

Problema A

Vida Artificial

Nome do arquivo fonte: vida. [c | cpp | java | hs | py]

Tudo morre! Plantas, animais, humanos, qualquer ser complexo feito de carbono que nasce, reproduz-se e, inevitavelmente, morre. No entanto, a célula, o menor componente da vida, é essencial para esse ciclo. Mas, será que existe um criador? A ciência ainda não tem uma resposta definitiva. O que sabemos, porém, é que Joãozinho, aluno da FATEC, deseja criar formas de vida unidimensionais.

Essa vida unidimensional é simples e sem grandes complicações: uma cadeia de bits representa suas células. Um bit ativado indica uma célula viva, enquanto um bit desativado indica uma célula morta. A reprodução dessas criaturas ocorre de uma maneira peculiar, regida por regras que se assemelham a uma gramática. Por exemplo, qualquer bit da cadeia que não seja em uma das extremidades pode ser centralizado e ter aplicado, por exemplo, a regra $001 \rightarrow 1$ e isso significa que, se três células adjacentes contêm duas mortas e uma viva, a reprodução resultará em uma célula viva. A reprodução, segundo Joãozinho, sempre ocorre em grupos de três células.

Para representar a reprodução, é necessária uma cadeia inicial de quatro bits. Essa cadeia é fornecida pela entrada e deve ser processada de acordo com a regra aplicada sequencialmente a cada bit, respeitando os limites da cadeia. Após a aplicação das regras, surge uma nova criatura, processo esse que chamamos de geração. Esse processo pode ser repetido diversas vezes, com o número de gerações também definido como parâmetro.

Joãozinho quer medir a “relevância” dessa criatura, atribuindo mais peso aos últimos bits da cadeia final. A primeira célula tem peso 1, a segunda peso 2, a terceira peso 3, e assim sucessivamente. Sua tarefa é calcular a relevância da criaturinha criada a partir das regras e das gerações fornecidas.

Entrada:

A primeira linha consiste em um inteiro G de modo que $10 \leq G \leq 100$. A segunda linha contém o tamanho T da cadeia inicial, com $1 \leq T \leq 50$. A terceira linha consiste em T valores binários S_i , com $1 \leq i \leq T$ separados por espaço (0 ou 1), representando o estado inicial das células.

As próximas oito linhas representam as regras de reprodução. Cada linha possui quatro valores binários $B_{i,j}$, onde $1 \leq i \leq 8$ e $1 \leq j \leq 4$.

Saída:

Um inteiro R representando a relevância da vida obtida.

Exemplo:

Entrada	Saída
20 7 0 0 0 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 1 0 0 1 0 1 1 1 0 1 0 1 0 0 1 1 0 0 0 0	9

Problema B

Polímeros

Nome do arquivo fonte: *polimeros*. [*c* | *cpp* | *java* | *hs* | *py*]

Você se encontra em um laboratório com um microscópio em mãos. O microscópio está se comportando de forma estranha, visualizando um polímero como uma cadeia de caracteres. Um polímero é composto por unidades, e cada unidade é representada por uma letra, sendo sua polaridade indicada pela capitalização. Por exemplo, *z* e *Z* são o mesmo tipo de polímero, mas com polaridades distintas, o que provoca uma reação que reduz o polímero. Em qualquer caso diferente, como *az*, *ZA*, *zz* ou *ZZ*, nenhuma reação ocorre.

Por exemplo, considere o polímero *dabAcCaCBACcaDA*. Nele, temos a reação entre *cC* (sempre em ordem), resultando em *dabAaCBACcaDA*. A seguir, ocorre uma reação entre *Aa*, produzindo *dabCBACcaDA*, e finalmente uma reação entre *cC* (ou *Cc*, mas o efeito é o mesmo), resultando no produto final *dabCBACA**DA*. O polímero resultante tem 10 unidades.

Sua tarefa é criar um programa que calcule o tamanho final do polímero após todas as reações, de acordo com as observações feitas pelo microscópio.

Entrada:

Uma cadeia de caracteres *S* contendo todas as letras do alfabeto (maiúsculas e minúsculas), e nada mais.

Saída:

Seu programa deve produzir *P*, representando o tamanho do polímero resultante.

