
```

Possível resposta: A={10,1,8}.

Exemplo 2

Entrada

```
5
12
3
53
8
91
74
2
3
4
64
```

Saída

```
1123
```

Possível resposta: A={3,91,53,12,8}.

4 Problema adaptado de RoyalTreasurer de http://www.topcoder.com/stat?c=problem_statement&pm=10007&rd=13695.

Exemplo 3

Entrada

```
5
1
1
1
6
0
2
7
8
3
1
```

Saída

```
18
```

Possível resposta: A={1,1,0,1,6}.

Exemplo 4

Entrada

```
9
5
15
100
31
39
0
0
3
26
11
12
13
2
3
4
5
9
1
```

Saída

```
528
```

Possível resposta: A={3,0,0,39,31,26,15,5,100}.

18 Problema da escrita no celular

Uma das formas pelas quais se pode escrever letras utilizando-se as teclas numéricas de um celular é pressionando-as repetidas vezes até que a letra correspondente seja exibida.

Os fabricantes geralmente usam a seguinte associação tecla numérica/letras:

- 2: a, b, c
- 3: d, e, f
- 4: g, h, i
- 5: j, k, l
- 6: m, n, o
- 7: p, q, r, s
- 8: t, u, v
- 9: w, x, y, z

Por exemplo, para se obter a letra **b** deve-se pressionar duas vezes a tecla 2, para a letra **m** uma vez tecla 6 e assim por diante.

Faça um programa que, recebendo uma palavra P (máximo de 50 caracteres), seja capaz de informar quantas vezes e quais teclas terão que ser pressionadas para obtenção da palavra.

As teclas devem ser informadas na ordem para a formação correta da palavra.

Conforme o exemplo abaixo, cada tecla deve ser precedidas por um sharp (#). A quantidade de vezes deve vir logo em seguida à tecla separada por um igual (=) conforme exemplo.

Exemplo 1

Entrada

```
internet
```

Saída

```
#4=3
#6=2
#8=1
#3=2
#7=3
#6=2
#3=2
#8=1
```

Exemplo 2

Entrada

```
preconceber
```

Saída

```
#7=1
#7=3
#3=2
#2=3
```

```
#6=3
#6=2
#2=3
#3=2
#2=2
#3=2
#7=3
```

Exemplo 3

Entrada

```
zunzunzum
```

Saída

```
#9=4
#8=2
#6=2
#9=4
#8=2
#6=2
#9=4
#8=2
#6=1
```

Variação 1

Faça o caminho inverso da dificuldade anterior, ou seja, recebendo a saída anterior como entrada, dê a entrada.

Você receberá um número N identificando quantas teclas serão informadas.

Repeite o formato apresentado.

Exemplo 1

Entrada

```
9
#9=4
#8=2
#6=2
#9=4
#8=2
#6=2
#9=4
#8=2
#6=1
```

Saída

```
Zunzunzum
```

19 Problema do *checksum*

Os algoritmos de verificação tipo *Checksum* trabalham executando operações sobre um conjunto de dados e são úteis para verificar se este conjunto de dados permanece íntegro.

Este problema considera o seguinte algoritmo de *checksum*:

- Para cada byte B na posição P (sendo a primeira posição zero, a segunda 1, etc.):
  - Para cada B que estiver em uma posição P impar, deve-se contar a quantidade de bits 0 em B e acumular esta quantidade em um inteiro W.
  - Para cada B que estiver em uma posição P par, deve-se contar a quantidade de bits 1 em B e acumular esta quantidade em um inteiro W.

O resultado deste algoritmo é W que deverá suportar um valor máximo de 32bits.

Atenção: este algoritmo é apenas um exemplo para fins didáticos e não deve ser utilizado em aplicações reais.

Faça um programa que receberá na entrada padrão um número indefinido de cadeias de caracteres (K), cada uma sem espaço e com máximo de 200 caracteres). Cada K representará o caminho de um arquivo para o qual deve ser calculado o checksum segundo algoritmo descrito acima. A entrada termina quando for encontrada a cadeia "0" (sem as aspas).

A saída deverá ser dada na saída padrão e deverá ser o número W descrito no algoritmo acima.

Exemplo

Entrada

```
exemplo1.in
exemplo2.in
exemplo3.in
exemplo4.in
exemplo5.in
exemplo6.in
0
```

Saída

```
400
402
207
42146
96072
36937
```

Os arquivos deste exemplo podem ser baixados em:

- http://www.adonaimedrado.pro.br/wiki/documentos/professor/problema_checksum_entradas.zip.

20 Problema das operações com conjuntos

Um professor de matemática deseja um programa para ensinar as operações com conjuntos a seus alunos. O programa deve ser simples, porém completo, aceitando as operações de intersecção, união e diferença entre dois conjuntos quaisquer. Na entrada dos dados o usuário fornecerá as informações, uma em cada linha, na seguinte ordem:

1. A operação, sendo i=intersecção, u=união e d=diferença.
2. O número de elementos do primeiro conjunto.
3. Os elementos do primeiro conjunto (um elemento por linha).
4. O número de elementos do segundo conjunto.
5. Os elementos do segundo conjunto (um elemento por linha).

A saída deverá ser os elementos do conjunto resultado **em ordem crescente** e separados por espaço. Não exiba elementos duplicados.

Garantias e considerações:

- As entradas sempre serão válidas e os elementos serão únicos dentro de cada conjunto.
- Um conjunto poderá ter de 0 até 100 elementos.
- Cada elemento do conjunto será um número entre 0 e 99.
- Sendo o primeiro conjunto informado A e o segundo B a diferença sempre deverá ser calculada assumindo-se o resultado da operação A - B.

Lembrete:

- Intersecção: é o conjunto formado pelos elementos comuns aos dois conjuntos.
- União: é o conjunto formado por todos os elementos comuns e não comuns.
- Diferença: é o conjunto formado pelos elementos do primeiro conjunto que não pertencem ao segundo.

Exemplo 1

Entrada

