Question 1

In [ ]:

```
from functionLibrary import fwdSub
from functionLibrary import bwdSub
from functionLibrary import dlittle
from functionLibrary import crout
from functionLibrary import solver

matrixA = open("q1-A.txt", "r+")
A = []
for row in matrixA:
    e1 = row.split()
    fe1 = []
    for i in range(len(e1)):
        fe1.append(float(e1[i]))
    A.append(fe1)


matrixb = open("q1-b.txt", "r+")
b = []
for row in matrixb:
    e1 = row.split()
    for i in range(len(e1)):
        b.append(float(e1[i]))


print("The solution using Doolittle's algorithm:" + "\n")
print( "x = " + str(solver(A,b, dlittle)) + "\n" )


print("\n" + "The solution using Crout's algorithm:" + "\n")
print( "x = " + str(solver(A,b, crout)) + "\n")
```

```
The solution using Doolittle's algorithm:

L = [[1, 0, 0, 0], [0.0, 1, 0, 0], [1.0, 2.0, 1, 0], [2.0, 1.0, 1.5, 1]]

U = [[1.0, 0.0, 1.0, 2.0], [0, 1.0, -2.0, 0.0], [0, 0, 2.0, -2.0], [0, 0, 0, -3.0]]

x = [1.0, -1.0, 1.0, 2.0]


The solution using Crout's algorithm:

L = [[1.0, 0, 0, 0], [0.0, 1.0, 0, 0], [1.0, 2.0, 2.0, 0], [2.0, 1.0, 3.0, -3.0]]

U = [[1, 0.0, 1.0, 2.0], [0, 1, -2.0, 0.0], [0, 0, 1, -1.0], [0, 0, 0, 1]]

x = [1.0, -1.0, 1.0, 2.0]
```

Question 2

In [ ]:

```
from functionLibrary import inverseLU
from functionLibrary import swapRows
from functionLibrary import LUDecomp
from functionLibrary import fwdBwdSub
from functionLibrary import PartialPivot
from functionLibrary import multiplysquare

matrixA = open("q2.txt", "r+")
A = []
for row in matrixA:
    e1 = row.split()
    fe1 = []
    for i in range(len(e1)):
        fe1.append(float(e1[i]))
    A.append(fe1)

I = [[0 for i in range(len(A))] for j in range(len(A))]
for k in range(len(A)):
    I[k][k] = 1.0


X = inverseLU(A,I)

verification = multiplysquare(X,A)
```

```
Lower Triangular
[1, 0, 0, 0]
[0.0, 1, 0, 0]
[0.0, 0.5, 1, 0]
[0.0, 0.0, -0.25, 1]
Upper Triangular
[3.0, 7.0, 1.0, 0.0]
[0, 2.0, 8.0, 6.0]
[0, 0, -4.0, -2.0]
[0, 0, 0, 1.5]
The inverse of the given matrix is
[0.3333333333333333, -0.25000000000000006, -1.8333333333333333, 1.6666666666666667]
[0.0, 0.08333333333333337, 0.8333333333333333, -0.6666666666666667]
[-0.0, 0.16666666666666666, -0.3333333333333333, -0.3333333333333333]
[0.0, -0.08333333333333333, 0.16666666666666666, 0.6666666666666666]
The product of
[0.3333333333333333, -0.25000000000000006, -1.8333333333333333, 1.6666666666666667]
[0.0, 0.08333333333333337, 0.8333333333333333, -0.6666666666666667]
[-0.0, 0.16666666666666666, -0.3333333333333333, -0.3333333333333333]
[0.0, -0.08333333333333333, 0.16666666666666666, 0.6666666666666666]
and
[3.0, 7.0, 1.0, 0.0]
[0.0, 2.0, 8.0, 6.0]
[0.0, 1.0, 0.0, 1.0]
[0.0, 0.0, 1.0, 2.0]
is
[1.0, 0, 0, 0]
[0, 1.0, 0, 0]
[0, 0, 1.0, 0]
[0, 0, 0, 1.0]
```

Question 3

In [ ]:

```
from functionLibrary import cholesky
from functionLibrary import choleskySolver
from functionLibrary import transpose
```

```python
matrixA = open("q3-A.txt", "r+")
A = []
for row in matrixA:
    e1 = row.split()
    fe1 = []
    for i in range(len(e1)):
        fe1.append(float(e1[i]))
    A.append(fe1)


matrixb = open("q3-b.txt", "r+")
b = []
for row in matrixb:
    e1 = row.split()
    for i in range(len(e1)):
        b.append(float(e1[i]))

l = cholesky(A)

x = choleskySolver(l, transpose(l), b)
print("Using Cholesky algorithm, the solution matrix is: ")
print(x)
```

```
Using Cholesky algorithm, the solution matrix is:
[0.09999999999999996, 0.2, 0.3, 0.4000000000000001]
```