´o
í
$_ineq =$
$' =':'=',' <=': r' \leq',' >=': r' \geq'$
$_init_{(self):self.A=[]self.b=[]self.c=[]self.t=[]self.tableau=[]self.entering=[]self.departing=[]self.ineq=[]self.prob="min"self.gen_doc=Falseself.doc=""self.f=0s}$
$_simplex(self, A, b, c, f, s, t, prob ='$
$min', ineq =$
$[], enable_msg =$
$False, latex =$
$False) :'''$
$Runsimplexalgorithm.'''self.prob =$
$probself.gen_doc =$
$latexCriarounaoodocself.ineq =$
$ineqself.s =$
$sself.f =$
$fself.t =$
$tAddslackartificialvariablesself.set_simplex_input(A, b, c)$
$_terminate()) :$
$Attempttofindanon-$
$negativepivot.pivot =$
$self.find_pivot()Dorowoperationstomakeeveryotherelementincolumnzero.self.pivot(pivot)solution =$
$self.get_current_solution()Obtemasolucaoself.doc_generate(solution)returnsolution$
$_simplex_input(self, A, b, c) :'''$
$Setinitialvariablesandcreatetableau.'''Convertallentriestofractionsforreadability.forainA :$
$self.A.append([Fraction(x)forxina])self.b =$
$[Fraction(x)forxinb]self.c =$
$[Fraction(x)forxinc]ifnotself.ineq :$
$ifself.prob =='$
$max' :$
$self.ineq =$
$['<='$
$]*$
$len(b)elifself.prob =='$
$min' :$
$self.ineq =$
$['<='$
$]*$
$len(b)Alteração$
$_enter_depart(self.get_Ab())$
$_Ab()m.append(self.c+$
$[0])m =$
$[list(t)fortinzip(*m)]Calculatesthetransposeself.A =$
$[x[:$
$(len(x)-$
$1)]forxinm]self.b =$
$[y[len(y)-$
$1]foryinm]self.c =$
$m[len(m)-$
$1]self.A.pop()self.b.pop()self.c.pop()self.ineq =$
$['<='$
$]*$
$len(self.b)$
$_tableau()self.ineq =$
$['='$
$]*$
$len(self.b)self.update_enter_depart(self.tableau)$
$_enter_depart(self, matrix) :$
$self.entering =$
$[]self.departing =$
$[]Createtablesforenteringanddepartingvariablesforiinrange(0, len(matrix[0])) :$
$ifi <$
$len(self.A[0]) :$
$prefix ='$
$x'ifself.prob =='$
$max'else'y'self.entering.append("elifi <$
$len(matrix[0])-$
$1 :$
$self.entering.append("s_self.departing.append("s_else :$
$self.entering.append("b")$
$_slack_variables(self) :'''$
$Addslackartificialvariablestomatrix Atotransformallinequalitiestoequalities.'''slack_vars =$
$self._generate_identity(len(self.tableau))foriinrange(0, len(slack_vars)) :$
$self.tableau[i]+ =$
$slack_vars[i]self.tableau[i]+ =$
$[self.b[i]]$
$_tableau(self) :'''$
$Createinitialtableautable.'''self.tableau =$
$copy.deepcopy(self.A)self.add_slack_variables()c =$
$copy.deepcopy(self.c)forindex, valueinenumerate(c) :$
$c[index] =$

$most_neg:$
$most_neg=$
$valuemost_neg_ind=$
$indexreturnmost_neg_ind$
${}_departing_var(self,entering_index):'''$
$Tocalculatethedepartingvariable,gettheminimumoftheratioofb(b_i)tothecorrespondingvalueintheenteringcollumn.'''sk$
$0min_ratio_index=$
$-1min_ratio=$
$0forindex,xinenumerate(self.tableau):$
$ifx[entering_index]!=$
$0andx[len(x)-$
$1]/x[entering_index]>$
$0:$
$skip=$
$indexmin_ratio_index=$
$indexmin_ratio=$
$x[len(x)-$
$1]/x[entering_index]break$
${}_ratio>$
$0:$
$forindex,xinenumerate(self.tableau):$
$ifindex>$
$skipandx[entering_index]>$
$0:$
$ratio=$
$x[len(x)-$
$1]/x[entering_index]ifmin_ratio>$
$ratio:$
$min_ratio=$
$ratiomin_ratio_index=$
$index$
${}_ratio_index$
${}_Ab(self):'''$
$GetAmatrixwithbvectorappended.'''matrix=$
$copy.deepcopy(self.A)foriinrange(0,len(matrix)):$
$matrix[i]+=$
$[self.b[i]]returnmatrix$
${}_terminate(self):'''$
$Determineswhetherthereareanynegativeelementsonthebottomrow'''result=$
$Trueindex=$
$len(self.tableau)-$
$1fori,xinenumerate(self.tableau[index]):$
$ifx<$
$0andi!=$
$len(self.tableau[index])-$
$1:$
$result=$
$Falsereturnresult$
${}_current_solution(self):'''$
$Getthecurrentsolutionfromtableau.'''solution=$
$forxinself.entering:$
$ifxisnot'b':$
$ifxinself.departing:$
$solution[x]=$
$self.tableau[self.departing.index(x)][len(self.tableau[self.departing.index(x)])-$
$1]else:$
$solution[x]=$
$0solution['z']=$
$self.tableau[len(self.tableau)-$
$1][len(self.tableau[0])-$
$1]$
${}_1,...,x_nfromlastelementoftheslackcolumns.bottom_row=$
$self.tableau[len(self.tableau)-$
$1]forvinself.entering:$
$if's'inv:$
$solution[v.replace('s','x')]=$
$bottom_row[self.entering.index(v)]$
${}_fraction_to_latex(self,fract):$
$iffract.denominator==$
$1:$
$returnstr(fract.numerator)else:$
$returnr"$
${}_generate_identity(self,n):'''$
$Helperfunctionforgeneratingasquareidentitymatrix.'''I=$
$[]foriinrange(0,n):$
$row=$
$[]forjinrange(0,n):$
$ifi==$
$j:$
$row.append(1)else:$
$row.append(0)I.append(row)returnI$
${}_generate(self,solution):$
$Criatabeladeenvioegeraodocumentolatexifnotself.gen_doc:$
$returnself.doc=$
$(r"$

$'+str(i)])$

çãá

á		

çõ

àáíáã