acm International Collegiate Programming Contest 2005 IBM event sponsor

Maratona de Programação

Sociedade Brasileira de Computação

10 de setembro de 2005

ACM International Collegiate Programming Contest 2005

South American Regional - Brazilian Round

Sessão de Aquecimento

(Este caderno contém 2 problemas; as páginas são numeradas de 1 a 2)

Sedes:

Centro Universitário do Triângulo – Uberlândia, MG
Faculdades COC – Riberão Preto, SP
Faculdades Integradas Módulo – Caraguatatuba, SP
Fundação Universidade Federal do Rio Grande – Rio Grande, RS
Pontifícia Universidade Católica de Campinas – Campinas, SP
Universidade Anhembi Morumbi – São Paulo, SP
Universidade da Amazônia – Belém, PA
Universidade de Brasília – Brasília, DF
Universidade de Fortaleza – Fortaleza, CE
Universidade do Vale do Itajaí – Itajaí, SC
Universidade do Vale do Paraíba - São José dos Campos, SP
Universidade Estadual do Oeste do Paraná – Cascavel, PR
Universidade Federal de Minas Gerais – Belo Horizonte, MG
Universidade Federal do Maranhão – São Luís, MA
Universidade Federal do Mato Grosso do Sul – Campo Grande, MS
Universidade Federal do Rio de Janeiro – Rio de Janeiro, RJ
Universidade Federal do Rio Grande do Sul – Porto Alegre, RS
Universidade Federal do Sergipe – Aracaju, SE
Universidade Salgado de Oliveira – Juiz de Fora, MG

Problema A
Brincadeira

Arquivo fonte: `brinca.c, brinca.cpp, brinca.java` *ou* `brinca.pas`

Alice e Beto são amigos desde crianças. Hoje em dia estão estudando na universidade, mas sempre que se encontram relembram os tempos de infância tirando par-ou-ímpar para decidir quem escolhe o filme a ser assistido, ou qual o restaurante em que vão almoçar, etc.

Ontem Alice confidenciou a Beto que ela guarda os resultados de cada vez que tiraram par-ou-ímpar desde que a brincadeira começou, no jardim de infância. Foi uma grande surpresa para Beto! Como Beto cursa Ciência da Computação, ele decidiu mostrar a Alice sua habilidade em programação, escrevendo um programa para determinar quantas vezes cada um ganhou o par-ou-ímpar no período de todos esses anos.

Entrada

A entrada contém vários casos de teste. A primeira linha de um caso de teste contém um único inteiro $1 \leq N \leq 10000$ que indica o número de vezes que os amigos tiraram par-ou-ímpar. A segunda linha de um caso de teste contém N inteiros R_i, separados por espaço, descrevendo a lista de resultados. Se $R_i = 0$ significa que Alice ganhou o i-ésimo jogo, se $R_i = 1$ significa que Beto ganhou o i-ésimo jogo ($1 \leq i \leq N$). O final da entrada é indicado por $N = 0$.

A entrada deve ser lida da entrada padrão.

Saída

Para cada caso de teste da entrada seu programa deve produzir uma linha na saída, no formato '`Alice ganhou X e Beto ganhou Y`', onde `X` ≥ 0 e `Y` ≥ 0.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**

`5`
`0 0 1 0 1`
`6`
`0 0 0 0 0 1`
`0`
	`Alice ganhou 3 e Beto ganhou 2`
	`Alice ganhou 5 e Beto ganhou 1`

Problema B
Socorro!

Arquivo fonte: `socorro.c, socorro.cpp, socorro.java` *ou* `socorro.pas`

Bem, nós temos que admitir: precisamos da ajuda de vocês. Este ano as coisas não aconteceram como previsto, e não conseguimos terminar o software do sistema de competição em tempo. Uma parte vital está faltando, e como vocês sabem, precisamos do sistema funcionando corretamente para esta tarde. A parte do sistema que está faltando é o módulo que computa a pontuação de um time, dada a lista de submissões desse time. Socorro, socorro, alguém nos ajude por favor!

Entrada

A entrada contém vários casos de teste. A primeira linha de um caso de teste contém um inteiro N indicando o número de submissões do time. Cada uma das N linhas seguintes descreve uma submissão, no formato

```
problema minutos resultado
```

onde `problema` é uma letra de 'A' a 'Z', `minutos` é um inteiro representando os minutos passados desde o início da competição até o momento dessa submissão ($0 \leq$ `minutos` ≤ 300) e `resultado` é o resultado dessa submissão ('`correto`' ou '`incorreto`'). As submissões estão em ordem crescente de minutos, e haverá no máximo um resultado `correto` para cada problema. O final da entrada é indicado por $N = 0$.

A entrada deve ser lida da entrada padrão.

Saída

Para cada caso de teste da entrada seu programa deve produzir uma linha na saída, contendo dois inteiros S e P, separados por espaço, onde S representa o número de problemas com o resultado `correto` e P é o tempo de submissão (em minutos) em que cada problema foi julgado `correto`, acrescido de 20 para cada submissão julgada `incorreto` de um problema que mais tarde foi julgado `correto`.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**
3 A 120 incorreto A 130 incorreto A 200 incorreto 5 A 100 correto B 110 incorreto B 111 correto C 200 correto D 300 incorreto 0	0 0 3 431