1.5 Habilidades Básicas de Processamento de Strings

Agora introduzimos várias habilidades básicas de processamento de strings que todo programador competitivo deve ter, já que nem todos os formatos de entrada e/ou saída em problemas de competições envolvem apenas números inteiros e/ou strings simples.

Nesta seção, fornecemos uma série de mini-tarefas que você deve resolver uma após a outra, sem pular nenhuma. Você pode usar qualquer linguagem de programação: C/C++, Python, Java e/ou OCaml. Esforce-se para criar a implementação mais curta e eficiente que puder. Depois, compare suas implementações com as nossas (veja as respostas no final deste capítulo ou o código-fonte em nosso repositório GitHub). Se você não se surpreender com nenhuma de nossas implementações (ou até mesmo criar soluções mais simples), estará bem preparado para enfrentar problemas diversos de processamento de strings. Caso contrário, reserve um tempo para estudar nossas implementações.

Tarefa 1:

Dado um arquivo de texto contendo apenas caracteres alfabéticos [A-Za-z], dígitos [0-9], espaços e pontos (‘.’), escreva um programa que leia esse arquivo linha por linha até encontrar uma linha que comece com sete pontos consecutivos (“.......”). Concatene (combine) cada linha em uma única string longa, T. Ao combinar duas linhas, insira um espaço entre elas para que a última palavra da linha anterior fique separada da primeira palavra da linha atual.

- O arquivo tem até 30 caracteres por linha e no máximo 10 linhas no bloco de entrada.
- Não há espaços em branco no final de cada linha, e cada linha termina com um caractere de nova linha.
- Exemplo de entrada:

plaintext
Copiar código

```
I love CS3233 Competitive
Programming. i also love
AlGoRiThM
.......you must stop after reading this line as it starts with 7
dots
after the first input block, there will be one looooooooooooong
line...
```

Subtarefas:

(a) Você sabe como armazenar uma string na sua linguagem de programação favorita?
(b) Como ler um arquivo de texto linha por linha?
(c) Como concatenar (combinar) duas strings em uma maior?
(d) Como verificar se uma linha começa com a string “.......” para parar a leitura?

Tarefa 2:

Dada uma string longa TTT, verifique se outra string PPP pode ser encontrada em TTT. Relate todos os índices onde PPP aparece em TTT ou -1 se PPP não for encontrada.

- Exemplo:
  - T=T =T= `"I love CS3233 Competitive Programming. i also love AlGoRiThM"`
  - P=P =P= `"I"` → Saída: {0}
  - P=P =P= `"love"` → Saída: {2, 46}
  - P=P =P= `"book"` → Saída: {-1}

Subtarefas:

(a) Como encontrar a primeira ocorrência de uma substring em uma string?
(b) Como encontrar as próximas ocorrências de uma substring em uma string?

Tarefa 3:

Faça uma análise simples dos caracteres em TTT e transforme todos os caracteres em minúsculas. Perguntas:

- Quantos dígitos, vogais ([aeiouAEIOU]) e consoantes (outros caracteres alfabéticos) existem em TTT?
- É possível fazer isso em O(n)O(n)O(n), onde nnn é o comprimento de TTT?

Tarefa 4:

Divida a string TTT em tokens (substrings) e armazene-os em um array de strings chamado `tokens`.

- Os delimitadores são espaços e pontos.
- Exemplo: Após tokenizar TTT (em minúsculas):
  - Tokens = {“i”, “love”, “cs3233”, “competitive”, “programming”, “i”, “also”, “love”, “algorithm”}.
  - Ordenando lexicograficamente:
    - {“algorithm”, “also”, “competitive”, “cs3233”, “i”, “i”, “love”, “love”, “programming”}.
    - Menor string lexicograficamente: “algorithm”.

Subtarefas:

(a) Como tokenizar uma string?
(b) Como armazenar tokens em um array de strings?
(c) Como ordenar lexicograficamente um array de strings?

Tarefa 5:

Identifique qual palavra aparece mais vezes em TTT.

- Exemplo: Para TTT, a saída é “i” ou “love”, pois ambas aparecem duas vezes.
- Qual estrutura de dados você usaria para esta tarefa?

Tarefa 6:

O arquivo de texto possui uma última linha após a linha que começa com “.....”, mas o comprimento desta última linha não é limitado.

- Conte quantos caracteres existem na última linha.
- Como ler uma string com comprimento desconhecido?