

S T Q Q S S D

~~IX/~~

José Douglas Gondim Scorte, 4053417

Prava 2 - CANO  
(1.9)

2) O primeiro passo é ordenar os tarefas em ordem decrescente de multa. A lógica é: Essas tarefas possuem uma prioridade de execução maior para diminuir a multa total.

Agora, precisamos iterar o vetor de tarefas e para cada tarefa  $T_i$ , queremos alocá-la (se possível) em algum dia entre 1 e  $PT$  (é o prazo daquela tarefa). Vamos escolher a última horária disponível da última dia ainda disponível entre 1 e  $PT$ . (Podemos fazer isso usando um array de alocação de horárias e procurando nesse array disponibilidade de acordo com a condição de escolha mencionada). Ao final, retornaremos o array de horárias.

↳ Se não houver mais espaço nos dias 1... $PT$ ?

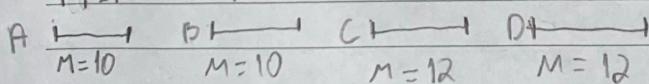
Argumenta: Esse algoritmo funciona porque sempre estamos tentando alocar uma tarefa que geraria uma multa maior caso atrasada, primeira. A reportagem está também na forma como a alocamos (na última horária disponível entre 1... $PT$ ). [Isso porque, dessa forma, ela vai atingir a mínima possível a execução de outras tarefas com multa menor que também não desejamos atingir.]

↳ demonstrar  
provar isso...

? Essa é a demonstração de o algo. funciona...

Para exemplificar: Em uma situação que podemos alocar 2 tarefas por dia:

$$PT=1 \quad P+=1 \quad P+=2 \quad PT=2$$



Se escolhessermos alocar as tarefas no primeiro horário disponível, iríamos executar C e D e servirmos feriados a atender A e B. V que não acontece se alocarmos C e D no último horário disponível. Assim, conseguimos executar todos as tarefas.

CONTINUA...

spiral®

S T A Q S S D

## (2) CONTINUAÇÃO

O exemplo mostra a importância da ordem de alocação após a ordenação inicial do vetor.

Para concluir: A nossa escolha é a última porque se existisse alguma tarifa  $T'$  que tornasse a multa total maior e ela tivesse sido cadastrada, então o vetor não estaria ordenado em ordem crescente de multa em vez estariamos escolhendo a intimação horária ainda disponível para aloca-la. ~~X~~ Fazendo então ficar alguma tarifa  $T_i$  da nossa seleção por  $T'$ .

3- a) O nosso problema inicial é contar um tronco de comprimento L em k partes, mas só podemos fazer um corte por vez. Logo, sempre queremos achar a melhor corte entre um ponto inicial e final do tronco. Percebemos que a cada corte, não ficamos com 2 troncos que podem ou não serem contados, e então acabamos com um problema semelhante ao original (achar uma forma ideal de cortar esses 2 troncos). Esse é a propriedade da subestrutura ótima (deslocando o problema em instâncias menores). Para concluir, somamos o custo do corte inicial com o custo mínimo dos 2 novos troncos e ao final teremos o custo mínimo para contar o tronco.

b)  $C(n) = C(p - \text{inicio}) + C(Fim - p) + (fim - \text{inicio})$

~~Explicação: P é onde o corte ocorre, restando创ar a  
franca idealmente na primeira parte (p-início) e na  
segunda parte (fim-p). Isto custa C para创ar francesas.  
Início e fim são variáveis que mudam.~~

## C1 VERSO

The image shows a page from a spiral-bound notebook with horizontal ruling lines. The text is written in red ink and is somewhat slanted and overlapping. It discusses different types of judgments or opinions. Key words and phrases include "jugement de valeur", "jugement de vérité", "jugement de justice", "l'opinion", "l'avis", "l'avis de fin de mission", and "l'avis de mission". There is a yellow vertical strip along the right edge of the page.

O que é o custo de um intervalo? Minimizar?

c) int custoMinimo(int inicio, int fim, int p){  
 inicio < fim  
 if (fim - inicio == 0) return 0;  
 int custo = custoMinimo(inicio, p) + custoMinimo(p, fim);  
 mem[inicio][fim] = custo;  
 return custo;

1- a) Temos 2 caixas A e B (contra-exemplo)

A  $\boxed{4}$   $P=4$   
 $K=100$

B  $\boxed{2}$   $P=2$   
 $K=1$

$$4 \cdot 100 + 1 \cdot (2+1)$$

Ordenando por peso, a caixa A ficaria encima da da caixa B, o que gera esforço  $100 \cdot (4+2) = 600$ .

Mas se colacarmos a caixa B encima da A, temos esforço  $1 \cdot (4+2) = 6$ , (esforço é menor). Mas trado o contra-exemplo.

b) O esforço é calculado pelo número de objetos da caixa mais superior multiplicado pela somatória dos pesos das caixas que estão embaladas, incluída a caixa atual.

Logo, a única forma de diminuir o esforço é diminuir a constante do número de objetos da caixa mais superior. É como os caixas estão em ordem decrescente de K, e menor K está encima o esforço já é o mínimo.

Nbs: Eu estou achando essa questão muito extensa e não estou vendo a semelhança com o problema da mochila do fumô como foi explicado na aula.

spiral® CONTINUA a Obs...

S T Q Q S S D

## CONTINUAÇÃO do Obs:

A minha interpretação é diferente, mas decidi fazer como foi explicado na aula. Eu acredite que o numero de objetos deveria influenciar no peso da caixa, de forma que a solução seria diferente. A não ser que o problema da mochila falada na sala seja a prioritária.

De qualquer forma, eu decidi fazer como foi explicado na aula de exercícios, onde o fator é a quantidade de objetos na caixa mais superior vezes a somatória dos pesos das caixas inferiores, incluindo a caixa de topo.

→ SIM

Y  
divisor  
varia, sempre