Exemplo:

Entrada	Saída
dabAcCaCBACcaDA	10

Entrada	Saída
aaAAaBbC	2

Problema C

C-c, C-v

Nome do arquivo fonte: *cccv*. [*c* | *cpp* | *java* | *hs* | *py*]

Uma operação de C-c, C-v (copiar e colar) em um array de inteiros é simples. Você pode copiar subsequências contíguas de um array e colá-las em qualquer posição desejada. Por exemplo, suponha que você tenha o array $[1, 2, 3, 4, 5]$ e queira copiar da segunda até a quarta posição, inserindo-a após a terceira. Seu array ficaria assim: $[1, 2, 3, 2, 3, 4, 4, 5]$. No entanto, o array original é composto apenas por entradas únicas, e seu tamanho é 5. Você se lembra de ter realizado algumas operações de C-c, C-v no seu array inicial (possivelmente nenhuma, o que também é uma possibilidade) e obteve um array final A . Sua tarefa é exibir o tamanho desse array.

Entrada:

A primeira linha contém um inteiro T , representando o número de casos de teste, de modo que $1 \leq T \leq 50$. A segunda linha contém um inteiro N , representando o tamanho do array, de forma que $1 \leq N \leq 10^3$. A terceira linha contém N inteiros A_i , separados por espaços, representando o array, de modo que $1 \leq A_i \leq 10^3$.

Saída:

Seu programa deve produzir T linhas, representando o tamanho do array original para cada caso de teste.

Exemplo:

Entrada	Saída
3	1
5	3
1 1 1 1 1	3
5	
1 2 3 1 2	
3	
1 2 3	

Problema D

Coordenadas Espaciais

Nome do arquivo fonte: *coords.* [*c* | *cpp* | *java* | *hs* | *py*]

Você está completamente perdido no espaço-tempo e possui apenas um radar especial para se localizar. Essa máquina gera uma série de coordenadas X, Y. Seu objetivo é se mover entre duas localidades, mas você percebe que, dependendo da direção escolhida, poderá cair em uma área infinita e se perder para sempre. Para navegar corretamente, você conclui que a área ao redor dos pontos do radar precisa ser finita. A maior área finita representa a localidade mais segura.

Para calcular essa área, você só pode utilizar a distância de Manhattan (contagem de posições em um grid 2D) entre os pontos mais próximos da coordenada fornecida pelo radar. Seu objetivo é calcular a maior área não-infinita com base nas coordenadas do radar. Suponha que o radar indique as seguintes coordenadas:

- 1,1
- 1,6
- 8,3
- 3,4
- 5,5
- 8,9

Nomeando as coordenadas de A até F e colocando-as em um grid 2D, onde a parte norte está à extrema esquerda, teríamos o seguinte posicionamento:

```
.....  
.A.....  
.....  
.....C.  
...D.....  
.....E....  
.B.....  
.....  
.....  
.....F.
```

Note que este grid é apenas uma visão parcial, pois ele se estende infinitamente. Observe que os pontos mais próximos de cada localidade, de A até F, são marcados no grid. Os pontos marcados com '.' estão equidistantes a mais de uma localidade e, portanto, não são contados como parte da área de qualquer localidade.

```

aaaaa.cccc
aAaaa.cccc
aaaddecccc
aadddeccCc
..dDdecccc
bb.deEeccc
bBb.eeee..
bbb.eeefff
bbb.eeffff
bbb.ffffFf

```

Neste exemplo, as áreas associadas às localidades A, B, C e F são infinitas. Por outro lado, a área de D abrange 9 posições, enquanto a área de E abrange 17 posições (incluindo os próprios pontos). Portanto, a localidade mais segura é E, com uma área de 17.

Entrada:

A primeira linha da entrada contém um número inteiro T ($1 \leq T \leq 50$), seguida por T linhas, cada uma contendo dois inteiros X_i, Y_i ($1 \leq X_i, Y_i \leq 400$) correspondendo às coordenadas do radar. As coordenadas não devem ser separadas por vírgulas para facilitar.

Saída:

Seu programa deve produzir um inteiro A contendo a área máxima não-infinita.

