Postavka Julie u Google Colab

Potrebno je preuzeti ovaj notebook, upload-ovati ga i pokrenuti kôd ispod prilikom kreiranja notebook-a za sve vježbe. Nakon pokretanja, potrebno je restartovati notebook. U *Runtime* kliknite na *Change runtime type* i odaberite Juliu i GPU. Sada možete koristiti Juliu za sve što radite.

U slučaju da vam krene izbacivati greške radi gubitka trenutne sesije izvršite sve naredbe opet i restartujte notebook.

```
!curl -sSL "https://julialang-s3.julialang.org/bin/linux/x64/1.5/julia-1.5.2-linux
!tar -xzf julia.tar.gz -C /usr --strip-components 1
!rm -rf julia.tar.gz*
!julia -e 'using Pkg; pkg"add IJulia; precompile"'
!echo "DONE"
```

```
 Installing known registries into `~/.julia`
###################################################################### 10
      Added registry `General` to `~/.julia/registries/General`
  Resolving package versions...
  Installed ZeroMQ_jll ──────── v4.3.2+5
  Installed Artifacts ───────── v1.3.0
  Installed MbedTLS_jll ─────── v2.16.8+1
  Installed VersionParsing ──── v1.2.0
  Installed ZMQ ─────────────── v1.2.1
  Installed IJulia ──────────── v1.23.0
  Installed Parsers ─────────── v1.0.12
  Installed MbedTLS ─────────── v1.0.3
  Installed Conda ───────────── v1.5.0
  Installed SoftGlobalScope ─── v1.1.0
  Installed JLLWrappers ─────── v1.1.3
  Installed JSON ────────────── v0.21.1
Downloading artifact: ZeroMQ
###################################################################### 10
Downloading artifact: MbedTLS
###################################################################### 10
Updating `~/.julia/environments/v1.5/Project.toml`
  [7073ff75] + IJulia v1.23.0
Updating `~/.julia/environments/v1.5/Manifest.toml`
  [56f22d72] + Artifacts v1.3.0
  [8f4d0f93] + Conda v1.5.0
  [7073ff75] + IJulia v1.23.0
  [692b3bcd] + JLLWrappers v1.1.3
  [682c06a0] + JSON v0.21.1
  [739be429] + MbedTLS v1.0.3
  [c8ffd9c3] + MbedTLS_jll v2.16.8+1
  [69de0a69] + Parsers v1.0.12
  [b85f4697] + SoftGlobalScope v1.1.0
  [81def892] + VersionParsing v1.2.0
  [c2297ded] + ZMQ v1.2.1
  [8f1865be] + ZeroMQ_jll v4.3.2+5
  [2a0f44e3] + Base64
  [ade2ca70] + Dates
  [8ba89e20] + Distributed
  [7b1f6079] + FileWatching
```

```
[b77e0a4c] + InteractiveUtils
[76f85450] + LibGit2
[8f399da3] + Libdl
[56ddb016] + Logging
[d6f4376e] + Markdown
[a63ad114] + Mmap
[44cfe95a] + Pkg
[de0858da] + Printf
[3fa0cd96] + REPL
[9a3f8284] + Random
[ea8e919c] + SHA
[9e88b42a] + Serialization
[6462fe0b] + Sockets
[8dfed614] + Test
[cf7118a7] + UUIDs
[4ec0a83e] + Unicode
 Building Conda ─→ `~/.julia/packages/Conda/x5ml4/deps/build.log`
 Building IJulia → `~/.julia/packages/IJulia/ljYVo/deps/build.log`
Precompiling project...
```

