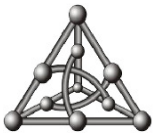


UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
Faculdade de Computação

# Implementação Algorítmica

## Atividade 2 — Problema do corte da tora

## 1 Descrição

Uma empresa compra longas toras de madeira e as corta em pedaços menores para revenda. Os cortes são realizados de tal modo que os pedaços resultantes têm sempre comprimento inteiro. Um corte não tem valor e não é cobrado pela empresa. Para cada $i = 1, 2, \ldots$, a empresa cobra o preço $p_i$ por uma tora de comprimento $i$.

Como exemplo, suponha que temos uma tora de comprimento $10$ metros com os valores dos seus pedaços de $1$ a $10$ metros dados pela seguinte tabela:

| comprimento $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| preço $p_i$ | 1 | 5 | 8 | 9 | 10 | 17 | 17 | 20 | 24 | 30 |

Dessa forma, temos o seguinte:

> **Problema do corte da tora de madeira**: Dadas uma tora de $n$ metros de comprimento e uma tabela de preços $p_i$ para $i = 1, 2, \ldots, n$, determine o valor máximo de venda $r_n$ que é possível obter pelo corte da tora e venda de seus pedaços.

Um exemplo para uma tora de $n = 4$ metros é mostrado na Figura 1.

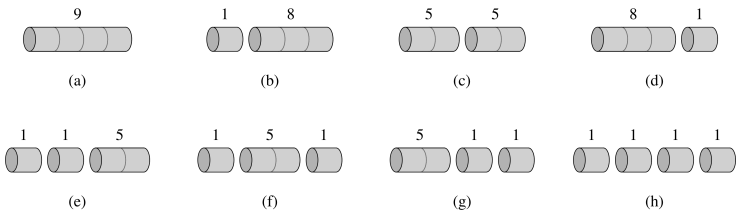
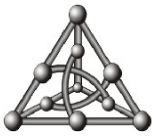



Figura 1: Exemplo com todos os cortes possíveis para uma tora de comprimento $n = 4$ e preços de venda dados pela tabela acima. Observe que o valor máximo de venda é obtido quando cortamos a tora em dois pedaços de 2 metros cada (letra (c)) e obtemos o valor de venda $p_2 + p_2 = 5 + 5 = 10$.

Nesta atividade, você tem de implementar dois algoritmos para o problema do corte das toras de madeira:

- um usando **programação dinâmica** (um dos algoritmos visto em aula - *bottom up* ou *top down with memoization*) e;
- outro usando uma **estratégia gulosa** (*greedy*) descrita a seguir.





**Estratégia gulosa para o problema do curte da tora de madeira:** Defina a **densidade** de uma tora de comprimento $i$ como $p_i/i$, isto é, seu valor por metro. A estratégia gulosa para uma tora de comprimento $n$ corta um primeiro pedaço de comprimento $i$, onde $1 \leq i \leq n$, com densidade máxima. Continua-se então aplicando a estratégia gulosa ao pedaço remascente de comprimento $n - i$.

A estratégia gulosa nem sempre encontra a solução ótima para este problema. De fato, o Exercício 15.1-2 do livro CLRS[1] pede para mostrar um contra-exemplo em que esta estratégia falha ao tentar encontrar uma solução ótima.

## 2 Programa, entrada e saída

Você deve desenvolver e implementar os algoritmos conforme a descrição da Seção 1.

Considere os seguintes parâmetros para execução dos experimentos, que são fornecidos como entrada:

- **`inc`** é o tamanho inicial da entrada;
- **`fim`** é o tamanho final;
- **`stp`** é o intervalo entre dois tamanhos de entrada.

Você deve construir conjuntos de dados de entrada (pseudo)aleatórios da seguinte forma. Por exemplo, para **`inc`** $= 1000$, **`fim`** $= 20000$ e **`stp`** $= 1000$ os comprimentos devem ser $n = 1000, 2000, 3000, \ldots, 20000$. Para cada valor $n$ do comprimento de uma tora, você deve determinar os preços de venda dos pedaços da tora de comprimentos inteiros de 1 a $n$. Limite os preços dos pedaços de uma tora no intervalo $[1, n]$. O vetor de preços deve estar ordenado em ordem não decrescente. Depois disso, você deve executar os algoritmos propostos, mostrando o valor total de venda obtidos por cada um e registrar seus tempos de execução. Ao final, mostre a porcentagem de acerto que a solução gulosa obtém em relação à solução da programação dinâmica.

