Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

**13**

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| **1** | **Write a Python program that can find the solution of following linear equations using LU Decomposition** |
| **2** | **Write a Python program that utilizes LU Decomposition to find the invers as well as determinant of following matrices** |
|  |  |

Submitted On:

**Date: 5/1/2024**

**Task No. 01:**

**Write a Python program that can find the solution of following linear equations using LU Decomposition**

* **2x + 5y = 21, x + 2y = 8**
* **2x + 3y – z = 5, 3x + 2y + z = 10, x - 5y + 3z = 0**

**Solution:**

import numpy as np

def doolittle\_lu\_decomposition(A):

n = len(A)

L = np.eye(n)

U = np.zeros((n, n))

for i in range(n):

for j in range(i, n):

U[i][j] = A[i][j] - sum(L[i][k] \* U[k][j] for k in range(i))

for j in range(i + 1, n):

L[j][i] = (A[j][i] - sum(L[j][k] \* U[k][i] for k in range(i))) / U[i][i]

return L, U

A = np.array([[2, 5],

[1, 2]], dtype=float)

a=np.array([21, 8])

B = np.array([[2, 3,-1],

[3, 2,1],

[1, -5,3]], dtype=float)

b=np.array([5, 10, 0])

LA, UA = doolittle\_lu\_decomposition(A)

LB, UB = doolittle\_lu\_decomposition(B)

ya = np.linalg.solve(LA, a)

res\_a = np.linalg.solve(UA, ya)

yb = np.linalg.solve(LB, b)

res\_b = np.linalg.solve(UB, yb)

print("Result 1:", res\_a)

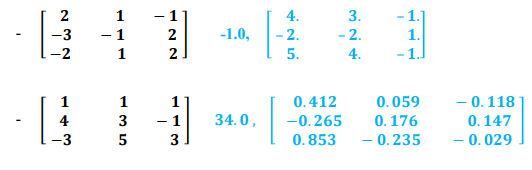
print("Result 2:", res\_b)

**Output:**



**Task No. 02:**

**Write a Python program that utilizes LU Decomposition to find the invers as well as determinant of following matrices:**



**Solution:**

import numpy as np

def doolittle\_lu\_decomposition(A):

n = len(A)

L = np.eye(n)

U = np.zeros((n, n))

for i in range(n):

for j in range(i, n):

U[i][j] = A[i][j] - sum(L[i][k] \* U[k][j] for k in range(i))

for j in range(i + 1, n):

L[j][i] = (A[j][i] - sum(L[j][k] \* U[k][i] for k in range(i))) / U[i][i]

return L, U

def determinant\_from\_U(U):

det = np.prod(np.diagonal(U))

return det

def matrix\_inverse(L, U):

n = len(L)

I = np.eye(n)

Y = np.zeros((n, n))

for i in range(n):

Y[i][i] = 1

for j in range(i):

Y[i] -= L[i][j] \* Y[j]

X = np.zeros((n, n))

for i in range(n - 1, -1, -1):

X[i] = Y[i]

for j in range(i + 1, n):

X[i] -= U[i][j] \* X[j]

X[i] /= U[i][i]

return X

A = np.array([[2, 1,-1],

[-3, -1,2],

[-2, 1,2]], dtype=float)

B = np.array([[1, 1,1],

[4, 3,-1],

[-3, 5,3]], dtype=float)

LA, UA = doolittle\_lu\_decomposition(A)

LB, UB = doolittle\_lu\_decomposition(B)

detA=determinant\_from\_U(UA)

print('Determinent A: ' ,detA)

detB=determinant\_from\_U(UB)

print('Determinent B: ' ,detB)

inverseA=matrix\_inverse(LA, UA)

print('Invere A: ')

print(inverseA)

inverseB=matrix\_inverse(LB, UB)

print('Invere B: ')

print(inverseB)

**Output:**

