

Problemas Fáceis (?)



Rotinas de input/output típicas

Qualquer que seja o problema, você deve **dominar as rotinas de input de dados e output de resultados**. Os problemas apresentam diversos tipos de input/output diferentes, por exemplo:

- **Número de casos de teste na primeira linha** e as demais linhas são os casos
- **Múltiplos casos de teste são terminados por valores especiais**, podendo ter ou não inputs adicionais que devem ser ignorados
- **Múltiplos casos de teste terminados por EOF**

Rotinas de input/output típicas

```
/* Números de casos de teste na 1ª linha */
int TC;
scanf("%d", &TC);
while (TC--)
{
    int a, b;
    scanf("%d %d", &a, &b);
    printf("%d\n", a + b);
}

/* Casos terminados por valor especial (0) */
int a, b;
while (scanf("%d %d", &a, &b), (a || b))
    printf("%d\n", a + b);

/* Múltiplos casos de teste terminados por EOF */
int x, y;
while (scanf("%d %d", &x, &y) == 2)
    printf("%d\n", x + y);
```

3	3
1 2	12
5 7	9
6 3	-----

1 2	3
5 7	12
6 3	9
0 0	-----
1 1	

1 2	3
5 7	12
6 3	9
-----	-----

Rotinas de input/output típicas

Alguns problemas exigem que o output dos casos de teste seja numerado sequencialmente, com ou sem uma quebra adicional de linha. Por exemplo: se o problema especificar o seguinte:

"Case [NUMBER]: [ANSWER]" seguida de linha em branco

```
int a, b, c = 0;
while (scanf("%d %d", &a, &b) == 2)
    printf("Case %d: %d\n\n", ++c, a + b);
```

ATENÇÃO: também imprime uma linha em branco após o último caso de teste! Tenha certeza de que o problema exige isso.

1 2	Case 1: 3
5 7	
6 3	Case 2: 12

	Case 3: 9

Rotinas de input/output típicas

Alguns problemas exigem que o output dos casos de teste sejam numerados sequencialmente, com ou sem uma quebra adicional de linha. Por exemplo: se o problema especificar o seguinte:

"Case [NUMBER]: [ANSWER]" seguida de linha em branco exceto após a última linha

```
int a, b, c = 0;
while (scanf("%d %d", &a, &b) == 2)
{
    if (c > 0) printf("\n");
    printf("Case %d: %d\n", ++c, a + b);
}
```

```
-----
| 1 2          | Case 1: 3
| 5 7          |
| 6 3          | Case 2: 12
|-----|
|              | Case 3: 9
|              |-----
```

Rotinas de input/output típicas

Alguns problemas têm um **número variável de inputs por linha**. Nesse caso, uma abordagem comum é dizer quantos inputs serão fornecidos no começo da linha, e depois colocar os inputs:

```
int k;  
while (scanf("%d", &k) == 1)  
{  
    int ans = 0, v;  
    while (k--)  
    {  
        scanf("%d", &v);  
        ans += v;  
        printf("%d\n", ans);  
    }  
}
```

	1	1						1	
	2	3	4					7	
	3	8	1	1				10	
	4	7	2	9	3			21	
	5	1	1	1	1	1		5	
	-----							-----	

Rotinas de input/output típicas

Se os problemas têm um **número variável de inputs por linha**,
mas a quantidade de inputs por linha não é informada, temos que
verificar se o último caractere é `\n`:

```
while (1)
{
    int ans = 0, v;
    char l;
    while (scanf("%d%c", &v, &l) == 2)
    {
        ans += v;
        if (l == '\n') break;
    }
    if (feof(stdin)) break;
    printf("%d\n", ans);
}
```

1	1
3 4	7
8 1 1	10
7 2 9 3	21
1 1 1 1 1	5
-----	-----

Idiomas típicos

Com o tempo você aprenderá diversos idiomas típicos para a programação competitiva. Alguns básicos são, por exemplo:

```
#include <bits/stdc++.h>           // inclui tudo

using namespace std;              // simplifica comandos

typedef long long int ll;         // tipos comuns
typedef pair<int, int> ii;
typedef vector<int> vi;
typedef vector<ii> vii;

memset(memo, -1, sizeof(memo));   // inicializa tabela para
vi memo(n, -1);                  // prog. dinâmica

memset(arr, 0, sizeof(arr));      // inicializa array de int

const int INF = 1e9;              // INF int
const int LLINF = 4e18;           // INF long long int
const double EPS = 1e-9;          // epsilon

index = (index + 1) % n;          // para direita ou volta p/ 0
index = (index + n - 1) % n;      // para esquerda ou volta p/ n-1

int ans = (int) ((double) d + 0.5); // arredondamento

int ans = min(ans, new_computation); // atalho min/max
```


Hora de resolver problemas!

