Competidor(a): _____

Número de inscrição: _____–_____ *(opcional)*

OBI2021

Caderno de Tarefas

Modalidade Programação • Nível 2 • Fase 1

14 a 16 de Junho de 2021

A PROVA TEM DURAÇÃO DE 2 horas

Promoção:

Sociedade Brasileira de Computação

Apoio:

Instruções

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- Este caderno de tarefas é composto por 8 páginas (não contando a folha de rosto), numeradas de 1 a 8. Verifique se o caderno está completo.
- A prova deve ser feita individualmente.
- É proibido consultar a Internet, livros, anotações ou qualquer outro material durante a prova. É permitida a consulta ao *help* do ambiente de programação se este estiver disponível.
- As tarefas têm o mesmo valor na correção.
- A correção é automatizada, portanto siga atentamente as exigências da tarefa quanto ao formato da entrada e saída de seu programa; em particular, seu programa não deve escrever frases como "Digite o dado de entrada:" ou similares.
- Não implemente nenhum recurso gráfico nas suas soluções (janelas, menus, etc.), nem utilize qualquer rotina para limpar a tela ou posicionar o cursor.
- As tarefas **não** estão necessariamente ordenadas, neste caderno, por ordem de dificuldade; procure resolver primeiro as questões mais fáceis.
- Preste muita atenção no nome dos arquivos fonte indicados nas tarefas. Soluções na linguagem C devem ser arquivos com sufixo *.c*; soluções na linguagem C++ devem ser arquivos com sufixo *.cc* ou *.cpp*; soluções na linguagem Pascal devem ser arquivos com sufixo *.pas*; soluções na linguagem Java devem ser arquivos com sufixo *.java* e a classe principal deve ter o mesmo nome do arquivo fonte; soluções na linguagem Python 3 devem ser arquivos com sufixo *.py*; e soluções na linguagem Javascript devem ter arquivos com sufixo *.js*.
- Na linguagem Java, **não** use o comando *package*, e note que o nome de sua classe principal deve usar somente letras minúsculas (o mesmo nome do arquivo indicado nas tarefas).
- Para tarefas diferentes você pode escolher trabalhar com linguagens diferentes, mas apenas uma solução, em uma única linguagem, deve ser submetida para cada tarefa.
- Ao final da prova, para cada solução que você queira submeter para correção, copie o arquivo fonte para o seu diretório de trabalho ou pen-drive, conforme especificado pelo seu professor.
- Não utilize arquivos para entrada ou saída. Todos os dados devem ser lidos da entrada padrão (normalmente é o teclado) e escritos na saída padrão (normalmente é a tela). Utilize as funções padrão para entrada e saída de dados:
  - em Pascal: *readln, read, writeln, write*;
  - em C: *scanf, getchar, printf, putchar*;
  - em C++: as mesmas de C ou os objetos *cout* e *cin*.
  - em Java: qualquer classe ou função padrão, como por exemplo *Scanner, BufferedReader, BufferedWriter* e *System.out.println*
  - em Python: *read,readline,readlines,input,print,write*
  - em Javascript: *scanf, printf*
- Procure resolver a tarefa de maneira eficiente. Na correção, eficiência também será levada em conta. As soluções serão testadas com outras entradas além das apresentadas como exemplo nas tarefas.

Idade de Camila

Nome do arquivo: "`idade.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js` *ou* `py`

Cibele, Camila e Celeste são três irmãs inseparáveis. Estão sempre juntas e adoram fazer esportes, ler, cozinhar, jogar no computador... Agora estão aprendendo a programar computadores para desenvolverem seus próprios jogos.

Mas nada disso interessa para esta tarefa: estamos interessados apenas nas suas idades. Sabemos que Cibele nasceu antes de Camila e Celeste nasceu depois de Camila.

Dados três números inteiros indicando as idades das irmãs, escreva um programa para determinar a idade de Camila.

Entrada

A entrada é composta por três linhas, cada linha contendo um número inteiro N, a idade de uma das irmãs.