```
u
4
1
2
4
5
2
3
4
```

Saída

```
1 2 3 4 5
```

Exemplo 2

Entrada

```
i
4
```

```
1
2
4
5
2
3
4
```

Saída

```
4
```

Exemplo 3

Entrada

```
d
4
1
2
4
5
2
3
4
```

Saída

```
1 2 5
```

21 Problema do decifrador de senhas

Um determinado banco fornece aos seus clientes, além da senha numérica, uma senha de 3 letras que deve ser utilizada no caixa eletrônico na realização de operações. O software do caixa eletrônico, para dificultar a ação de ladrões, solicita a entrada da senha de 3 letras em um "teclado" gráfico especial gerado aleatoriamente todas as vezes e exibido na tela do monitor.

Neste teclado, cada uma das teclas contém uma quantidade de 4 letras distintas. Para digitar a senha, o usuário deve escolher em ordem as teclas que contenham as letras da sua senha. Por exemplo, sendo a senha ABC e existindo as teclas MNAU, ILKC, JIYB e OKLM o usuário deve pressionar as teclas MNAU, JIYB e ILKC nesta ordem.

Um contraventor observou determinados usuários do banco durante várias operações no caixa eletrônico e anotou as teclas pressionadas durante a operação. O banco, que grava a cada operação as teclas pressionadas pelo usuário, foi informado da atuação de tal criminoso e deseja fazer um programa para saber de quais clientes é necessário bloquear a senha.

As senhas bloqueadas serão aquelas que, a partir das teclas pressionadas, o contraventor é capaz de saber, sem sombra de dúvida, qual é a senha de 3 letras do cliente.

Faça um programa que receberá 9 cadeias de caracteres do mesmo tamanho. Cada bloco de 3 cadeias conterá as letras das "teclas" gráficas pressionadas pelo usuário com relaçao a uma letra da senha.

O primeiro bloco de 3 cadeias irá conter as teclas pressionadas com relação à primeira letra da senha, o segundo bloco as teclas pressionadas com relação à segunda e o terceiro bloco as teclas pressionadas com relação à terceira letra da senha.

O programa deverá retornar 1 caso seja possível identificar (sem sombra de dúvidas) quais as letras da senha e 0 caso contrário.

Exemplo 1

Entrada

```
MNAU
LACN
AKIM
IKJH
ILKC
JKBM
MUDH
ADIK
HUDM
```

Saída

```
1
```

(Senha: AKD)

Exemplo 2

Entrada

```
HJMI
OPJH
AJMO
MIUN
ABCI
IJKB
ONHK
LOUN
UODN
```

Saída

```
0
```

(Senha: JI(O ou N))

22 Problema da operação entre números binários

Faça um programa capaz de operar dois números de 0-255 informados em binário.

A resposta deve ser dada também em binário.

Deve-se aceitar os operadores abaixo:

```
+,-,*,/,%
```

Exemplo 1

Entrada

```
+
00000001
00000011
```

Saída

```
00000100
```

Exemplo 2

Entrada

```
-
00000010
00000001
```

Saída

```
00000001
```

Exemplo 3

Entrada

```
*
00000001
00000011
```

Saída

```
00000011
```

Exemplo 4

Entrada

```
%
00010100
00000011
```

Saída

```
00000010
```

23 Problema da sopa de letras na formação de palavras

Faça um programa capaz de aceitar uma string S e um grupo de dez palavras P. Para cada palavra digitada deve-se verificar a possibilidade de formá-la com as letras da string informada.

Considere que:

- Todas as palavras serão digitadas em letras minúsculas.
- S e cada elemento de P terão tamanho máximo de 100 caracteres.

Para cada elemento de P caso seja possível formá-lo com os caracteres de S o programa deverá responder OK, caso contrário -1.

A resposta pode ser dada imediatamente quando um elemento de P for digitado.

Exemplo 1

Entrada

```
sopadeletras
sopas
tras
traz
trazer
talo
proda
adelok
kulad
mea
lopr
```

Saída

```
OK
OK
-1
-1
OK
OK
-1
-1
-1
OK
```

Variação 1

Considere a mesma situação anterior só que assuma que quando um caractere C for utilizado na formação de um elemento possível de P, C deverá ser descartado.

Entrada

```
sopadeletrasparaformarpalavras
palavra
palavras
sapo
dele
akmopr
```

```
rrstafmp
juaarso
orsaaaa
ors
aaarr
```

Saída

```
OK
-1
OK
OK
-1
OK
-1
-1
OK
-1
```

24 Problema do número binariamente contido

Faça um programa que receba um número N de 32 bits, um número M também de 32 bits e um número K. A saída deverá ser 1 caso os K últimos bits de M se encontram em algum local da sequência de bits de N. Deve-se mostrar zero caso contrário.

Exemplo 1

Entrada

```
448
3
3
```

Sendo N=448, M=3, K=3.

Saída

```
1
```

Exemplo 2

Entrada

```
448
5
3
```

Saída

```
0
```

25 Problema das placas com anagrama perfeito

No país Abfu as placas dos veículos só possuem 4 dígitos onde apenas duas letras podem aparecer X e Y (não há números na placa). Os emplacamentos são agendados e só é feito um por dia (mas sempre há emplacamento no dia). As placas são disponibilizadas seguindo a ordem alfabética, assim XXXX foi a primeira placa, XXXY a segunda e YYYY será a última. Um pai possui dois filhos Pedro e Ordep, ele deseja um programa que, recebendo uma placa, diga para quantos dias depois ele terá de agendar o emplacamento para conseguir a placa com as letras na ordem inversa. Caso não seja possível deve-se retornar -1.

Exemplo 1

Entrada

