Universidade de São Paulo – ICMC
Bacharelado em Ciência da Computação
SCC0221 – Introdução à Ciência da Computação I
Prof. Rodrigo Fernandes de Mello – mello@icmc.usp.br
Monitores: Filipe Mariano– filipe.mariano.silva@usp.br,
Igor Martinelli – igor.martinelli@usp.br,
Victor Forbes – victor.forbes@usp.br,
Yule Vaz – yule.vaz@usp.br

Trabalho 5 - Deque

1 Proposta

Nesse trabalho você deve implementar uma Estrutura de Dados dinâmica para Deques na forma de uma Lista Duplamente Encadeada e resolver dois problemas descritos na Seção 4.

2 Descrição

2.1 Deque

Deque (ou *Double-Ended Queue*) é um Tipo Abstrato de Dado (TAD) que representa uma Fila generalizada no qual é permitido apenas inserir ou remover elementos da frente ou de trás, como ilustra a Figura 1.

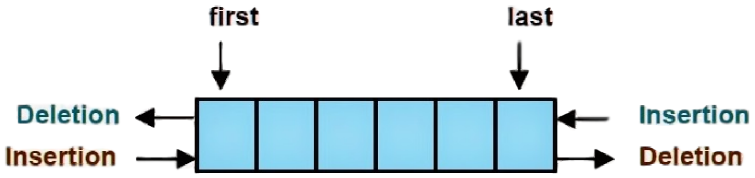

Figura 1: Exemplo de Deque.

2.2 Lista Duplamente Encadeada

Uma Lista Duplamente Encadeada consiste em uma coleção de "nós" ligados entre si. Cada nó, além de armazenar um elemento (um `char` no caso desse trabalho) deve armazenar também uma referência (um ponteiro) para o próximo nó e para o nó anterior, como ilustra a Figura 2. Para facilitar que a lista seja percorrida, o nó do extremo da esquerda usualmente possui um nó "sentinela" como vizinho "anterior" ou apenas `NULL`. O mesmo ocorre para o nó do extremo da direita.

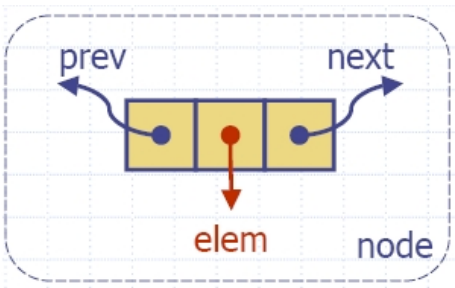

Figura 2: Um nó de uma Lista Duplamente Encadeada.

3 Implementação

Foi disponibilizado um código base no Run Codes com o cabeçalho de todas as funções que devem ser implementadas. Você deve implementar o seguinte conjunto mínimo de operações (funções):

- `create`: Aloca um Deque vazio na Heap.
- `clear`: Remove todos os elementos do Deque.
- `destroy`: Remove todos os elementos e desaloca o Deque.
- `push_front`: Insere um elemento na frente (na cabeça) do Deque.
- `push_back`: Insere um elemento atrás (na cauda) do Deque.
- `front`: Retorna o elemento armazenado na frente do Deque.
- `back`: Retorna o elemento armazenado atrás do Deque.
- `size`: Retorna a quantidade de elementos do Deque.
- `empty`: Retorna 1 caso o Deque esteja vazio e 0 caso contrário.
- `pop_front`: Remove o elemento da frente do Deque.
- `pop_back`: Remove o elemento de trás do Deque.
- `print`: Imprime todos os elementos armazenados no Deque.

Atenção: A impressão do conteúdo do Deque deve ser feita de maneira recursiva. Para que isso possa ser feito você deve criar uma nova função que é chamada pela "print()", pois esta última recebe apenas um ponteiro para o Deque, o que é insuficiente para realizar a recursão.

4 Problemas

4.1 The Last Word

Google Code Jam - Round 1A 2016

On the game show The Last Word, the host begins a round by showing the contestant a string S of uppercase English letters. The contestant has a whiteboard which is initially blank. The host will then present the contestant with the letters of S, one by one, in the order in which they appear in S. When the host presents the first letter, the contestant writes it on the whiteboard; this counts as the first word in the game (even though it is only one letter long). After that, each time the host presents a letter, the contestant must write it at the beginning or the end of the word on the whiteboard before the host moves on to the next letter (or to the end of the game, if there are no more letters).

