



## — Atividade Prática 2 —

Nome: \_\_\_\_\_ Matrícula: \_\_\_\_\_

- Esta atividade avaliativa deve ser feita em grupos de até 3 pessoas.
- É permitida a consulta somente ao Campus Virtual, materiais impressos, e os seguintes sites:

- PuLP: <https://pypi.org/project/PuLP/>.
- Gurobi: <https://www.gurobi.com/>.
- Python: <https://www.python.org/>.

- Caso prefiram utilizar o Google Colab, peço para que desabilitem o auxílio por IA: <https://cursos.alura.com.br/forum/topico-duvida-como-desativar-a-auto-complementacao-do-google-colab-450291>.
- Qualquer tentativa de fraude acarretará em nota nula.
- Os três códigos .py (ou .ipynb) serão inseridos no Campus Virtual.

Boa atividade de implementação!

**Questão 1 - Planejamento da merenda:** Uma escola infantil promove a participação das famílias (\_\_\_\_\_/33%) por meio de um revezamento na preparação das merendas. Em cada dia de aula, exatamente **uma família** é responsável por preparar a merenda de toda a turma. A turma tem 10 alunos, cada um pertencente a uma família distinta, e o horizonte de planejamento é de **30 dias corridos**, começando em uma segunda-feira.

Não há aula aos sábados e domingos, de modo que esses dias não precisam ser cobertos. Assim, há um total de **22 dias de aula** no horizonte (segunda a sexta de cada semana).

Cada família escolhe três dias da semana (*segunda* a *sexta*) nos quais tem disponibilidade para levar a merenda. Esses dias são classificados por ordem de preferência, com pesos:

mais preferido = 100,    médio = 10,    menos preferido = 1.

Se um dia da semana não foi escolhido pela família, então ela **não pode ser escalada** em nenhum dia correspondente. Além disso, se uma família levar a merenda em um dia, ela só poderá levar novamente **após pelo menos 10 dias corridos**.

Formule um **modelo de programação inteira binária** que:

- Escolha, para cada dia de aula, qual família ficará responsável pela merenda;
- Garanta que cada dia de aula tenha exatamente uma família responsável;
- Respeite as restrições de disponibilidade e o intervalo mínimo de 10 dias entre duas designações da mesma família;
- Maximize a soma dos pesos de preferência das famílias atribuídas aos seus dias.

A instância consiste em um horizonte de 30 dias (dia 1 = segunda-feira) contém 22 dias de aula:

$$T = \{1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 15, 16, 17, 18, 19, 22, 23, 24, 25, 26, 29, 30\}.$$

As preferências das 10 famílias são apresentadas abaixo. Cada família escolheu três dias da semana distintos, classificados como 1<sup>a</sup>, 2<sup>a</sup> e 3<sup>a</sup> preferência (pesos 100, 10 e 1, respectivamente).

| Família | 1 <sup>a</sup> preferência | 2 <sup>a</sup> preferência | 3 <sup>a</sup> preferência |
|---------|----------------------------|----------------------------|----------------------------|
| 1       | Seg                        | Qua                        | Sex                        |
| 2       | Ter                        | Qui                        | Seg                        |
| 3       | Qua                        | Ter                        | Sex                        |
| 4       | Seg                        | Ter                        | Qui                        |
| 5       | Qui                        | Sex                        | Seg                        |
| 6       | Ter                        | Qua                        | Qui                        |
| 7       | Sex                        | Qua                        | Ter                        |
| 8       | Seg                        | Qui                        | Sex                        |
| 9       | Qua                        | Seg                        | Ter                        |
| 10      | Ter                        | Sex                        | Qui                        |

Os pesos de preferência (para  $w \in \{\text{Seg, Ter, Qua, Qui, Sex}\}$ ) são:

| Família | Seg | Ter | Qua | Qui | Sex |
|---------|-----|-----|-----|-----|-----|
| 1       | 100 | 0   | 10  | 0   | 1   |
| 2       | 1   | 100 | 0   | 10  | 0   |
| 3       | 0   | 10  | 100 | 0   | 1   |
| 4       | 100 | 10  | 0   | 1   | 0   |
| 5       | 1   | 0   | 0   | 100 | 10  |
| 6       | 0   | 100 | 10  | 1   | 0   |
| 7       | 0   | 10  | 10  | 0   | 100 |
| 8       | 100 | 0   | 0   | 10  | 1   |
| 9       | 10  | 10  | 100 | 0   | 0   |
| 10      | 0   | 100 | 0   | 10  | 1   |

Intervalo mínimo entre duas designações consecutivas da mesma família:  $L = 10$  dias. Complete o código *merenda.py*.