### 2.1 Exemplo de entrada e saída

Um pequeno exemplo, para casos de teste com $n = 1000, 2000, \ldots, 20000$ é mostrado a seguir. Os tempos de execução dos algoritmos são mostrados em segundos. Os valores `vDP` e `tDP` indicam o valor total da venda obtido pelo algoritmo de programação dinâmica e o tempo de execução do algoritmo em segundos, respectivamente. Os valores `vGreedy` e `tGreedy` fazem as mesmas referências ao algoritmo guloso. Na última coluna temos a porcentagem de acerto da solução gulosa em relação à solução da programação dinâmica.

[1] *Introduction to Algorithms 3rd Edition*, Cormen *et al.*, MIT Press 2009.

| n | vDP | tDP | vGreedy | tGreedy | % |
|---|---|---|---|---|---|
| 1000 | 1010 | 0.002483 | 1001 | 0.000012 | 99.11 |
| 2000 | 4000 | 0.009958 | 4000 | 0.004503 | 100.00 |
| 3000 | 3220 | 0.004706 | 3001 | 0.000007 | 93.20 |
| 4000 | 12000 | 0.011367 | 12000 | 0.006983 | 100.00 |
| 5000 | 15000 | 0.009384 | 15000 | 0.010566 | 100.00 |
| 6000 | 12000 | 0.013527 | 12000 | 0.007746 | 100.00 |
| 7000 | 9333 | 0.019007 | 7008 | 0.000009 | 75.09 |
| 8000 | 13332 | 0.024973 | 8002 | 0.000013 | 60.02 |
| 9000 | 45000 | 0.032246 | 45000 | 0.034783 | 100.00 |
| 10000 | 11818 | 0.039680 | 10002 | 0.000014 | 84.63 |
| 11000 | 22000 | 0.047734 | 22000 | 0.054955 | 100.00 |
| 12000 | 36000 | 0.054486 | 36000 | 0.060317 | 100.00 |
| 13000 | 39000 | 0.065371 | 39000 | 0.071316 | 100.00 |
| 14000 | 42000 | 0.081981 | 42000 | 0.084247 | 100.00 |
| 15000 | 30000 | 0.090125 | 30000 | 0.097949 | 100.00 |
| 16000 | 80000 | 0.102366 | 80000 | 0.112413 | 100.00 |
| 17000 | 28332 | 0.116382 | 17005 | 0.000019 | 60.02 |
| 18000 | 72000 | 0.131464 | 72000 | 0.141133 | 100.00 |
| 19000 | 38000 | 0.144960 | 38000 | 0.159061 | 100.00 |
| 20000 | 40000 | 0.161872 | 40000 | 0.174811 | 100.00 |

## 3 Entrega

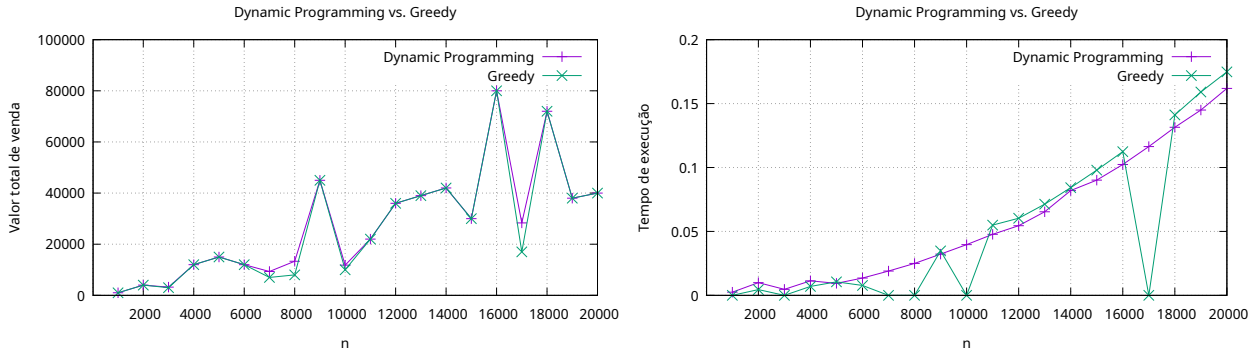
A atividade deve ser feita, preferencialmente, em duplas. Na primeira linha do código-fonte, coloque o nome completo de ambos como comentário. Apenas um dos integrantes da dupla submete a atividade no AVA.

