Preparação do trabalho

- Para o trabalho é necessário instalar Julia: https://julialang.org,
- e instalar os pacotes necessários em Julia

```
> <caminho>/bin/julia
julia> import Pkg
julia> Pkg.add("JuMP")
julia> Pkg.add("GLPKMathProgInterface")
julia> Pkg.add("IJulia")
```

- Caso não já está instalado, também é necessário instalar o GNU Linear Programming Kit (GLPK)
  - Windows: Baixar e instalar o GNU Linear Programming Kit (GLPK) em http://gnuwin32.sourceforge.net/packages/glpk.htm.
  - Linux: `sudo apt-get install glpk`
- A documentação de Julia/JuMP está disponível em http://www.juliaopt.org/JuMP.jl/stable.

Exemplo

Para demonstrar a especificação de um problema nos formatas CPLEX lp e GNU mathprog, considere o exemplo de um importador de Whisky:

Um importador de Whisky tem as seguintes restrições de importação

- no máximo 2000 garrafas de *Johnny Ballantine* por 70 R$ cada uma,
- no máximo 2500 garrafas de *Old Gargantua* por 50 R$ cada uma,
- no máximo 1200 garrafas de *Misty Deluxe* por 40 R$ cada uma.

Dos Whiskies importados ele produz três misturas A, B, C, que ele vende por 68 R$, 57 R$ e 45 R$, respectivamente. As misturas são

- A: no mínimo 60% Johnny Ballantine, no máximo 20% Misty Deluxe,
- B: no mínimo 15% Johnny Ballantine, no máximo 60% Misty Deluxe,
- C: no máximo 50% Misty Deluxe.

Quais seriam as misturas ótimas, e quantas garrafas de cada mistura devem ser produzidas para maximizar o lucro? Formule como programa linear.

Observações:

- Use como variáveis o número de garrafas $x_{m,i}$ da marca m usadas na mistura i.
- Desconsidere a integralidade das garrafas.

Especificação em Julia

http://nbviewer.jupyter.org/url/www.inf.ufrgs.br/~mrpritt/oc/whisky.ipynb
http://www.inf.ufrgs.br/~mrpritt/oc/whisky.jl
A solução é

```
using JuMP
using GLPK
using Formatting

I=collect(1:3); V=collect(1:3);

m = Model()
set_optimizer(m, GLPK.Optimizer);
@variable(m, x[i in I, v in V] >= 0);

pv=[68, 57, 45];

ci=[70, 50, 40];

@variable(m, y[v in V]) ## total vendas
@variable(m, z[i in I]) ## total importação
@objective(m, Max, sum(pv[v]*y[v] for v in V)-sum(ci[i]*z[i]
   for i in I))

@constraint(m, [v in V], y[v]==sum(x[i,v] for i in I))
@constraint(m, [i in I], z[i]==sum(x[i,v] for v in V));

li=[2000,2500,1200];

@constraint(m, [i in I], z[i] <= li[i]);

@constraint(m,x[1,1] >= 0.6 *y[1]) ## no mínimo 60% Johnny
   Ballantine em A
@constraint(m, x[3,1] <= 0.2 *y[1]) ## no máximo 20% Misty
   Deluxe em A

@constraint(m, x[1,2] >= 0.15*y[2]) ## no mínimo 15% Johnny
   Ballantine em B
@constraint(m, x[3,2] <= 0.6 *y[2]) ## no máximo 60% Misty
   Deluxe em B

@constraint(m, x[3,3] <= 0.6 *y[3]); ## no máximo 50% Misty
   Deluxe em C
```

```
println(m)
optimize!(m)

printfmt("O lucor máximo é {:.2f}.\n",objective_value(m))
printfmt("Importar {:.2f} garrafas de Johnny Ballantine, {:.2f
   } garrafas de Old Gargantua, e {:.2f} garrafas de Misty
   Deluxe.",value(z[1]),value(z[2]),value(z[3]) )
for v in V
    printfmt("Mistura {}: {:.2f} garrafas.\n","ABC"[v],value(y
       [v]))
    if value(y[v])>0
      printfmt("  {:.2f}% JB, {:.2f}% OG, {:.2f}% MD.\n","ABC"
         [v],value(x[1,v])/value(y[v]),value(x[2,v])/value(y[v
         ]),value(x[3,v])/value(y[v]))
    end
end
```