**Questão 2 - Modelagem lúdica:** No aplicativo *Thermometer Puzzles*, uma grelha com  $n \times n$  células (\_\_\_\_\_/33%)

é preenchida com termômetros. Cada termômetro ocupa duas ou mais células consecutivas, numa linha ou numa coluna (ver Figura 1). O jogador deve determinar quais células estão preenchidas com mercúrio. Para cada linha e para cada coluna, é dado o número de células preenchidas. Além disso, deve-se levar em conta que os termômetros são sempre preenchidos de baixo para cima (o fundo do termômetro é indicado por um círculo). Isso implica que, se uma célula estiver preenchida com mercúrio, então todas as células abaixo dela, pertencentes ao mesmo termômetro, também devem estar preenchidas. Do mesmo modo, se uma célula não estiver preenchida, então todas as células acima dela no mesmo termômetro não devem estar preenchidas.

Apresente um modelo matemático para esse problema, que resolva a instância da Figura 1, levando em conta as quantidades de linhas e colunas que devem ser preenchidas de acordo com as duas tabelas abaixo. Complete o código *termometros.py*.

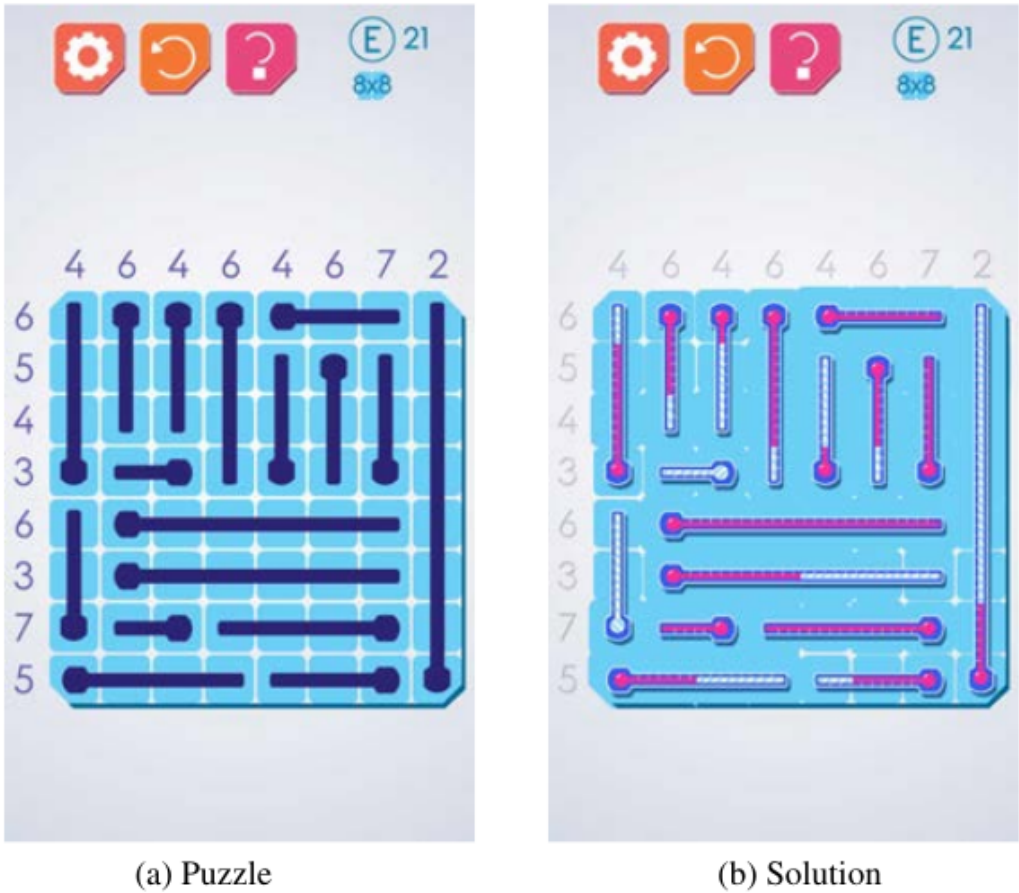


Figura 1: Jogo Thermometer Puzzles.

Tabela 1: Quantidade de células com mercúrio em cada linha e coluna do puzzle da Figura 1

| Coluna             | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|--------------------|---|---|---|---|---|---|---|---|
| Mercúrio (células) | 4 | 6 | 4 | 6 | 4 | 6 | 7 | 2 |

| Linha              | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|--------------------|---|---|---|---|---|---|---|---|
| Mercúrio (células) | 6 | 5 | 6 | 7 | 6 | 7 | 6 | 5 |

**Questão 3 - Branch & Bound:** O objetivo dessa questão é compreender e executar o algoritmo (\_\_\_\_\_/34%)

**Branch and Bound** para o problema da mochila 0-1, buscando maximizar o valor total sem ultrapassar a capacidade da mochila.

Você deve selecionar itens para carregar em uma mochila com **capacidade máxima de 10 kg**. Os itens disponíveis são:

| Item | Peso ( $w_i$ ) | Valor ( $v_i$ ) |
|------|----------------|-----------------|
| 1    | 2 kg           | R\$ 6           |
| 2    | 5 kg           | R\$ 20          |
| 3    | 3 kg           | R\$ 12          |
| 4    | 4 kg           | R\$ 15          |

Complete o código *bb.py* e execute o algoritmo **Branch and Bound** para resolver o problema da mochila com os dados fornecidos.