Solucione os **70** (sim, SETENTA) **primeiros** problemas para essa semana (sim, tem mais ainda), divididos em 10 categorias:

Não são as categorias "oficiais" que discutimos anteriormente, são apenas uma divisão didática.

- a) I/O + Seqüências
- b) Repetição
- c) Seleção
- d) Múltiplos casos de teste + Seleção
- e) Controle de fluxo
- f) Funções
- g) Manipulação de Arrays 1D
- h) Diversos, fáceis
- i) Diversos, ainda fáceis
- j) Diversos, médios

- USE A FOLHA DE CONTROLE!
- FAÇA NA ORDEM INDICADA!
- ANOTE NA PLANILHA ONLINE!

1a. Exercícios de Programação para "Esquentar"				
a) I/O + Seqüências				
Ordem	Local	Problema	Acertou (S/N)	Data
1	Kattis	hello		
2	UVa	10071 – Back to High School		
3	UVa	11614 – Etruscan Warriors		
4	UVa	13025 – Back to the Past		
5	Kattis	carrots		
6	Kattis	r2		
7	Kattis	thelastproblem		
b) Repetição				
Ordem	Local	Problema	Acertou (S/N)	Data
1	Kattis	timeloop		
2	UVa	01124 – Celebrity Jeopardy		
3	UVa	11044 – Searching for Nessy		
4	UVa	11547 – Automatic Answer		
5	Kattis	different		
6	Kattis	qaly		
7	Kattis	tarifa		
c) Seleção				
Ordem	Local	Problema	Acertou (S/N)	Data
1	Kattis	moscowdream		
2	Kattis	isithalloween		
3	Kattis	judgingmoose		
4	Kattis	onechicken		
5	Kattis	provincesandgold		
6	Kattis	quadrant		
7	Kattis	temperature		
d) Múltiplos Casos de Teste + Seleção				
Ordem	Local	Problema	Acertou (S/N)	Data
1	Kattis	oddities		
2	UVa	11172 – Relational Operators		
3	UVa	12250 – Language Detection		
4	UVa	12372 – Packing for Holiday		
5	Kattis	elebigility		
6	Kattis	helpaphd		
7	Kattis	leftbehind		

Habilidades BÁSICAS de processamento de strings

Muitos problemas da maratona (e da computação em geral) exigem que você consiga obter, processar e exibir strings (input, formatação e output).

Solucione todas as **6 tarefas** que são correspondentes às **6 habilidades básicas** de processamento de strings.

NÃO CONTINUE ANTES DE REALIZAR TODAS ESSAS TAREFAS NA ORDEM EXATA EM QUE ESTÃO. DOMINE ESSE CONHECIMENTO!

Tente fazer por conta própria, se esforce e, se não conseguir, peça ajuda.

Habilidades BÁSICAS de processamento de strings

HABILIDADE 1:

Dado um arquivo de texto que contenha apenas caracteres alfabéticos [A-Za-z], dígitos [0-9], espaços e pontos (“.”), escreva um programa para ler esse arquivo linha por linha até que seja encontrado uma linha que começa com sete pontos (“.....”). Concatene (combine) cada linha em uma grande string T. Quando duas linhas são combinadas, coloque um espaço entre elas para que a última palavra da linha anterior esteja separada da primeira palavra da próxima linha (atual). Existirão até 30 caracteres por linha e não mais do que 10 linhas nesse arquivo de texto. Não existirão espaços sobrando no final das linhas e cada linha será terminada por um caractere newline (“\n”). Um exemplo de um arquivo que seu programa deve ser capaz de processar é o seguinte:

```
I love CS101 Competitive
Programming. i also love
AlGoRiTm
.....you must stop after reading this line as it starts with 7 dots
after the first input block, there will be one loooooooooooooog line...
```

Agora responda as seguintes perguntas e tenha certeza de que você domina tudo o que está sendo perguntado (se você não dominar, estude!!!):

- Você sabe como armazenar uma string corretamente na linguagem que está usando?
- Como ler um texto de input linha por linha?
- Como concatenar (combinar) duas string em uma string maior?
- Como verificar linhas que começam “.....” para interromper a leitura do input?