```
XXYX
```

Saída

```
2
```

O objetivo é conseguir emplacar o veículo com a placa XYXX, para tanto deve-se deixar o emplacamento para exatos 2 dias depois, assim “pularia-se” a placa XXYY.

Exemplo 2

Entrada

```
XXXX
```

Saída

```
-1
```

Exemplo 3

Entrada

```
XXXY
```

Saída

```
7
```

Nível avançado

1 Problema da sequência de algarismos agrupados com ordenação

Faça um programa capaz de agrupar os algarismos de uma sequência de N números inteiros maiores que nove. O agrupamento será feito pela casa das unidades. Os algarismos agrupados e a sequência de saída devem aparecer ordenados conforme o exemplo abaixo.

Considere que:

- O número de sequências será o primeiro número a ser informado.
- Cada número pode ter até 25 algarismos.
- Não haverá agrupamento com mais de 100 algarismos.

Exemplo

Entrada

```
10
56348631
897434
4321895
98746321
695413245
129078
89078905432
89021
78903278
89278
```

Saída

```
1.022333445666788899
2.0034578899
4.34789
5.11223344456899
8.0012223777788999
```

2 Problema do professor de terceiro ano

No final de cada aula, Um professor de terceiro ano solicita aos alunos que respondam uma questão estilo vestibular-UFBA.

Estas questões funcionam da seguinte maneira:

- A questão consta de até 7 proposições numeradas como 1, 2, 4, 8, 16, 32 e 64.
- A resposta da questão é a soma do número das proposições julgadas como verdade, por exemplo, caso o aluno julgue as questões 2 e 16 como corretas e as demais como falsas, a resposta do aluno deve ser 18 (2+16).

Para verificar o quanto os alunos compreenderam o assunto, o professor verifica o gabarito e acha o mínimo, o máximo e a média do número das proposições julgadas corretamente como **verdade**.

Como ele perde muito tempo fazendo estes cálculos, você foi solicitado para ajudá-lo.

Faça um programa que receberá os seguintes dados um em cada linha conforme o exemplo abaixo:

1. um número inteiro G (0<=G<=99) que representa o gabarito da questão
2. um número inteiro N (1<=N<=1000) que representa o número de alunos que responderam a questão.
3. N linhas no formato: <nome> <valor_reposta>, onde <nome> representa o nome do aluno sem espaço com até 50 caracteres e <valor_resposta> é um valor entre 0 e 99.

A saída do programa deve ser o mínimo, o máximo e a média (arredondada para uma casa decimal) do número das proposições julgadas corretamente como **verdade**. Cada valor deve ser informado em uma linha. Os valores mínimos e máximos devem ser seguidos pelo nome dos alunos em ordem alfabética que obtiveram os respectivos valores conforme exemplo abaixo.

Exemplo 1

Entrada

```
18
4
aluno1 1
aluno2 2
aluno3 3
aluno4 4
```

Saída

```
0 aluno1 aluno4
1 aluno2 aluno3
0.5
```

Exemplo 2

Entrada

```
18
3
```

```
aluno1 0
aluno2 2
aluno4 18
```

Saída

```
0 aluno1
2 aluno4
1.0
```

Exemplo 3

Entrada

```
33
1
aluno1 32
```

Saída

```
1 aluno1
1 aluno1
1.0
```

3 Problema da prefeitura em crise

Uma prefeitura está em período de baixa arrecadação e não tem condições de fazer o pagamento integral da sua folha de pagamento.

O prefeito decidiu que iria pagar com a verba disponível somente aquelas pessoas que recebem os menores salários, pois, segundo sua interpretação, elas seriam as mais necessitadas.

Você foi contratado como consultor externo para fazer um programa que seja capaz de informar quais os funcionários que receberão salarial.

Algumas considerações:

- Não havendo possibilidade de atender todas as pessoas de uma mesma faixa salarial, a prioridade é para aquelas com maior idade. Caso ainda haja empate, a prioridade é para aquelas cujo nome venha primeiro na ordem alfabética.
- Não há homônimos na prefeitura.
- A idade sempre será um número inteiro e estará no intervalo fechado entre 18 e 100.
- O salário sempre será um número inteiro e estará no intervalo fechado entre 1 e 1000.
- As entradas não estão ordenadas.

Entrada

A entrada consistirá de vários casos de teste. Cada caso de teste é constituído por:

1. Uma linha contendo um inteiro K (0<=K<=1000000) e um inteiro J (0<=J<=1000). K representa a verba disponível para o pagamento da folha. J representa o número de servidores.
2. As próximas J linhas serão apresentadas no formato:

```
<nome> <idade> <salario>
```

O nome terá no máximo 50 caracteres e não conterá espaços. A entrada termina quando K e J forem informados como zero.

Saída

A saída deverá seguir rigorosamente o exemplo abaixo, indicando para cada K o nome dos funcionários que receberão salário. Assim, para cada caso de testes deve ser gerada a seguinte saída:

- A primeira linha deve conter a cadeia "Teste N", onde N é representa o número do caso de teste começando em 1.
- As linhas seguintes são os nomes dos funcionários em ordem alfabética.
- A última linha deve ser em branco.

Exemplo

Entrada

```
5 5
Z 29 10
```

```
D 30 10
A 30 10
E 68 12
C 25 5
15 5
Z 29 10
D 30 10
A 30 10
E 68 12
C 25 5
20 5
Z 29 10
D 30 10
A 30 10
E 68 12
C 25 5
25 5
Z 29 10
D 30 10
A 30 10
E 68 12
C 25 5
35 5
Z 29 10
D 30 10
A 30 10
E 68 12
C 25 5
50 5
Z 29 10
D 30 10
A 30 10
E 68 12
C 25 5
0 0
```

Saída