LAB. VJEŽBA 45

```
using LinearAlgebra
function rijesi_simplex(A, b, c)
    if(A==nothing||b==nothing||c==nothing||size(b,1)!=size(A,1)||size(c,2)!=size(A
            error("Doslo je do greske!")
    end
    pomb=Array{Float64}(b)
    push!(pomb,0)
    simplex=hcat(pomb,vcat(A,c));
    J=vcat(Matrix(I,size(A,1),size(b,1)),zeros(1,size(A,1)))
    simplex=hcat(simplex,J)
    vrijednostVarijabliX=Array{Float64,1}()

    while(true)
    iteracija=rijesi_iteraciju(simplex)
        if(iteracija==-1)
        break
        end
    end

    for i=2:size(simplex,2)
        provjera1=false
        for j=1:size(simplex,1)-1
            provjera2=true
            if(simplex[j,i]==1)
                for k=1:size(simplex,1)-1
                    if(k!=j && simplex[k,i]!=0)
                    provjera2=false
                    end
                end
                if(provjera2==true)
                push!(vrijednostVarijabliX,simplex[j,1])
                provjera1=true
                j=size(simplex,1)-1
```

```
                    end
                end
            end
             if(provjera1==false)
                push!(vrijednostVarijabliX,0)
                end

        end
        display(simplex)
        return simplex[size(simplex,1),1]*(-1),vrijednostVarijabliX

    end

    function rijesi_iteraciju(simplex)
        najveciURedu=0
        k=1
        kolona=0
        x=0
           for x in simplex[size(simplex,1),2:end]
            if(x > najveciURedu)
                    najveciURedu=x;
                    kolona=k
                end
                k=k+1
            end
            kolona=kolona+1

        if (najveciURedu==0)
          return -1
        end

        najveciUKoloni=-1
            najveciUKoloni,index1=findmax(simplex[1:end-1,kolona])

        if (najveciUKoloni<=0)
           return -1
        end

        i=1;
        minKol=100000
        imin=0
        while(i<=size(simplex,1)-1)
            if(simplex[i,1]/simplex[i,kolona]<minKol && simplex[i,1]/simplex[i,kolona]>
            minKol=simplex[i,1]/simplex[i,kolona]
            imin=i
            end
            i=i+1
        end

        pivot= simplex[imin,kolona]

        for i=1:size(simplex,1)
            if(i==imin)
            for j=1:size(simplex,2)
                simplex[i,j]=simplex[i,j]*(1/pivot)
            end
```

```
            end
            end
        end

        for i=1:size(simplex,1)
            el=simplex[i,kolona]
            if(i!=imin)
            for j=1:size(simplex,2)
                simplex[i,j]=simplex[i,j]-(el*simplex[imin,j])
            end
            end
        end
        return simplex
    end
```

```
     rijesi_iteraciju (generic function with 1 method)
```

PRIMJER 1

```
c = [150 40]
A = [3 0; 0 2; 9 4]
b = [36; 54; 144 ]
optimalnoZ,vrijednostiX=rijesi_simplex(A,b,c);
println("Optimalna vrijednost funkcije cilja Z: ", optimalnoZ)
println("Vrijednosti varijabli x: ")
for i=1:size(vrijednostiX,1)
    println("x",i,"=", round(vrijednostiX[i]))
end
```

```
     4×6 Array{Float64,2}:
         12.0  1.0  0.0    0.333333  0.0    0.0
         36.0  0.0  0.0    1.5       1.0   -0.5
          9.0  0.0  1.0   -0.75      0.0    0.25
      -2160.0  0.0  0.0  -20.0       0.0  -10.0
     Optimalna vrijednost funkcije cilja Z: 2160.0
     Vrijednosti varijabli x:
     x1=12.0
     x2=9.0
     x3=0.0
     x4=36.0
     x5=0.0
```

PRIMJER 3.9

```
c1 = [3 1]
A1 = [0.5 0.3; 0.1 0.2]
b1 = [150; 60]
optimalnoZ1,vrijednostiX1=rijesi_simplex(A1,b1,c1);
println("Optimalna vrijednost funkcije cilja Z: ", optimalnoZ1)
println("Vrijednosti varijabli x: ")
for i=1:size(vrijednostiX1,1)
    println("x",i,"=", round(vrijednostiX1[i]))
end
```

```
3×5 Array{Float64,2}:
  300.0  1.0   0.6    2.0  0.0
   30.0  0.0   0.14  -0.2  1.0
 -900.0  0.0  -0.8   -6.0  0.0
Optimalna vrijednost funkcije cilja Z: 900.0
Vrijednosti varijabli x:
x1=300.0
x2=0.0
x3=0.0
x4=30.0
```

PRIMJER 3.10

```
c2 = [800 1000]
A2 = [30 16; 14 19; 11 26; 0 1]
b2 = [22800; 14100; 15950; 550]
optimalnoZ2,vrijednostiX2=rijesi_simplex(A2,b2,c2);
println("Optimalna vrijednost funkcije cilja Z: ", optimalnoZ2)
println("Vrijednosti varijabli x: ")
for i=1:size(vrijednostiX2,1)
    println("x",i,"=", round(vrijednostiX2[i]))
end
```

```
5×7 Array{Float64,2}:
    1550.0  0.0  0.0   0.447977    -1.74566    1.0  0.0
     250.0  0.0  0.0   0.0404624   -0.0867052  0.0  1.0
     600.0  1.0  0.0   0.0549133   -0.0462428  0.0  0.0
     300.0  0.0  1.0  -0.0404624    0.0867052  0.0  0.0
 -780000.0  0.0  0.0  -3.46821    -49.711      0.0  0.0
Optimalna vrijednost funkcije cilja Z: 780000.0
Vrijednosti varijabli x:
x1=600.0
x2=300.0
x3=0.0
x4=0.0
x5=1550.0
x6=250.0
```

