**GROUP 2 ASSIGNMENT**

**Q. WRITE A CODE TO SOLVE A SIMPLEX ALGORITHM.**

#include <iostream>

#include <vector>

#include <iomanip>

#include <limits>

using namespace std;

void printTableau(const vector<vector<double>> &tableau) {

for (const auto &row : tableau) {

for (double value : row) {

cout << setw(8) << fixed << setprecision(2) << value << " ";

}

cout << "\n";

}

}

int findPivotColumn(const vector<vector<double>> &tableau) {

int pivotCol = -1;

double minValue = 0.0;

for (int j = 0; j < tableau[0].size() - 1; ++j) {

if (tableau.back()[j] < minValue) {

minValue = tableau.back()[j];

pivotCol = j;

}

}

return pivotCol;

}

int findPivotRow(const vector<vector<double>> &tableau, int pivotCol) {

int pivotRow = -1;

double minRatio = numeric\_limits<double>::infinity();

for (int i = 0; i < tableau.size() - 1; ++i) {

double value = tableau[i][pivotCol];

if (value > 0) {

double ratio = tableau[i].back() / value;

if (ratio < minRatio) {

minRatio = ratio;

pivotRow = i;

}

}

}

return pivotRow;

}

void pivot(vector<vector<double>> &tableau, int pivotRow, int pivotCol) {

double pivotValue = tableau[pivotRow][pivotCol];

for (double &value : tableau[pivotRow]) {

value /= pivotValue;

}

for (int i = 0; i < tableau.size(); ++i) {

if (i != pivotRow) {

double factor = tableau[i][pivotCol];

for (int j = 0; j < tableau[0].size(); ++j) {

tableau[i][j] -= factor \* tableau[pivotRow][j];

}

}

}

}

int main() {

vector<vector<double>> tableau = {

{2, 3, 1, 0, 0, 100},

{4, 1, 0, 1, 0, 80},

{3, 2, 0, 0, 1, 60},

{-3, -5, 0, 0, 0, 0}

};

cout << "Initial Tableau:\n";

printTableau(tableau);

while (true) {

int pivotCol = findPivotColumn(tableau);

if (pivotCol == -1) {

cout << "Optimal solution found.\n";

break;

}

int pivotRow = findPivotRow(tableau, pivotCol);

if (pivotRow == -1) {

cout << "Problem is unbounded.\n";

break;

}

pivot(tableau, pivotRow, pivotCol);

cout << "Updated Tableau:\n";

printTableau(tableau);

}

}