Exemplo:

Entrada	Saída
6 1 1 1 6 8 3 3 4 5 5 8 9	17

Problema E

Derivadas Aritméticas

Nome do arquivo fonte: *derivadas*. [*c* | *cpp* | *java* | *hs* | *py*]

Maria é monitora de Cálculo na Fatec-RL e é especialista no assunto. Certo dia, enquanto estudava as regras de derivação, ela se perguntou se poderia aplicar o mesmo conceito aos números inteiros, em vez de funções. Ela chamou essa operação de D e definiu algumas regras, que são as seguintes:

- $D(1) = D(0) = 0$
- $D(p) = 1$, quando p é primo
- $D(n \cdot m) = D(n) \cdot m + n \cdot D(m)$
- $D(p^n) = n \cdot p^{n-1}$, onde p é primo.

Com base nessas definições, podemos calcular a derivada aritmética de um número. Por exemplo, para calcular $D(12)$, sabemos que $12 = 2 \cdot 6$, então:

$$D(12) = D(2 \cdot 6) = D(2) \cdot 6 + 2 \cdot D(6)$$

Sabemos que $6 = 2 \cdot 3$, o que nos dá:

$$D(12) = D(2) \cdot 6 + 2 \cdot D(2 \cdot 3) = D(2) \cdot 6 + 2 \cdot (D(2) \cdot 3 + 2 \cdot D(3))$$

Como $D(2) = 1$ e $D(3) = 1$ (porque ambos são primos), obtemos:

$$D(12) = 6 + 2 \cdot (3 + 2) = 6 + 10 = 16$$

Agora, vamos calcular $D(125)$, usando a quarta regra, já que $125 = 5^3$. Assim:

$$D(125) = D(5^3) = 3 \cdot 5^2 = 3 \cdot 25 = 75$$

Sua tarefa é escrever um programa que calcule a derivada aritmética de um número inteiro N .

Entrada:

Um número inteiro N , tal que $0 \leq N \leq 10^6$.

Saída:

O valor de $D(N)$, a derivada aritmética de N .

Exemplos:

Entrada	Saída
12	16

Entrada	Saída
125	75

Entrada	Saída
19	1

Entrada	Saída
1	0

Problema F

Palíndromos Octais

Nome do arquivo fonte: *octais*. [*c* | *cpp* | *java* | *hs* | *py*]

João adora diferentes bases numéricas e sua favorita é a base octal. Certo dia, ele estava brincando com essa base e se perguntou se seria capaz de verificar quais números inteiros são palíndromos em sua base favorita, a octal. Por exemplo, o número inteiro 3582_{10} é escrito em octal como 6776_8 , que é um palíndromo (se invertermos a ordem, obtemos o mesmo número). Note que $3582 = 6 \cdot 8^3 + 7 \cdot 8^2 + 7 \cdot 8^1 + 6 \cdot 8^0$.

Para converter um número da base decimal (base 10) para a base octal (base 8), você deve dividir o número por 8 repetidamente, armazenando os restos, até que o quociente seja 0. O número em base octal será formado pelos restos lidos de trás para frente.

Entrada:

Um número inteiro K em base 10, tal que $1 \leq K \leq 10^6$.

Saída:

Um caractere "S" ou "N", indicando se o número é um palíndromo na base octal ou não.

Exemplos:

Entrada	Saída
3582	S

Entrada	Saída
668	N


Entrada	Saída
4745	S

Entrada	Saída
10	N

BOCA	Username: Administrator (site=1)	contest not running
------	----------------------------------	---------------------

Runs	Score	Clarifications	Users	Problems	Languages	Answers	Export
Tasks	Site	Contest	Logs	Reports	Backups	Options	Logout

Available scores: [General](#) [Site_1](#)

#	User	Name	vida	polimeros	cccv	coords	derivadas	octais	Total
1	team01/1	ModalGR	1/-	 9/182	 1/81		 1/68	 1/5	4 (496)
2	team10/1	Captcha Raiders		 4/74	 1/7		3/-	 1/12	3 (153)
3	team05/1	Andrew	2/-	 2/45	 1/76	2/-		 1/22	3 (163)
4	team07/1	Barões do Café	2/-	 3/101	 1/5		2/-	 1/19	3 (165)
5	team12/1	Macau		 4/111	 1/10			 1/50	3 (231)
6	team15/1	Sergio Salgados		3/-	 2/94			 1/25	2 (139)
7	team03/1	Conducate	1/-	 1/181	1/-	1/-		 1/54	2 (235)
8	team06/1	Moon People		1/-	2/-		2/-	2/-	0 (0)
9	team11/1	Garotos de Programa						4/-	0 (0)
10	team13/1	Os Renegados							0 (0)