HABILIDADE 2:

Suponha que você tenha uma grande string T. Nós queremos checar se uma outra string P pode ser encontrada em T. Imprima todos os índices onde P aparece em T ou imprima “-1” se P não estiver contida em T. Por exemplo,

```
se T = “I love CS101 Competitive Programming. i also love AlGoRiTm”
se P = “I”
```

então o output será apenas {0} (índices começam em zero) pois “I” e “i” são caracteres diferentes em ASCII.

Outros exemplos:

```
se P = “love”, então o output será {2, 45};
```

```
se P = “book”, então o output será {-1}.
```

Agora responda as seguintes perguntas e tenha certeza de que você domina tudo o que está sendo perguntado (se você não dominar, estude!!!):

- Como encontrar a primeira ocorrência de uma substring dentro de uma string (se existir)? Será preciso implementar um algoritmo de emparelhamento de strings (como o Knuth-Morris-Pat) ou podemos utilizar as bibliotecas da linguagem?
- Como encontrar a próxima ocorrência de uma substring em uma string (se existir)?

HABILIDADE 3:

Suponha que você precise fazer alguma análise simples dos caracteres em uma string T e também transformar todos os caracteres de T em minúsculas. A análise necessária é: a) Quantos dígitos [0-9], quantas vogais [aeiouAEIOU] e consoantes [todas as não-vogais, maiúsculas ou minúsculas] existem em T? Como você consegue fazer isso em O(n) onde n = comprimento da string?

Hora de resolver problemas da MARATONA!

Vamos começar com a categoria de problemas **Ad Hoc, que são aqueles que não podem ser classificados em outro lugar, já que cada problema e cada solução são "únicos".**

ID	Categoria	Frequência
1	Ad Hoc	1-2
2	Estrutura de Dados (principalmente)	0-1
3	Busca Completa (iterativa/recursiva)	1-2
4	Dividir e Conquistar	0-1
5	Gulosos (os não-clássicos)	1
6	Programação Dinâmica (os não-clássicos)	1-2
7	Grafos (exceto fluxo em rede/emparelhamento em grafo)	1
8	Matemática	1-2
9	Processamento de string	1
10	Geometria computacional	1
11	Problemas difíceis, raros, emergentes	2-3

Tipo	Categoria	Confiança na Resolução
A1	Eu já resolvi esse tipo de problema antes	E tenho certeza de que posso resolver de novo (rapidamente).
A2	Eu já resolvi esse tipo de problema antes	E tenho certeza de que posso resolver de novo (lentamente).
B	Eu já vi esse tipo de problema antes	Mas daquela vez eu sabia que ainda não conseguia resolver.
C	Eu nunca vi esse tipo de problema antes	Danou-se... ver abaixo algumas dicas.

[illegible]

→ TREINE CLASSIFICAR OS PROBLEMAS!

Maratona: problemas Ad Hoc

Tente resolver **sozinho** e/ou junto com o **time**. A lista contém **112** problemas Ad Hoc no total, sendo que cada aluno deve solucionar, obrigatoriamente, pelo menos **4** de cada categoria (64 problemas). Quanto mais, melhor!

a) Jogos (cartas)

b) Jogos (xadrez)

c/d) Jogos (outros)

e/f/g) Problemas da vida real

h/i) Problemas envolvendo tempo

j) Numerais romanos

k/l) Criptografia

m) Parsing de input

n) Formatação de output

o/p) "Desperdiçadores de Tempo"

→ ANOTE NA PLANILHA ONLINE!

