Experiment 3a:

Aim: To solve a system of linear equations using Gaussian elimination method.

Software packages used: language- Python3, packages- Numpy

Theory: Gaussian elimination method or row reduction method, is an algorithm for solving systems of linear equations. If we have a system of n number of equations for n unknowns, x_n

$$\begin{aligned}
A_{11}x_1 + A_{12}x_2 + A_{13}x_3 + \ldots + A_{1n}x_n &= B_1 \\
A_{21}x_1 + A_{22}x_2 + A_{23}x_3 + \ldots + A_{2n}x_n &= B_2 \\
A_{31}x_1 + A_{32}x_2 + A_{33}x_3 + \ldots + A_{3n}x_n &= B_3 \\
&\ldots \\
A_{n1}x_1 + A_{n2}x_2 + A_{n3}x_3 + \ldots + A_{nn}x_n &= B_n
\end{aligned}$$

then, by reducing the augmented matrix

$$\begin{bmatrix}
A_{11} & A_{12} & A_{13} & \ldots & A_{1n} & B_1 \\
A_{21} & A_{22} & A_{23} & \ldots & A_{2n} & B_2 \\
A_{31} & A_{32} & A_{33} & \ldots & A_{3n} & B_3 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
A_{n1} & A_{n2} & A_{n3} & \ldots & A_{nn} & B_n
\end{bmatrix}$$

to a row-echelon form using elementary row operations, and then using back substitution, we can find out the unknowns x_n.

Code:

In [ ]:
```
# importing required packages
import numpy as np
import sys
```

In [ ]:
```
# reading number of unknowns
n = int(input('Enter number of unknowns, n: '))
```

In [ ]:
```
# Making numpy array of n x n+1 size and initializing to zero for storing augmented matrix
A = np.zeros((n, n+1))

# Reading augmented matrix coefficients
print('Enter Augmented Matrix Coefficients:')
for i in range(n):
    for j in range(n):
        A[i][j] = float(input( 'A'+str(i+1)+ str(j+1)+'='))
    A[i][n] = float(input( 'B'+str(i+1)+'='))
```

In [ ]:
```
#### Applying Gauss Elimination
for i in range(n):
    # Checking for zero pivot
    if A[i][i] == 0.0:
        sys.exit('Divide by zero detected!')

    # Eliminating subdiagonal elements
    for j in range(i+1, n):
        ratio = A[j][i]/A[i][i]

        # Subtracting row i from row j
        for k in range(n+1):
            A[j][k] = A[j][k] - ratio * A[i][k]
```

In [ ]:
```
# Back Substitution

for i in range(n-1,-1,-1):
    for j in range(i+1,n):
        ratio = A[i][j]/A[j][j]
        for k in range(n+1):
            A[i][k] = (A[i][k] - ratio * A[j][k])

for i in range(n):
    for j in range(n,-1,-1):
        A[i][j]= A[i][j]/A[i][i]
```

In [ ]:
```
# Displaying Solutions

for i in range(n):
    print("x%d = %0.2f" %(i,A[i][n]) )
```

Observation and Result:

Input =

Output =