```
Teste 1
C

Teste 2
A
C

Teste 3
A
C

Teste 4
A
C
D

Teste 5
A
C
D
Z
```

```
Teste 6
A
C
D
E
Z
```

4 Problema da separação das sílabas

Geralmente um processador de textos utiliza algum algoritmo para fazer a hifenização das palavras. Neste algoritmo são consideradas posições onde a palavra pode ser divida. Por exemplo, a palavra **programação** têm as seguintes possibilidades para a divisão silábica:

```
pro-gramação
progra-mação
programa-ção
```

O divisor silábico do BrOffice está sob a Licença Pública Geral Menor versão 2.1 (LGPLv2.1) e funciona "com base no léxico do VERO, através de análise combinatória, extraindo-se os casos reais e descartando-se as condições inexistentes."[5]

Segundo os desenvolvedores do VERO[6], ele funciona conforme o algoritmo de Frank M. Liang da seguinte forma:

1. são carregadas uma série de partículas indicando pontos onde a divisão é possível e onde a divisão deve ser evitada.
2. quando uma palavra precisa ser divida as partículas são utilizadas e processadas para identificar os pontos de divisão.

Cada partícula possui o seguinte formato:

```
.<letra0><digito0><letra1><digito1>...<letran><dígiton>.
```

Onde:

- .(caractere ponto) caso presente no início da partícula indica que ela deve ocorrer no início da palavra.
- <letra$_n$> é uma letra minúscula do alfabeto.
- <dígito$_n$> pode ser omitido e é um dígito de inteiro positivo no intervalo fechado entre 1 e 9.
- .(caractere ponto) caso presente no final da partícula indica que ela deve ocorrer no final da palavra.

Caso <dígito$_n$> seja um número par, o ponto não é preferível para divisão silábica, caso seja impar o ponto é preferível. Quanto maior o valor, maior a preferência pela divisão (caso impar) ou pela não divisão (caso par).

O processamento é realizando sobrepondo-se as partículas na palavra considerando-se o maior dígito em caso de partículas que tratem de uma mesma subcadeia da palavra. Observe abaixo o exemplo dos desenvolvedores do VERO[7] para o processamento da palavra **silábicas**. As partículas pertinentes são: s2i, i3l2á, l4á, á1b2, 3b2i, i1c4, 3c2a, 2s.

5 Arquivo README_hyph_pt_BR.txt em http://www.broffice.org/files/hyph_pt_BR-203.zip.
6 *Ibdem*.
7 *Ibdem*.

```
s i l á b i c a s
s2i
    l4á
  i3l2á
    l4á
      á1b2
       3b2i
          i1c4
           3c2a
               2s.
------------------
s2i3l4á3b2i3c4a2s   <--- Resultado
```

Faça um programa que, recebendo um número N, um conjunto R de N partículas e uma palavra P, mostre:

1. o resultado do processamento.
2. todas as divisões silábicas possíveis (uma por linha) em ordem de preferência.
3. a separação das sílabas das palavras.

Considere que:

- N será um número inteiro tal que 1<=N<=1000.
- cada partícula do conjunto R tem até 10 caracteres.
- a palavra P tem até 100 caracteres.

Exemplo 1

Entrada

```
8
s2i
i3l2á
l4á
á1b2
3b2i
i1c4
3c2a
2s.
silábicas
```

Saída

```
s2i3l4á3b2i3c4a2s
si-lábicas
silá-bicas
silábi-cas
si-lá-bi-cas
```

Exemplo 2

Entrada

```
5
1c2i
ê2n1c2
i1ê
```

```
i2a
n3c42
ciência
```

Saída

```
ci1ê2n3c4i2a
ciên-cia
ci-ência
ci-ên-cia
```

Variação

Resolva o mesmo problema só que desta vez, ao invés de você receber uma lista de partículas para cada palavra, você deverá ler uma lista fixa no arquivo http://www.adonaimedrado.pro.br/wiki/documentos/outros/hyph_pt_BR.dic[8].

Nesta dificuldade, o programa só receberá a palavra como entrada e retornará além do solicitado na dificuldade 1 as partículas de fato utilizadas no processamento da palavra.

Exemplo 1

Entrada

```
programa
```

Saída

```
1g4r2
1m2a
a1m2a
o3g2
o3g2
r2o
r4a
pr2o3g4r4a1m2a
pro-grama
progra-ma
pro-gra-ma
```

Exemplo 2

Entrada

```
computador
```

Saída

```
1d2o
1p2u
1t2a
2m3p4
4r.
a1d2o
o2m1p2u
o4r.
```

8 Este arquivo também é do projeto VERO do BrOffice (http://www.broffice.org/verortografico), porém sofreu uma pequena alteração para este problema (exclusão da primeira linha).

```
u3t2
co2m3p4u3t2a1d2o4r
com-putador
compu-tador
computa-dor
com-pu-ta-dor
```

Exemplo 3

Entrada

