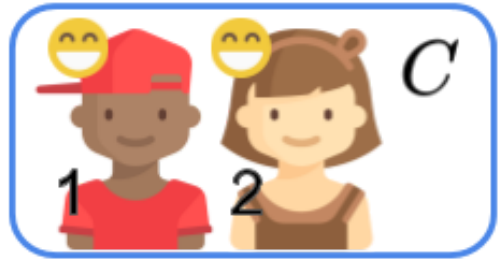
# Modelo Matemático

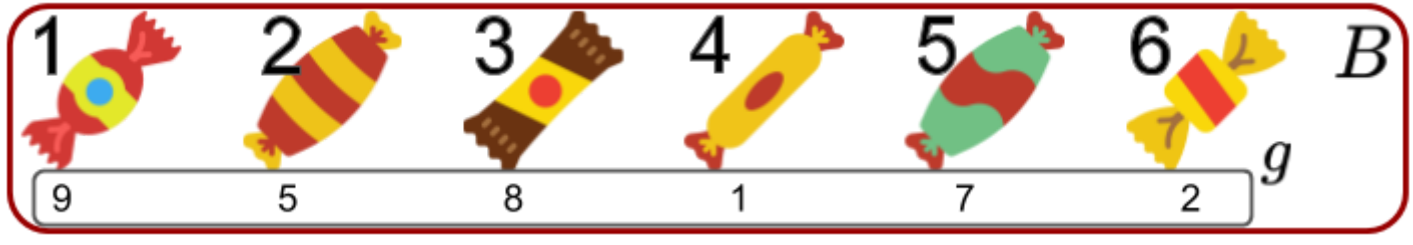
$$\begin{aligned} \min \quad & (\sum_B g_i x_{i1} - \sum_B g_i x_{i2})^2 \\ s.t. \quad & x_{i1} + x_{i2} = 1, \forall i \in B \\ & x_{ij} = \{0, 1\}, \forall i \in B \wedge \forall j \in C \end{aligned}$$

In [1]:
```
import pyomo.environ as poe
modelo = poe.AbstractModel()
```




Conjunto representando as crianças:

In [2]:
```
modelo.criancas = poe.RangeSet(1,2)
```




Conjunto representando os bombons:

In [3]:
```
modelo.bombons = poe.Set()
```

Nivel de gostosura dos bombons:

In [4]:
```
modelo.gostosura = poe.Param(modelo.bombons, within=poe.NonNegativeReals)
```

Variáveis de decisão:

$$x_{11}\ x_{21}\ x_{31}\ x_{41}\ x_{51}\ x_{61}$$
$$x_{12}\ x_{22}\ x_{32}\ x_{42}\ x_{52}\ x_{62}$$

- 1, se o bombom i for para a criança j
- 0, caso contrário

In [5]:
```
modelo.x = poe.Var(modelo.bombons, modelo.criancas, within=poe.Binary)
```

Função objetivo:

$$\min \quad (\sum_B g_i x_{i1} - \sum_B g_i x_{i2})^2$$

In [6]:
```
def funcao_objetivo(modelo):
    return((sum(modelo.gostosura[i] * modelo.x[i, 1] for i in modelo.bombons))
           - sum(modelo.gostosura[i] * modelo.x[i, 2] for i in modelo.bombons
))**2

modelo.OBJ = poe.Objective(rule=funcao_objetivo)
```

Restrições:

$$s.t. \quad x_{i1} + x_{i2} = 1, \forall i \in B$$

In [7]:
```
def funcao_restricao_de_cobertura(modelo, i):
    return modelo.x[i, 1] + modelo.x[i, 2] == 1

modelo.restricao_de_cobertura = \
    poe.Constraint(modelo.bombons, rule=funcao_restricao_de_cobertura)
```

In [8]:
```
! cat candy_box_problem_instance.dat
```

```
# AMPL format

set bombons := 1 2 3 4 5 6;

param gostosura :=
1 9
2 5
3 8
4 1
5 7
6 2
;
```

In [9]:
```
!pyomo solve candy_box_problem.py candy_box_problem_instance.dat --solver=bonmin
```

```
[    0.00] Setting up Pyomo environment
[    0.00] Applying Pyomo preprocessing actions
[    0.00] Creating model
[    0.01] Applying solver
[    0.05] Processing results
    Number of solutions: 1
    Solution Information
      Gap: None
      Status: optimal
      Function Value: 0.0
    Solver results file: results.yml
[    0.05] Applying Pyomo postprocessing actions
[    0.05] Pyomo Finished
```

In [10]:
```
!grep -B 1 -A 99 "#   Solution Information" results.yml
```

```
# ------------------------------------------------------------
#   Solution Information
# ------------------------------------------------------------
Solution:
- number of solutions: 1
  number of solutions displayed: 1
- Gap: None
  Status: optimal
  Message: bonmin\x3a Optimal
  Objective:
    OBJ:
      Value: 0
  Variable:
    x[1,2]:
      Value: 1
    x[2,2]:
      Value: 1
    x[3,1]:
      Value: 1
    x[4,1]:
      Value: 1
    x[5,1]:
      Value: 1
    x[6,2]:
      Value: 1
  Constraint: No values
```