Universidade Federal de São Carlos – Departamento de Computação
Algoritmos e Estruturas de Dados I
Prof. Alexandre L. M. Levada

Atividade Avaliativa 3

Estruturas de Dados Elementares: Pilhas, Filas, Deques e Filas de Prioridades

A entrega de apenas um documento com os textos dos códigos, sem um link para execução online, acarretará em desconto na nota.

📘 1. Exercício: O Desafio da Compra Inteligente (2.5 pontos) (prático)

Você está em um supermercado com um orçamento limitado e precisa comprar o maior número possível de itens para a sua festa. O supermercado tem duas prateleiras principais (Prateleira A e Prateleira B), e os itens estão dispostos em fila, de modo que você só pode pegar o item que está na frente de cada prateleira.

Cada item tem um preço (um número inteiro não negativo). Você mantém um controle do custo total acumulado dos itens que você pega das duas prateleiras.

Você será desqualificado da promoção do supermercado se, a qualquer momento, seu custo total acumulado se tornar maior que o ORÇAMENTO pré-definido.

Sua pontuação final é o número total de itens que você conseguiu pegar das duas prateleiras.

Dadas as listas de preços dos itens na Prateleira A e na Prateleira B, e o ORÇAMENTO, encontre a pontuação máxima possível que você pode alcançar.

Considere a seguinte estratégia (gulosa): em cada momento, você deve sempre pegar o item com o menor preço disponível no topo da Prateleira A ou da Prateleira B. Implemente duas pilhas para representar as duas prateleiras do supermercado.

Prateleira A = [5, 2, 9, 4, 6, 3, 1, 2, 4] (topo)
Prateleira B = [2, 6, 9, 7, 2, 5, 1, 4, 2, 5, 3] (topo)
ORÇAMENTO = 20

OBS: É preciso implementar um TAD/classe Pilha a partir do zero.

2. 📘 Exercício: O Desafio da Distribuição de Ferramentas (2.5 pontos) (prático)

Imagine que você é o mestre de um canteiro de obras e precisa distribuir ferramentas para seus trabalhadores. Há uma pilha de ferramentas disponíveis e uma fila de trabalhadores, cada um com uma preferência específica por um tipo de ferramenta.

As ferramentas são de dois tipos: chave de fenda (representada por 0) e martelo (representada por 1). Cada trabalhador na fila tem uma preferência por apenas um desses tipos de ferramenta.

O número total de ferramentas na pilha é igual ao número de trabalhadores. As ferramentas estão dispostas em uma pilha, e você só pode pegar a ferramenta que está no topo.

Em cada etapa da distribuição:

Se o trabalhador que está na frente da fila preferir a ferramenta que está no topo da pilha, ele a pegará e sairá da fila.
Caso contrário (se ele não preferir a ferramenta do topo), ele sairá sem pegar a ferramenta e irá para o final da fila, dando a chance para o próximo trabalhador.

Esse processo continua até que a pilha de ferramentas esteja vazia e todos os trabalhadores tenham pegado suas ferramentas, ou até que nenhum dos trabalhadores na fila queira pegar a ferramenta do topo, impedindo que a distribuição continue.

Você recebe dois arrays binários: trabalhadores e ferramentas.

ferramentas[i] é o tipo da i-ésima ferramenta na pilha (onde i = 0 é o topo da pilha).

trabalhadores[j] é a preferência do j-ésimo trabalhador na fila inicial (onde j = 0 é o primeiro da fila).

Seu objetivo é implementar uma pilha para as ferramentas, uma fila para os trabalhadores e retornar o número de trabalhadores que não conseguem pegar uma ferramenta, o que é equivalente ao número de ferramentas que sobram na pilha. Considere a seguinte configuração inicial na pilha e fila

ferramentas = [1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1] (topo)
trabalhadores = (início) [1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1] (fim)

OBS: É preciso implementar um TAD/classe Pilha e um TAD/classe Fila a partir do zero.

3. 📘 Exercício: O Problema do Intervalo de Ações (2.5 pontos) (prático)

O Problema do Intervalo de Ações (Stock Span Problem) envolve determinar, para cada dia em uma série de preços de ações, o número de dias anteriores consecutivos (incluindo o dia atual) em que o preço das ações foi menor ou igual ao preço das ações daquele dia.

Definição formal

Dada uma lista de preços de ações P[1...n], o intervalo de ações S[i] para um dia i é definido como o número máximo de dias consecutivos (incluindo o dia i) em que o preço da ação foi menor ou igual a P[i].