```
universidade
```

Saída

```
1d2a
1d2e
1n2i
1s2i
1v2e
a1d2e
e2r3s4i
i3d2
i3v2
r1s2i
u1n2i
u1n2i3v2e2r3s4i3d2a1d2e
uni-versidade
univer-sidade
universi-dade
u-niversidade
universida-de
u-ni-ver-si-da-de
```

5 Problema da sexta-feira treze[9]

É incomum uma sexta-feira treze?

Isto é, o décimo terceiro dia do mês cai em uma sexta-feira menos vezes do que nos outros dias da semana? Para responder à esta questão escreva um programa que, dado um intervalo de anos, compute a frequência em que o décimo terceiro dia de cada mês foi/será um domingo, uma segunda, uma terça, uma quarta, uma quinta, uma sexta e um sábado. Para um dado número N de anos o intervalo a ser testado será de primeiro de janeiro de 1900 até 31 de dezembro de 1900+N-1. N será não negativo e irá ser menor que 400.

Existem alguns fatos que você precisa saber para resolver este problema:

- O dia primeiro de janeiro de 1900 foi uma segunda-feira.
- Setembro, abril, junho e novembro têm 30 dias, todos os outros meses tem 31 exceto fevereiro que tem 28, exceto nos anos bissextos nos quais ele tem 29.
- Todo ano par divisível por 4 é um ano bissexto (1992 = 4*468 então 1992 será um ano bissexto, mas 1990 não é um ano bissexto).
- A regra acima não é válida para anos de virada de século. Estes anos devem ser divisíveis por 400 para serem anos bissextos, todos os outro não são. Assim, o ano 1700, 1800, 1900 e 2100 não são bissextos, mas 2000 é bissexto.

Para resolver esta questão não é permitido o uso de funções de data da linguagem.

Não compute previamente as respostas.

Formato de entrada

Uma linha com o inteiro N.

Exemplo de entrada

```
20
```

Formato de saída

Uma linha com sete inteiros separados por espaço. Estes inteiros representam o número de vezes em que o décimo terceiro dia do mês caiu no sábado, no domingo, na segunda, na terça, ..., na sexta **[atenção à ordem]**.

Exemplo de saída

```
36 33 34 33 35 35 34
```

9 Tradução de Friday the Thirteenth da USACO (http://ace.delos.com/usacoprob2?a=rRnbNp27COM&S=friday).

6 Problema do arranjo dos caracteres

Considere um arranjo com repetição de N (2<=N<=9) elementos (caracteres maiúsculos) agrupados de M em M (2<=M<=9) e dispostos em ordem alfabética.

Assim, seja N=4 (A, B, C, D) e M=2 os arranjos dispostos de forma ordenada são:

```
AA AB AC AD BA BB BC BD CA CB CC CD DA DB DC DD
```

Outro exemplo, sendo N=5 (A, B, C, D, E) e M=3 os arranjos dispostos de forma ordenada são:

```
AAA AAB AAC AAD AAE ABA ABB ABC ABD ABE ACA ACB ACC
ACD ACE ADA ADB ADC ADD ADE AEA AEB AEC AED AEE
BAA BAB BAC BAD BAE BBA BBB BBC BBD BBE BCA BCB BCC
BCD BCE BDA BDB BDC BDD BDE BEA BEB BEC BED BEE
CAA CAB CAC CAD CAE CBA CBB CBC CBD CBE CCA CCB CCC
CCD CCE CDA CDB CDC CDD CDE CEA CEB CEC CED CEE
DAA DAB DAC DAD DAE DBA DBB DBC DBD DBE DCA DCB DCC
DCD DCE DDA DDB DDC DDD DDE DEA DEB DEC DED DEE
EAA EAB EAC EAD EAE EBA EBB EBC EBD EBE ECA ECB ECC
ECD ECE EDA EDB EDC EDD EDE EEA EEB EEC EED EEE
```

Faça um programa que recebendo N, M e K (K>=1) seja capaz de informar qual o elemento na posição K desta lista de elementos do arranjo.

Considere que:

- Os arranjos devem ser informados em letras maiúsculas.
- Os elementos do conjunto origem do arranjo vão de A até o caractere de número N do alfabeto. Assim, para N=2 o conjunto origem é {A,B}, para N=6, {A, B, C, D, E, F}.
- O primeiro elemento tem a posição K=1.

Exemplo 1

Entrada

```
4
2
5
```

Saída

```
BA
```

Exemplo 2

Entrada

```
5
3
10
```

Saída

```
ABE
```

Exemplo 3

Entrada

```
5
6
1000
```

Saída

```
ABCEEE
```

Exemplo 4

Entrada

```
2
5
20
```

Saída

```
BAABB
```

7 Problema da cifra no DNA

Um cientista resolveu cifrar uma mensagem em uma molécula de DNA. O método era bastante simples, baseado na ordem alfabéticas das combinações das quatro bases que compõem esta molécula (adenina - A -, citosina - C -, guanina - G - e timina - T) agrupadas três a três.

Desta forma, AAA (3 bases de adenina) representam a letra A, AAC (2 bases de adenina e uma de citosina) a letra B, AAG (2 bases de adenina e uma de timina) a letra C e assim por diante até CGC (1 base citosina, 1 base de guanina e outra de citosina) que representa a letra Z.

Qualquer combinação de 3 elementos (ou bloco de combinações) que não represente letra, representa um espaço.

Faça um programa que, recebendo um conjunto de moléculas de DNA, seja capaz de informar a mensagem nela contida.

Considerações:

- O tamanho máximo da molécula de DNA é de 200 bases.
- A entrada termina com um caractere zero.
- Todas as entradas e as saídas devem ser em letras maiúsculas.
- Não é necessário fazer verificação da entrada.

Exemplo 1

Entrada

```
AAACCAAGTAAA
ATTCACATGACGCACAAAATAAAAAAGAAAATG
AAGATGATAATTCCACATAAAAATATGCAC
AGCAAAATCACAAGTAAA
AAGAAAATCACACATAAA
ACCAAAAAGAAA
ACACAGAAGATGAGTAAA
CCAATCAGACCCACACACCAGAGAAATAAAAATACA
CATACAAAGAGTAAAAATATG
ATCATGCATACAAACATGATGAGG
AACAAACATACACACAGAAAA
ATAACACAGAAA
AAGAAAAATACAAGACACAAA
AAGATGATCCAGATGAGTACA
CGCAAAATCACGAAA
0
```

Saída