Instruções para entrega da sua atividade:

1. **O que entregar?**

   Um arquivo compactado a ser entregue deve conter o seguinte:

   - Programa desenvolvido;
   - Tabela dos valores dos cortes das toras e tempos de execução dos algoritmos (em texto puro, por exemplo, .txt), de acordo com a formatação apresentada na Seção 2.1, e
   - Um relatório em `.pdf` contendo gráficos gerados a partir das tabelas dos valores e dos tempos de execução dos algoritmos. Exemplos relativos à tabela da Seção 2.1 usando o software `gnuplot` são mostrados na Figura 2. Você pode utilizar qualquer outro software de sua preferência para gerar os gráficos. No relatório, faça uma discussão sobre os resultados obtidos.




(a) Valor total de venda.

(b) Tempo de execução dos algoritmos.

Figura 2: Gráficos da execução dos algortimos do corte a tora de madeira para o caso de teste com **`inc`** $= 1000$, **`fim`** $= 20000$ e **`stp`** $= 1000$.

Compacte todos esses arquivos com o compactador de sua preferência e entregue um único arquivo (com extensão **`.tgz`**, **`.bz2`**, **`.zip`**, **`.rar`**, ...).

2. **Forma de entrega** A entrega será realizada diretamente no Sistema (AVA/UFMS), na disciplina de Implementação Algorítmica – T01. Você pode entregar a atividade quantas vezes quiser até às **23 horas e 59 minutos** do dia **29 de outubro de 2024**. A última versão entregue é aquela que será corrigida. Encerrado o prazo, trabalhos não serão mais aceitos.

3. **Linguagem de programação**

   O programa deve ser implementado em uma das seguintes linguagens: C/C++, Python ou Java.

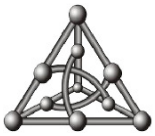
4. **Erros**

   Trabalhos com erros de compilação/interpretação receberão nota ZERO. Faça todos os testes necessários para garantir que seu programa está livre de erros de compilação/interpretação.

5. **Critérios de correção**

   Serão atribuídos até $5{,}0$ pontos para a implementação e até $5{,}0$ pontos para o relatório. Possíveis penalizações que podem diminuir a pontuação:

   - Implementação incorreta;
   - Códigos mal escritos ou desorganizados. Por exemplo, códigos com trechos repetitivos que executam a mesma tarefa;
   - Realizar os experimentos de maneira incorreta;
   - Gráficos pouco legíveis;
   - Incluir informações incorretas no relatório;
   - Não entregar algo que foi solicitado.





6. **Arquivo com o programa fonte**

   Seu(s) arquivo(s) contendo o(s) fonte(s) do(s) programa(s) na linguagem escolhida deve(m) estar bem organizado(s). Um programa tem de ser muito bem compreendido por uma pessoa. Verifique se seu programa tem a indentação adequada, se não tem linhas muito longas, se tem variáveis com nomes significativos, entre outros. Não esqueça que um programa bem descrito e bem organizado é a chave de seu sucesso.

7. **Conduta Ética**

   O trabalho deve ser feito INDIVIDUALMENTE/COM SEU GRUPO. Cada estudante tem responsabilidade sobre cópias de seu trabalho, mesmo que parciais. Não compartilhe seu programa ou trechos de seu programa. Você pode consultar seus colegas para esclarecer dúvidas e discutir idéias sobre o trabalho, ao vivo ou no fórum de discussão da disciplina, mas NÃO copie o programa!

   Trabalhos considerados plagiados terão nota ZERO.