Laboratório

Questão 1 (Rações, fácil)

Um fabricante de rações quer determinar a fórmula mais econômica de uma certa ração. A composição nutritiva dos ingredientes disponíveis no mercado e os seus custos são os seguintes:

	Ingredientes		
Nutrientes	Soja (%)	Milho (%)	Cana (%)
Cálcio	0,2	1	3
Proteína	50	9	0
Carboidratos	0,8	2	2
Custo/quilo	15,00	20,00	8,00

O fabricante deve entregar 1000 quilos de ração por dia e garantir que esta contenha:

no máximo (%)	no mínimo (%)	de
1,2	0,8	Cálcio
-	22	Proteína
20	-	Carboidratos

Questão 2 (Atlântico gauchês)

Um grupo de investores construindo o novo hotel “Atlântico gauchês” tem que definir a distribuição dos quartos. Eles querem pelo menos 20 quartos simples, 35 quartos duplos e 10 suítes. Um quarto simples precisa $10\,m^2$, um quarto duplo $18\,m^2$ e uma suíte

$25\,m^2$. Em total $950\,m^2$ são disponíveis. O restaurante do hotel tem capacidade para 100 pessoas, e a gerência considera ocupações de uma, duas e quatro pessoas por quarto simples, quarto duplo e suíte, respectivamente. Os investores estimam que um quarto simples rende R$ 42000 por ano, um quarto duplo R$ 76000 e uma suíte R$ 138000 por ano.
Formule um programa linear, que determine o número de quartos de cada tipo que maximiza o lucro (não considerando restrições de integralidade).

Questão 3 (Refinar óleo (da Costa))

Um certo óleo é refinado a partir da mistura de outros óleos, vegetais ou não vegetais. Temos óleos vegetais V1 e V2 e óleos não vegetais NV1 NV2 NV3. Por restrições da fábrica, um máximo de 200 toneladas de óleos vegetais podem ser refinados por mês, e um máximo de 250 toneladas de óleos não vegetais. A acidez do óleo desejado deve estar entre 3 e 6 (dada uma unidade de medida) e a acidez depende linearmente das quantidades/acidez dos óleos brutos usados. O preço de venda de uma tonelada do óleo é R$ 150. Calcule a mistura que maximiza o lucro, dado que:

Óleo	V1	V2	NV1	NV2	NV3
Custo/ton	110	120	130	110	115
Acidez	8,8	6,1	2,0	4,2	5,0

Formule e resolve este problema com Julia/JuMP.

Questão 4 (Companhia de torneiras)

Formule e resolve o problema das torneiras.

- Define valores dos dados mencionados.
- Implemente o modelo em Julia/JuMP e resolve.

Criando iPython notebooks com Julia A forma mais simples de trabalhar com a Julia é usar um Notebook. O primeiro passo é rodar um servidor para notebooks na máquina local:

```
> <caminho>/bin/julia
julia> using IJulia
julia> notebook()
```

Isso abre um navegador no browser.
Um Notebook consiste de uma séria de células. Um célula pode conter texto (markup) ou código. Células com código podem ser executadas diretamente no Notebook e o resultado aparece no Notebook.
Em `http://nbviewer.jupyter.org/url/www.inf.ufrgs.br/~mrpritt/oc/oils.ipynb` tem um notebook visualizado. Vocês podem baixar o Notebook e gravar no diretório do navegador. O Notebook vai aparecer no navegador, e vocês podem abri-lo e rodar. Tem várias possibilidades, por exemplo vocês podem rodar tudo com Cell > Run All, ou célula por célula por clicar em Run ou usar ⇧+Enter.