Exemplo:

Suponha que os preços das ações ao longo de 7 dias sejam:

P = [100, 80, 60, 70, 60, 75, 85]

Os intervalos para cada dia são calculados da seguinte forma:

S = [1, 1, 1, 2, 1, 4, 6] pois

Dia i = 1: P[i] = 100 -> S[i] = 1 (primeiro dia)
Dia i = 2: P[i] = 80 -> S[i] = 1 (preço válido somente para hoje)
Dia i = 3: P[i] = 60 -> S[i] = 1 (preço válido somente para hoje)
Dia i = 4: P[i] = 70 -> S[i] = 2 (preço válido para os dias 3 e 4)
Dia i = 5: P[i] = 60 -> S[i] = 1 (preço válido somente para hoje)
Dia i = 6: P[i] = 75 -> S[i] = 4 (preço válido para os dias 6, 5, 4 e 3)
Dia i = 7: P[i] = 85 -> S[i] = 6 (preço válido para os dias 7, 6, 5, 4, 3 e 2)

Para resolver o problema de forma eficiente, podemos usar um deque decrescente para armazenar os índices de preços de ações. O deque ajudará a encontrar o intervalo de preços em tempo O(n). A ideia é manter índices de preços de forma que os preços correspondentes estejam em ordem decrescente.

Implemente uma função para resolver o problema do intervalo de ações com complexidade O(n). Para isso implemente uma estrutura de dados deque (Double-Ended Queue). Considere n = 20 e P dado por:

P = [120, 100, 80, 90, 80, 95, 105, 100, 90, 80, 70, 90, 100, 115, 125, 110, 95, 90, 80, 100]

Mostre matematicamente qual é a complexidade computacional da função.

OBS: É preciso implementar um TAD/classe Deque a partir do zero.

4. 📘 Exercício: O Problema da Conexão das Cordas (2.5 pontos) (prático)
O problema do Custo Mínimo para Conectar Cordas é um problema clássico de otimização. O objetivo é conectar n cordas de comprimentos variados em uma única corda com o custo total mínimo. O custo de conectar duas cordas é igual à soma de seus comprimentos. Em cada etapa, combinamos duas cordas, e a corda combinada pode então ser usada em combinações subsequentes.

Definição do problema:
Dada uma matriz de comprimentos inteiros [1...n], onde cada inteiro representa o comprimento de uma corda, encontre o custo mínimo para conectar todas as cordas em uma única corda.

Ideia-chave:
Combinar duas cordas incorre em um custo igual à soma de seus comprimentos. Para minimizar o custo total, sempre combine as duas menores cordas em cada etapa.

Exemplo ilustrativo

Entrada: n = 4 (4 cordas de comprimentos 4, 3, 2 e 6)
lengths = [4, 3, 2, 6]

a) Combine as duas cordas menores (2 e 3):
Custo = 2 + 3 = 5

Cordas restantes = [4, 5, 6]

b) Combine as duas cordas menores (4 e 5):
Custo = 4 + 5 = 9

Cordas restantes = [6, 9]

c) Combine as duas cordas restantes (6 e 9):
Custo = 6 + 9 = 15

Cordas restantes = [15]
Custo total: 5 + 9 + 15 = 29

A estratégia é sempre combinar as duas menores cordas primeiro. Uma fila de prioridades é uma estrutura de dados ideal para esse propósito.

Passos

1. Inicializar uma fila de prioridades:
   - Inserir todos os comprimentos de corda na fila de prioridades.

2. Iterar até que apenas uma corda permaneça:
   - Remover as duas menores cordas da fila de prioridades.
   - Calcular o custo de combiná-las e adicionar esse custo ao custo total.
   - Inserir o comprimento de corda combinada de volta na fila de prioridades.

3. Retornar custo total:
   - Uma vez que a fila de prioridades tenha um único elemento, o custo total é a soma de todos os custos intermediários.

Implemente uma função para resolver o problema do Custo Mínimo para Conectar Cordas com 20 cordas de tamanhos utilizando uma fila de prioridades padrão. Para isso implemente uma Fila de Prioridades (em que o custo da operação dequeue é O(n)):

L = [20, 13, 15, 10, 19, 11, 8, 14, 17, 2, 9, 3, 18, 1, 16, 4, 12, 7, 5, 6]

Qual é o custo total obtido? Mostre matematicamente qual é a complexidade computacional da função.

OBS: É preciso implementar um TAD/classe FilaPrioridades a partir do zero.

Observação: em todos exercícios práticos, a resolução deve vir acompanhada de um link para execução do código fonte em ambiente virtual (onlineGDB ou Google Colab). O não envio do link acarretará em penalização da nota.

"Rejeição não significa que você não é bom o suficiente; significa que alguém falhou em perceber o que você tem a oferecer."
-- Mark Amend