1b. Exercícios AdHoc

a) Jogos (cartas)

Ordem	Local	Problema	Acertou (S/N)	Data
1	UVa	10646 – What is the Card?		
2	UVa	10388 – Snap		
3	UVa	11678 – Card's Exchange		
4	UVa	12247 – Jollo		
5	Kattis	bela		
6	Kattis	shuffling		
7	Kattis	memorymatch		

b) Jogos (xadrez)

Ordem	Local	Problema	Acertou (S/N)	Data
1	UVa	00278 – Chess		
2	UVa	00255 – Correct Move		
3	UVa	00696 – How Many Knights		
4	UVa	10284 – Chessboard in FEN		
5	Kattis	chess		
6	Kattis	empleh		
7	Kattis	helpme		

c) Jogos (outros)

Ordem	Local	Problema	Acertou (S/N)	Data
1	UVa	10189 – Minesweeper		
2	UVa	00489 – Hangman Judge		
3	UVa	000947 – Master Mind Helper		
4	UVa	11459 – Snakes and Ladders		
5	Kattis	connecteddots		
6	Kattis	gamerank		
7	Kattis	guessinggame		

d) Jogos (outros), mais difíceis (entediantes)

Ordem	Local	Problema	Acertou (S/N)	Data
1	Kattis	rockpaperscissors		
2	UVa	00584 – Bowling		
3	UVa	10813 – Traditional BINGO		
4	UVa	11013 – Get Straight		
5	Kattis	battleship		
6	Kattis	tictactoe2		
7	Kattis	turtlemaster		

Já acabou e quer praticar mais?

Só faça se realmente tiver tempo!

1a. Exercícios EXTRAS de Programação para "Esquentar"

a) I/O + Sequências

Ordem	Local	Problema	Acertou (S/N)	Data
1	UVa	11805 – Bafana Bafana		
2	UVa	12478 – Hardest Problem Ever (Easy)		
3	Kattis	faktor		
4	Kattis	planina		
5	Kattis	romans		
6				
7				

b) Repetição

Ordem	Local	Problema	Acertou (S/N)	Data
1	UVa	10055 – Hashmat the Brave Warrior		
2				
3				
4				
5				
6				
7				

c) Seleção

Ordem	Local	Problema	Acertou (S/N)	Data
1				
2				
3				
4				
5				
6				
7				

d) Múltiplos Casos de Teste + Seleção

Ordem	Local	Problema	Acertou (S/N)	Data
1	UVa	00621 – The PATH		
2	UVa	11723 – Numbering Roads		
3	UVa	11727 – Cost Cutting		
4	UVa	12289 – One-Two-Three		
5	UVa	12468 – Zapping		
6	UVa	12577 – Hajj-e-Akbar		
7	UVa	12646 – Zero or One		
8	UVa	12917 – Prop hunt!		
9	Kattis	nastyhacks		
10	Kattis	numberfun		

1b. Exercícios AdHoc EXTRAS

a) Jogos (cartas)

Ordem	Local	Problema	Acertou (S/N)	Data
1	UVa	162 - Beggar My Neighbour		
2	UVa	462 - Bridge Hand Evaluator		
3	UVa	555 - Bridge Hands		
4	UVa	10205 - Stack 'em Up		
5	UVa	10315 - Poker Hands		
6	UVa	11225 - Tarot scores		
7	UVa	12366 - King's Poker		
8	UVa	12952 - Tri-du		
9	Kattis	karte		

b) Jogos (xadrez)

Ordem	Local	Problema	Acertou (S/N)	Data
1	UVa	10196 - Check The Check		
2	UVa	10849 - Move the bishop		
3	UVa	11494 – Queen		
4	Kattis	bijele		
5				
6				
7				

c) Jogos (outros)

Ordem	Local	Problema	Acertou (S/N)	Data
1	UVa	340 - Master-Mind Hints		
2	UVa	10279 - Mine Sweeper		
3	UVa	10409 - Die Game		
4	UVa	12239 – Bingo!		
5	Kattis	trik		
6				
7				

d) Jogos (outros), mais difíceis (entediantes)

Ordem	Local	Problema	Acertou (S/N)	Data
1	UVa	114 - Simulation Wizardry		
2	UVa	141 - The Spot Game		
3	UVa	220 – Othello		
4	UVa	227 – Puzzle		
5	UVa	232 - Crossword Answers		
6	UVa	339 - SameGame Simulation		
7	UVa	379 – Hi-Q		
8	UVa	647 - Chutes and Ladders		
9	Kattis	rockscissorspaper		

Dúvidas?



Foto de Towfiqu Barbhuiya, no Unsplash (<https://unsplash.com/photos/a-blue-question-mark-on-a-pink-background-oZuBNC-6E2s>)