For example, for $S = CAB$, after writing the word C on the whiteboard, the contestant could make one of the following four sets of choices:

- put the A before C to form AC, then put the B before AC to form BAC
- put the A before C to form AC, then put the B after AC to form ACB
- put the A after C to form CA, then put the B before CA to form BCA
- put the A after C to form CA, then put the B after CA to form CAB

The word is called the last word when the contestant finishes writing all of the letters from S, under the given rules. The contestant wins the game if their last word is the last of an alphabetically sorted list of all of the possible last words that could have been produced. For the example above, the winning last word is CAB (which happens to be the same as the original word). For a game with $S = JAM$, the winning last word is MJA.

You are the next contestant on this show, and the host has just showed you the string S. What's the winning last word that you should produce?

4.1.1 Input

The string S ($1 \leq$ length of $S \leq 10^5$).

4.1.2 Output

The winning last word.

4.2 Strings

Seletiva UFMG para a III Maratona Mineira de Programação

Um jogo envolvendo strings foi utilizado em uma competição de programação recente. Nesse jogo, o jogador recebe caractere por caractere de uma string A, da esquerda para a direita, e pode formar uma string resultante B colocando o caractere recebido no começo ou no final da string que ele tem atualmente. No começo do jogo, o jogador tem uma string vazia.

O objetivo do jogo é obter a maior string possível B, considerando a ordem lexicográfica. Então, os competidores tinham que escrever programas que, dada a string inicial, determinasse qual era a maior string (em ordem lexicográfica) que poderia ser obtida. Quando a competição acabou, os organizadores notaram que várias submissões não computavam a resposta correta. Eles ficaram ainda mais tristes quando perceberam que várias das strings calculadas como resposta por várias submissões nem poderiam ser obtidas seguindo as regras do jogo. Para escrever a análise da competição, um dos organizadores teve a brilhante ideia de computar quantas das strings dadas como resposta nem poderiam ser obtidas seguindo a regra do jogo.

Os organizadores da competição estão muito ocupados preparando novas competições. Por isso, eles pediram que você escrevesse um programa que, dadas a string inicial A do jogo e a string computada B em uma submissão, determine se é possível obter tal string seguindo as regras do jogo.

4.2.1 Entrada

A entrada contém duas linhas com uma string em cada. A string da primeira linha é a string inicial A do jogo, e a da segunda é a string computada B em alguma solução. Todas as strings são formadas apenas por letras maiúsculas e não possuem mais do que 100 caracteres.

4.2.2 Saída

Caso seja possível obter a segunda string, partindo da primeira e seguindo as regras do jogo, imprima uma linha contendo o caractere S. Caso contrário, imprima uma linha contendo o caractere N.

5 Entrada

A primeira linha da entrada contém 1 ou 2, indicando que o problema que seu programa deve resolver é o problema descrito na subseção 4.1 ou na subseção 4.2, respectivamente. Na linha seguinte haverá a entrada do respectivo problema.

6 Saída

Seu programa deve produzir a saída de acordo com os problemas da seção 4.

7 Instruções Complementares

- É obrigatório o uso do código base disponibilizado no Run Codes.
- Não é permitido alterar o cabeçalho das funções do código base no desenvolvimento do trabalho.
- Você deve usar o Deque para resolver ambos os problemas descritos na seção 4.
- Você deve utilizar a Heap para armazenar as `structs`. Apenas o ponteiro para a `struct` Deque deve ser alocado na Stack.
- Todas as `strings` e todos os vetores de tamanho variável devem ser armazenados na Heap.
- Comente seu código.
- Para resolver os casos mais difíceis do segundo problema você provavelmente precisará implementar um algoritmo de Backtracking.

8 Exemplos

Entrada	Saída
1 JAM	MJA

Tabela 1: Primeiro exemplo do problema The Last Word.

Entrada	Saída
1 ABCABCABC	CCCBAABAB

Tabela 2: Segundo exemplo do problema The Last Word.

Entrada	Saída
2 ABC BAC	S

Tabela 3: Primeiro exemplo do problema Strings.

Entrada	Saída
2 ABC BCA	N

Tabela 4: Segundo exemplo do problema Strings.