PRIMJER 3.11

```
c3 = [1 2]
A3 = [-2 1; 0 1]
b3 = [1; 3]
optimalnoZ3,vrijednostiX3=rijesi_simplex(A3,b3,c3);
println("Optimalna vrijednost funkcije cilja Z: ", optimalnoZ3)
println("Vrijednosti varijabli x: ")
for i=1:size(vrijednostiX3,1)
    println("x",i,"=", round(vrijednostiX3[i]))
end
```

```
3×5 Array{Float64,2}:
  3.0  0.0  1.0   0.0   1.0
  1.0  1.0  0.0  -0.5   0.5
 -7.0  0.0  0.0   0.5  -2.5
Optimalna vrijednost funkcije cilja Z: 7.0
Vrijednosti varijabli x:
x1=1.0
x2=3.0
```

Rješenje 2

```
using DelimitedFiles

function simplex_method(A,b)
    a_row = size(A, 1) + 1     #redovi
    a_col = size(A, 2) + size(A, 1) + 1    #kolone
    a = zeros((a_row, a_col))
    m = size(A,1)
    n = size(A,2)
    for i = 1 : m
      for j = 1 : n
        a[i + 1,j + 1] = A[i, j]
      end
    end
    B = zeros(Int8, 1, m + 1)
    for i = 1 : m
      B[i] = n + i
      a[i + 1,1] = b[i]
      for j = n + 2 : n + m + 1
        a[i + 1,j] = 0
      end
      a[i + 1, n + i + 1] = 1
    end
    for i = 1 : n
      a[1,i + 1] = c[i]
    end
    for j = n + 2 : n + m + 1
      a[1,j] = 0
    end
    a[1,1] = 0
    optimal = false
    println("Unesena matrica je: ")
    writedlm(stdout, a)

    #ITERACIJE DOK SE NE DODJE DO OPTIMUMA
    while(!optimal)
      cmax = -1
      q = 2
      for j = 2 : n + m + 1
        if a[1,j] > 0 &&  a[1,j] > cmax
          cmax = a[1,j]
          q = j
        end
      end
      p = 2
```

```
        tmax = Inf
        if cmax == -1
          optimal = true
        else
          for i = 2 : m + 1
            if a[i, q] > 0
              if a[i, 1] / a[i,q] < tmax
                tmax = a[i,1] / a[i,q]
                p = i
              end
            end
          end
        if tmax == Inf
          println("Rješenje je neograniceno")
          break;
        end
        B[p] = q-1
        # println(q) vodeca kolona
        pivot = a[p,q]
        # println(p) vodeci red
        for j = 1 : m + n + 1      #svodjenje  pivota na 1
          a[p,j] = a[p,j] / pivot
        end
        for i = 1 : m + 1
          if i != p
            factor = a[i,q]
            for j = 1 : n + m + 1
              a[i,j] = a[i,j] - factor * a[p,j]
            end
          end
        end
        end
      end

    x = zeros((1, n + m + 1))
    for j = 2 : n + m + 1
      x[j] = 0
    end

    for i = 2 : m+1
      x[B[i]] = a[i,1]
    end
    Z = a[1,1]
    return x, Z
  end

  # main dio / probni dio zadatak 1 sa zadace 1
  A = [1 1 ; 3 6]
  b = [29 145]
  c = [18 22]
  (x, Z) = simplex_method(A,b)
  println("Optimalno rješenje: ")
  for i = 1 : size(x,2)
    println(x[i])
  end
```

```
println("Funkcija cilja: ")
Z = (-1) * Z
print(Z)
```

```
Unesena matrica je:
0.0     18.0    22.0    0.0     0.0
29.0    1.0     1.0     1.0     0.0
145.0   3.0     6.0     0.0     1.0
Optimalno rješenje:
9.666666666666664
19.333333333333336
0.0
0.0
0.0
Funkcija cilja:
599.3333333333334
```