```
AULA
PROGRAMACAO
COMPUTADOR
JANELA
CANETA
FACA
ESCOLA
UNIVERSIDADE
TECLADO
NOTEBOOK
```

```
BATERIA
MESA
CADEIRA
CONSOLE
ZANGA
```

Exemplo 2

Entrada

```
AAGATGAGACAGAAACGGCAACCAAAAAGTCAACCAACACAC
AGTAAAAACATGCACAAACATATGCACAGAATGCGGAATACACGTATTCACATGACGCACAAAATAAAAAAGAAAATGCTAAGAAGA
CCAATCAGACCCACACACCAGAGAAATAAAAATACACGGACCACAAATACACACAAAAGTCGTAATAAACTAAACAAAACTAGAAAA
0
```

Saída

```
COISA QUALQUER
LABORATORIO DE PROGRAMACAO II
UNIVERSIDADE FEDERAL DA BAHIA
```

8 Problema do grafo conexo

Informalmente, podemos definir que um grafo G é dito conexo se cada um dos vértices está ligado ao outro por pelo menos um "caminho".

Faça um programa que recebendo dois número inteiro V (2<=V<=1000) e A (1<=A<=2000), seja capaz de ler A pares de vértices um por linha.

Cada par de vértices estará no formato

```
X Y
```

sendo que X e Y estão conectados por uma aresta e são identificados por números positivos no intervalo fechado entre [0,999].

O programa deverá retornar 1 caso o grafo seja conexo ou 0 caso contrário.

Exemplo 1

Entrada

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

Saída

```
1
```

Exemplo 2

Entrada

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

Saída

```
0
```

9 Problema do dicionário de sinônimos

Segundo o dicionário Larousse, sinônimo é aquilo "que ou o que tem a mesma significação, ou quase idêntica".

Sua tarefa é construir um programa para identificar se uma palavra é sinônima de outra. Para executá-la, você receberá um dicionário de sinônimos e deve interpretá-lo considerando a seguinte definição:

- Sendo A e B palavras não necessariamente distintas, se A é sinônimo de B, B é sinônimo de A.
- Sendo A, B e C palavras, se A é sinônimo de B e B é sinônimo de C então A é sinônimo de C.

A entrada do programa será composta por um inteiro N (1<=N<=100) e N definições de sinônimo, uma por linha, sendo que cada linha terá o formato:

```
palavra: palavra_sinonimo
```

Ou seja, uma palavra (de até 50 caracteres), seguida de dois pontos um espaço e a palavra sinônimo (também de até 50 caracteres). Por exemplo:

```
inocente: ingênuo
```

Após entrar com N e o dicionário, o usuário informará no mesmo formato acima uma expressão que deve ser avaliada como verdade, caso com base no dicionário e na regra descrita, as palavras informadas sejam sinônimas ou falso caso contrário.

Se a expressão for avaliada como verdade, o programa deve retornar uma linha com o caractere "V", caso seja falsa ou quando no mínimo uma das palavras informadas pelo usuário não esteja no dicionário de sinônimos a resposta do sistema deverá ser "F".

Diferenças entre maiúsculas e minúsculas devem ser desconsideradas.

Exemplo 1

Entrada

```
2
inocente: ingênuo
ingênuo: simples
inocente: ingênuo
```

Saída

```
V
```

Exemplo 2

Entrada

```
2
inocente: ingênuo
ingênuo: simples
inocente: simples
```

Saída

```
V
```

Exemplo 3

Entrada

```
2
inocente: ingênuo
ingênuo: simples
simples: inocente
```

Saída

```
V
```

Exemplo 4

Entrada

```
3
inocente: ingênuo
ingênuo: simples
contagiar: contaminar
simples: contaminar
```

Saída

```
F
```

Exemplo 5

Entrada

```
2
inocente: ingênuo
ingênuo: simples
simples: puro
```

Saída

```
F
```

Exemplo 6

Entrada

```
6
a: b
b: c
b: d
b: e
b: f
b: g
a: g
```

Saída

```
V
```

Exemplo 7

Entrada

```
6
a: e
b: c
b: d
b: e
b: f
b: g
a: g
```

Saída

```
V
```

Exemplo 8

Entrada

```
7
a: e
b: c
b: d
b: e
b: f
b: g
h: d
a: h
```

Saída

```
V
```

Exemplo 9

Entrada

```
5
a: a
b: b
c: d
d: b
b: a
a: d
```

Saída

```
V
```

Exemplo 10

Entrada

```
2
a: b
b: a
a: b
```

Saída

```
V
```

Exemplo 11

Entrada

```
2
a: b
b: a
a: a
```

Saída

```
V
```

10 Problema da matriz do Paint

Sendo M uma matriz e P um elemento da matriz com o valor V, considera-se que a matriz foi "pintada" corretamente se dado um novo valor K, P e todos os seus elementos vizinhos com o mesmo valor V passarem a ter o novo valor K. Cada novo vizinho pintado deve ser considerado um novo elemento P para o qual a operação descrita deverá novamente ser repetida.

Considere que:

- a posição de P é dada por dois valores x e y. x sendo o índice de coluna e y o de linha, ambos iniciados em zero no local onde normalmente corresponderia ao elemento 1x1 da matriz.
- se P já contiver o valor K, ou seja, V=K, nenhuma operação deverá ser executada.
- são considerados vizinhos os elementos da horizontal, vertical ou diagonais.

Faça um programa que receberá como entrada seis parâmetros na seguinte ordem:

- Quantidade L (1<=L<=1000) de linhas da matriz M.
- Quantidade C (1<=C<=1000) de colunas da matriz M.
- L linhas de C inteiros entre 1 e 1000 correspondente à matriz M.
- Posição x do elemento P.
- Posição y do elemento P.
- K (novo valor do elemento P).

A saída será a matriz M alterada conforme descrição.

Exemplo 1

Entrada

```
7
7
1 1 3 0 0 0 1
0 0 1 2 0 1 0
0 0 1 0 1 0 1
9 6 3 1 0 1 0
6 2 4 7 1 0 1
0 0 0 0 0 0 1
0 1 9 6 0 1 0
0
1
2
```

Saída