Saída

Seu programa deve produzir uma única linha, contendo um único número inteiro, a idade de Camila.

Restrições

- $5 \leq N \leq 100$

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
6 9 7	7

Exemplo de entrada 2	**Exemplo de saída 2**
34 36 38	36

Exemplo de entrada 3	**Exemplo de saída 3**
22 25 22	22

Exemplo de entrada 4	**Exemplo de saída 4**
91 91 91	91

Zero para cancelar

Nome do arquivo: "`zero.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js` *ou* `py`

Seu chefe está ao telefone, nervoso. Ele quer que você compute a soma de uma sequência de números que ele vai falar para você ao telefone, para saber o total das vendas em sua mais recente viagem de negócios.

Infelizmente, de vez em quando seu chefe fala números errados para você ao telefone.

Felizmente, seu chefe rapidamente percebe que falou um número errado e diz "zero", que como combinado previamente quer dizer *ignore o último número corrente*.

Infelizmente, seu chefe pode cometer erros repetidos, e diz "zero" para cada erro.

Por exemplo, seu chefe pode falar ao telefone "Um, três, cinco, quatro, zero, zero, sete, zero, zero, seis", o que significa uma soma total igual a 7, conforme explicado na tabela abaixo:

Fala do chefe	**Números correntes**	**Explicação**
"um, três, cinco, quatro"	1,3,5,4	registre os quatro números
"zero, zero"	1, 3	ignore os dois últimos números
"sete"	1, 3, 7	registre o sete ao final da lista
"zero, zero"	1	ignore os dois últimos números
"seis"	1, 6	registre seis ao final da lista

Para não deixar seu chefe ainda mais nervoso, escreva um programa que determine a soma total dos números falados por seu chefe ao telefone.

Entrada

A primeira linha da entrada contém um inteiro N, a quantidade de números inteiros (incluindo os "zeros") que o seu chefe falou ao telefone. Cada uma das N linhas seguintes contém um número inteiro X_i.

Saída

Seu programa deve produzir uma única linha na saída, contendo um único inteiro, a soma correta dos números, levando em conta que o valor 0 significa erro, conforme descrito.

Restrições

- $1 \leq N \leq 100\ 000$
- $0 \leq X_i \leq 100$, para $(1 \leq i \leq N)$
- $0 \leq$ resultado $\leq 1\ 000\ 000$

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
4 3 0 4 0	0

Exemplo de entrada 2	**Exemplo de saída 2**
10 1 3 5 4 0 0 7 0 0 6	7

Tempo de resposta

Nome do arquivo: "`tempo.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js` *ou* `py`

Sara adora trocar mensagens com amigos. Como ela recebe e envia muitas mensagens, está preocupada com o tempo que seus amigos esperam para receber respostas das mensagens.

As seguintes regras de etiqueta são sempre obedecidas:

- as únicas mensagens que Sara envia são respostas a mensagens que ela recebeu.
- Sara envia no máximo uma mensagem como reposta a uma mensagem que recebeu.
- um amigo de Sara nunca envia uma nova mensagem para Sara até que tenha recebido resposta da mensagem que enviou anteriormente.

O aplicativo de mensagens que Sara e seus amigos usam recebe e envia mensagens instantaneamente. O envio e o recebimento de mensagens são chamados de *eventos*. O aplicativo registra cada evento na ordem em que os eventos ocorrem, usando dois tipos de registro:

- $R\ X$ indica que uma mensagem foi recebida do amigo X.
- $E\ X$ indica que uma mensagem foi enviada ao amigo X.

O aplicativo usa ainda um outro tipo de registro, para indicar o tempo que se passou entre dois eventos consecutivos, na forma

- $T\ X$ indicando que X segundos se passaram entre o evento anterior e o evento posterior a esse registro.

Se não há registro do tipo $T\ X$ entre dois registros de eventos consecutivos significa que exatamente 1 segundo se passou entre esses dois eventos.