```
1 1 3 0 0 0 1
2 2 1 2 0 1 0
2 2 1 0 1 0 1
9 6 3 1 0 1 0
6 2 4 7 1 0 1
0 0 0 0 0 0 1
0 1 9 6 0 1 0
```

Exemplo 2

Entrada

```
7
7
1 1 3 0 0 0 1
0 0 1 2 0 1 0
0 0 1 0 1 0 1
9 6 3 1 0 1 0
6 2 4 7 1 0 1
0 0 0 0 0 0 1
0 1 9 6 0 1 0
0
0
2
```

Saída

```
2 2 3 0 0 0 2
0 0 2 2 0 2 0
0 0 2 0 2 0 2
9 6 3 2 0 2 0
6 2 4 7 2 0 2
0 0 0 0 0 0 2
0 1 9 6 0 2 0
```

11 Problema da memória transacional

Considere um sistema hipotético de memória no qual cada posição de memória está identificada por uma letra maiúscula do alfabeto (assim, existem 26 posições possíveis de armazenamento). Assuma que o trabalho de escrita é executado em um esquema transacional (tudo ou nada).

Você deverá fazer um programa que deverá executar a escrita transacional considerando uma memória inicialmente vazia.

A entrada do programa consistira de uma cadeia sem espaços de até 1000 caracteres. Este cadeia pode ser vista como uma sequência de operações. Cada operação é separada da outra por uma vírgula e tem um dos seguintes formatos:

- L=V, sendo L uma letra maiúscula do alfabeto e V uma valor qualquer (numérico ou alfanumérico). Esta simbologia indica que deve-se armazenar na posição de memória L o valor V.
- B, indicando o início de uma transação.
- R, indicando que a última transação aberta deve ser desfeita.
- C, indicando que a última transação aberta deve ser efetivada.

A saída do programa deverá ser o conteúdo das posições ocupadas da memória no formato L=V, sendo L a posição de memória e V o conteúdo. Posições vazias não devem ser exibidas.

Exemplo 1

Entrada

```
A=28
```

Saída

```
A=28
```

Exemplo 2

Entrada

```
B,A=28,B,A=29,C,A=50,R,B=50
```

Saída

```
B=50
```

Exemplo 3

Entrada

```
B,A=25,B,A=29,R,C,B=50
```

Saída

```
A=25
B=50
```

12 Problema do palíndromo[10]

Números palíndromos são aqueles que são lidos da mesma forma de trás para frente. O número 12321 é um palíndromo típico.

Dado uma base B (2<=B<=20), imprima todos os inteiros N (1<=N<=300) tais que o quadrado de N seja um palíndromo quando representado na base B; imprima também todos os quadrados. Use as letras 'A', 'B'... para representar os dígitos 10, 11...

Lembre-se de imprimir tanto o número quanto seu quadrado na base B.

Formato de entrada

Uma única linha com B, a base.

Exemplo de entrada

```
10
```

Formato de saída

Linhas com dois inteiros na base B. O primeiro sendo o número cujo quadrado é um palíndromo; o segundo o quadrado.

Exemplo de saída

```
1 1
2 4
3 9
11 121
22 484
26 676
101 10201
111 12321
121 14641
202 40804
212 44944
264 69696
```

Variação 1[11]

Considere mais de um caso de teste por vez. O final do teste será identificado com B igual a zero.

Entrada

```
16
17
0
```

Saída

```
1 1
2 4
3 9
```

10 Tradução de Palindromic Squares da USACO (http://ace.delos.com/usacoprob2?a=2jMrvubPVMa&S=palsquare).
11 Esta variação não faz parte do problema original da USACO.

```
11 121
22 484
101 10201
111 12321
121 14641
1 1
2 4
3 9
4 G
6 22
C 88
11 121
1B 2C2
22 484
4G 1771
66 2662
101 10201
```

13 Problema dos fazendeiros trabalhadores[12]

Três fazendeiros acordam à 5 toda manhã e se dirigem ao curral para ordenhar três vacas. O primeiro fazendeiro começa a ordenhar sua vaca no tempo 300 (mensurado em segundos corridos após às 5) e termina no tempo 1000. O segundo fazendeiro começa no tempo 700 e termina no tempo 1200. O terceiro fazendeiro começa no tempo 1500 e termina no tempo 2100. O maior tempo contínuo durante o qual no mínimo um fazendeiro ordenhava uma vaca foi de 900 segundos (do tempo 300 ao tempo 1200). O maior tempo onde nenhuma ordenha foi feita durante a primeira e a última ordenha foi de 300 segundos (1500 menos 1200).

Seu trabalho é escrever um programa que examine uma lista de tempos iniciais e finais para N (1<=N<=5000) fazendeiros e compute (em segundos):

- O intervalo de tempo mais longo onde pelo menos uma vaca era ordenhada.
- O intervalo de tempo mais longo (depois do início da primeira ordenha) durante o qual nenhuma vaca foi ordenhada.

Formato da entrada

- Linha 1: Um único inteiro.
- Linha 2..(N+1): Com dois inteiros não negativos menores que 1000000, representado o tempo em segundos de início e de fim da ordenha de cada fazendeiro.

Exemplo de entrada

```
3
300 1000
700 1200
1500 2100
```

Formato de saída

Uma única linha com dois inteiros que representam [respectivamente] o maior tempo contínuo de ordenha e ocioso.

Exemplo de saída

```
900 300
```

12 Tradução do problema Milking Cows da USACO (http://ace.delos.com/usacoprob2?a=rRnbNp27COM&S=milk2) .

14 Problema do teste oftálmico para programadores

O cientista Oculam resolveu desenvolver um teste oftálmico para programadores de computador. Neste teste um bloco de código está escrito em espiral. Por exemplo, o código **IF(A==1)B=(1+2);** no teste estaria representado da seguinte forma:

```
IF(A
1+2=
(;)=
=B)1
```

O objetivo do programador no teste é identificar se a colocação dos parênteses está correta, ou seja, se

- todo parêntese aberto é fechado;
- nenhum parêntese é fechado sem outro ser aberto antes.

Nenhum outro aspecto da sintaxe da linguagem é importante.

Faça um programa capaz de dar o gabarito de um teste a partir de uma matriz NxN.

Formato da entrada

- Uma linha contendo um inteiro N representado quantidade de linhas e colunas da matriz quadrada M.
- N linhas contendo os elementos de cada linha de M.

Formato de saída

Uma única linha contendo a palavra OK, caso os parênteses estejam postos corretamente ou a palavra ERRO, caso contrário.

Exemplo 1

Entrada

```
4
IF(A
1+2=
(;)=
=B)1
```

Saída

```
OK
```

Exemplo 2

Entrada

```
4
IF)A
1+2=
(;(=
=B)1
```

Saída

```
ERRO
```

Exemplo 3

Entrada

```
1
A
```

Saída

```
OK
```

Exemplo 4

Entrada

```
1
(
```

Saída

```
ERRO
```

Exemplo 5

Entrada

```
3
)))
ABC
(((
```

Saída

```
ERRO
```

Exemplo 6

Entrada

```
3
(((
ABC
)))
```

Saída

```
OK
```

Exemplo 7

Entrada

```
3
(()
ABC
))(
```

Saída

```
OK
```

15 Problema da grade de programação

Um professor adora séries de televisão. Com o guia de programação em mãos, ele selecionou vários programas que desejava assistir sem se importar com os horários de início e de fim.

Faça um programa que receba a lista de séries com seus horários de início e fim e que retorne a lista de séries que não chocam com nenhuma outra.

Formato de entrada

- Uma linha com um inteiro N (1<=N<=100).
- N linhas, cada uma com o nome do programa (com até 10 caracteres), o momento início e o momento fim, ambos intervalos fechados entre 1 e 300.

Formato de saída

- Uma linha contendo a lista dos programas que não se chocam em ordem alfabética e separados por espaço. Caso não haja programas deve-se retornar uma linha em branco.

Exemplo 1

Entrada

```
1
a 1 2
```

Saída

```
a
```

Exemplo 2

Entrada

```
2
a 1 3
b 2 4
```

Saída

```

```

A saída neste caso é uma linha vazia.

Exemplo 3

Entrada

```
3
a 1 3
b 4 5
c 6 10
```

Saída

```
a b c
```

Exemplo 4

Entrada

```
4
a 1 3
b 2 7
c 6 10
d 11 12
```

Saída

```
d
```

Exemplo 5

Exemplo

```
8
a 1 3
b 2 7
c 6 10
d 11 12
e 1 13
f 14 16
g 12 15
h 17 18
```

Saída

```
h
```

16 Problema da permutação

De modo informal e para os objetivos deste problema, podemos dizer que permutação é a forma de rearranjar as letras de uma palavra.

Faça um problema capaz de receber vários casos de teste. Cada caso de teste contém uma cadeia M e uma cadeia N ambas sem espaço com até 10 caracteres.

Seu objetivo é identificar para cada caso de teste a ordem em que a cadeia N apareceria caso ordenássemos todas as permutações de M em ordem alfabética.

A saída deve conter três linhas. A primeira linha é uma cadeia no formato "Teste K", onde K é o número do caso de testes começando de 1. A segunda linha é um número inteiro que identifica a posição da permutação N segundo os critérios descritos. A terceira linha deve ser uma linha em branco.

A entrada termina quando N=M=0.

Algumas considerações:

- A própria string M é considerada uma permutação de si própria.
- Caso N não seja uma permutação de M o valor da ordem deve ser informado como -1.

Exemplo

Entrada

```
abc abc
abc acb
abc bac
abc bca
abc cab
abc cba
0 0
```

Saída

```
Teste 1
1

Teste 2
2

Teste 3
3

Teste 4
4

Teste 5
5

Teste 6
6
```

17 Problema da caminhada perfeita

Considere uma representação de um tabuleiro de Xadrez em uma matriz de String 8x8 e o seguinte padrão:

- T para Torre (movimentam-se N casas na vertical e horizontal).
- C para Cavalo (movimentam-se em L, um L por vez).
- B para Bispo (movimentam-se N casas em diagonal).
- Q para Rainha (movimentam-se N casas na vertical, horizontal e diagonal).
- K para Rei (movimenta-se uma casa por vez na vertical, horizontal e diagonal).
- 0 (zero) casa não ocupada.

Uma peça disposta neste tabuleiro executa uma caminha perfeita se é possível que, em jogadas **sucessivas**, ela tome ("coma") todas as outras peças do tabuleiro sem que em nenhuma jogada deixe de tomar peça.

Faça um programa que, recebendo uma matriz 8x8 (oito linhas de oito caracteres), seja capaz de retornar um V se existir uma peça capaz de executar uma caminhada perfeita ou F caso não exista.

Exemplo 1

Entrada

```
00000000
0C000000
00000000
00T00000
0000Q000
0B000000
000K0000
00000000
```

Saída

```
V
```

O cavalo é capaz de realizar uma caminhada perfeita.

Exemplo 2

Entrada

```
00000000
0T000000
00000000
00000000
0000Q000
000K0000
00B00000
0C000000
```

Saída

```
V
```

O bispo e a rainha são capaz de realizar uma caminhada perfeita.

Exemplo 3

Entrada

```
00000000
00000000
00000000
00000000
0000T000
000K0000
00C00000
00000000
```

Saída

```
F
```