O *Tempo de Resposta* de uma mensagem é o tempo que se passa entre o recebimento da mensagem por Sara e o envio da resposta a essa mensagem por Sara. Se um amigo recebeu respostas para todas as suas mensagens, o *Tempo de Resposta Total* para esse amigo é a soma dos Tempos de Respostas para as mensagens desse amigo; caso contrário o Tempo de Resposta Total para esse amigo é -1.

Dada a lista de registros do aplicativo de Sara, sua tarefa é determinar o Tempo de Resposta Total para cada amigo.

Entrada

A primeira linha da entrada contém um inteiro N, o número de registros. Os amigos de Sara são identificados por números inteiros. Cada uma das N linhas seguintes descreve um registro e contém um caractere (R, E ou T) seguido de um número inteiro X. No caso de registros dos tipos R e E o valor de X indica um amigo de Sara; no caso do registro de tipo T, o valor de X indica o número de segundos que se passaram entre o evento anterior e o posterior.

Saída

Para cada amigo de Sara seu programa deve produzir uma linha na saída contendo dois inteiros: o número do amigo e o Tempo de Resposta Total para esse amigo, em ordem crescente dos números dos amigos.

Restrições

- $1 \leq N \leq 20$
- $1 \leq X \leq 100$

Informações sobre a pontuação

- Para um conjunto de casos de testes valendo 20 pontos, $1 \leq N \leq 10$.
- Para um conjunto de casos de testes valendo 80 pontos, nenhuma restrição adicional.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
5 R 2 R 3 T 5 E 2 E 3	2 6 3 6

Exemplo de entrada 2	**Exemplo de saída 2**
14 R 12 T 2 R 23 T 3 R 45 E 45 R 45 E 23 R 23 T 2 E 23 R 34 E 12 E 34	12 13 23 8 34 2 45 -1

Cifra da Nlogônia

Nome do arquivo: "`cifra.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js` *ou* `py`

O rei da Nlogônia ordenou que todos os documentos importantes sejam "cifrados", para que apenas quem tem conhecimento da cifra possa lê-los (*cifrar* um documento significa alterar o original modificando as letras de acordo com algum algoritmo de cifra).

O rei decretou que o seguinte algotimo deve ser usado para cifrar os documentos:

- Cada consoante deve ser substituída por exatamente três letras, na seguinte ordem:
  1. a própria consoante (vamos chamar de *consoante original*);
  2. a vogal mais próxima da consoante original no alfabeto, com a regra adicional de que se a consoante original está à mesma distância de duas vogais, então a vogal mais próxima do início do alfabeto é usada. Por exemplo, se a consoante original é *d*, a vogal que deve ser usada é *e*, pois esta é a vogal mais próxima; se a consoante original é *c*, a vogal que deve ser utilizada é *a*, porque *c* está à mesma distância de *a* e *e*, e *a* é mais próxima do início do alfabeto.
  3. a consoante que segue a consoante original, na ordem do início ao fim do alfabeto. Por exemplo, se a consoante original é *d*, a consoante a ser utilizada é *f*. No caso de a consoante original ser *z*, deve ser utilizada a própria letra *z*.
- As vogais não são modificadas.

Nesta tarefa, o alfabeto é

a b c d e f g h i j k l m n o p q r s t u v x z

e as vogais são

a e i o u

Por exemplo, a cifra da palavra "ter" é "tuveros" (tuv-e-ros) e a cifra da palavra "paz" é "poqazuz" (poq-a-zuz).

O rei da Nlogônia procura por alguém que saiba escrever um programa que produza a cifra de uma palavra dada. Você pode ajudá-lo?

Entrada

A primeira e única linha da entrada contém uma palavra P formada por letras minúsculas sem acentuação.

Saída

Seu programa deve produzir uma única linha, contendo a palavra cifrada.

Restrições

- A palavra P tem no mínimo uma e no máximo 30 letras, todas minúsculas e sem acentuação.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
`ter`	`tuveros`

Exemplo de entrada 2	**Exemplo de saída 2**
`rei`	`rosei`

Exemplo de entrada 3	**Exemplo de saída 3**
`arteiro`	`arostuveiroso`