1. My Python Code:

```python
# Hsuan-You Lin Module 10 Problem Set Question 1.
import numpy as np

def lower_triangular_system(Lower_Matrix, b_Matrix):
    for i in range(len(Lower_Matrix)):
        b_Matrix[i] = b_Matrix[i] / Lower_Matrix[i][i]
        Lower_Matrix[i][i] = 1
        for j in range(i + 1, len(Lower_Matrix)):
            b_Matrix[j] -= Lower_Matrix[j][i] * b_Matrix[i]
            Lower_Matrix[j][i] = 0
            x_Matrix = b_Matrix
    return x_Matrix

if __name__ == "__main__" :
    Lower_Matrix = np.array([[1, 0, 0],
                             [4, 1, 0],
                             [-6, 5, 1]])
    b_Matrix = np.array([3, 14, -7])
    print("Q1: Solve the lower triangular system: Lx = b.")
    print("L = \n", Lower_Matrix)
    print("b = ", b_Matrix)
    ans = lower_triangular_system(Lower_Matrix, b_Matrix)
    print("x = ", ans)
```

```
Module12 — -bash — 80×24
(base) pisces:Module12 pisces$ python Q1.py
Q1: Solve the lower triangular system: Lx = b.
L =
 [[ 1  0  0]
 [ 4  1  0]
 [-6  5  1]]
b =  [ 3 14 -7]
x =  [3 2 1]
(base) pisces:Module12 pisces$
```

2. My Python Code:

```python
# Hsuan-You Lin Module 10 Problem Set Question 2.
import numpy as np

def upper_triangular_system(Upper_Matrix, b_Matrix):
    for i in range(len(Upper_Matrix)-1, -1, -1):
        b_Matrix[i] = b_Matrix[i] / Upper_Matrix[i][i]
        Upper_Matrix[i][i] = 1
        for j in range(i):
            b_Matrix[j] -= Upper_Matrix[j][i] * b_Matrix[i]
            Upper_Matrix[j][i] = 0
            x_Matrix = b_Matrix
    return x_Matrix

if __name__ == "__main__" :
    Upper_Matrix = np.array([[1, 2, 1],
                             [0, 1, 4],
                             [0, 0, 3]])
    b_Matrix = np.array([5, 5, 3])
    print("Q2: Solve the upper triangular system: Ux = b.")
    print("U = \n", Upper_Matrix)
    print("b = ", b_Matrix)
    ans = upper_triangular_system(Upper_Matrix, b_Matrix)
    print("x = ", ans)
```

```
Module12 — -bash — 80×24
(base) pisces:Module12 pisces$ python Q2.py
Q2: Solve the upper triangular system: Ux = b.
U =
 [[1 2 1]
 [0 1 4]
 [0 0 3]]
b =  [5 5 3]
x =  [2 1 1]
(base) pisces:Module12 pisces$
```

3. My Step:

```
L =  [1, 0, 0]    x =  [x1]  b =  [3]              Answer x = [3]
     [4, 1, 0]         [x2]      [14]      ->               [2]
     [-6, 5, 1]        [x3]      [-7]                       [1]
```

Each iteration for this lower triangular matrix will be:

```
Iteration 1:                Iteration 2:                Iteration 3:
L:          b:              L:          b:              L:          b:
 [1, 0, 0]   [3]             [1, 0, 0]   [3]             [1, 0, 0]   [3]
 [0, 1, 0]   [2]      ->     [0, 1, 0]   [2]      ->     [0, 1, 0]   [2]
 [0, 5, 1]   [11]            [0, 0, 1]   [1]             [0, 0, 1]   [1]
```

---

4. My Python code:

```python
# Hsuan-You Lin Module 10 Problem Set Question 4.
import numpy as np

def LU_decomposition(A_Matrix, n):
    Lower_Matrix = [[0 for x in range(n)]
                    for y in range(n)]
    Upper_Matrix = [[0 for x in range(n)]
                    for y in range(n)]

    for i in range(n):
        for k in range(i, n):
            LU = 0
            for j in range(i):
                LU += (Lower_Matrix[i][j] * Upper_Matrix[j][k])
            Upper_Matrix[i][k] = A_Matrix[i][k] - LU
        for k in range(i, n):
            if (i == k):
                Lower_Matrix[i][i] = 1
            else:
                LU = 0
                for j in range(i):
                    LU += (Lower_Matrix[k][j] * Upper_Matrix[j][i])

                Lower_Matrix[k][i] = int((A_Matrix[k][i] - LU) /
                                         Upper_Matrix[i][i])
    print("Lower Triangular Matrix \tUpper Triangular Matrix")
    for i in range(n):
        for j in range(n):
            print(Lower_Matrix[i][j], end="\t")
        print("", end="\t")
        for j in range(n):
            print(Upper_Matrix[i][j], end="\t")
        print("")
    return Lower_Matrix, Upper_Matrix

if __name__ == "__main__" :
    A_Matrix = np.array([[4, -5, 6],
                         [8, -6, 7],
                         [12, -7, 12]])
    LU_decomposition(A_Matrix, 3)
```

```
Module12 — -bash — 80×24
(base) pisces:Module12 pisces$ python Q4.py
Lower Triangular Matrix         Upper Triangular Matrix
1       0       0               4       -5      6
2       1       0               0       4       -5
3       2       1               0       0       4
(base) pisces:Module12 pisces$
```

Each row operation in the LU decomposition will be:

```
Iteration 1:          Iteration 2:          Iteration 3:
 [4, -5, 6]            [4, -5, 6]            [4, -5, 6]
 [2, 4, -5]     ->     [2, 4, -5]     ->     [2, 4, -5]
 [3, 8, -6]            [3, 2, 4]             [3, 2, 4]
```

5. My Python code:

```python
# Hsuan-You Lin Module 10 Problem Set Question 5.
def karatsuba(x, y):
    a, b = x // 100, x % 100
    c, d = y // 100, y % 100

    ac = a * c
    print("\n<In Karatsuba algorithm> \na * c = ", ac)
    bd = b * d
    print("b * d = ", bd)
    abcd = (a+b) * (c+d)
    print("(a + b) * (c + d) = ", abcd)
    result = ac * 10000 + (abcd - ac - bd) * 100 + bd

    return result

if __name__ == "__main__" :
    x = 5822
    y = 4104
    print("Input Number: \n X = ", x, "; Y = ", y)
    print("\nExpected X * Y = ", x*y)
    ans = karatsuba(x, y)
    print("\n<After Karatsuba algorithm> \nX * Y = ", ans)
```

```
Module12 — -bash — 80×24
(base) pisces:Module12 pisces$ python Q5.py
Input Number:
 X =  5822 ; Y =  4104

Expected X * Y =  23893488

<In Karatsuba algorithm>
a * c =  2378
b * d =  88
(a + b) * (c + d) =  3600

<After Karatsuba algorithm>
X * Y =  23893488
(base) pisces:Module12 pisces$
```

From above python code, we can see that (a * c) = 2378, (b * d) = 88, (a + b) * (c + d) = 3600, and we can calculate [(a + b) * (c + d) - (a * c) - (b * d)] = 1134, Finally we could get 5822 * 4104 = 23893488 using Karatsuba algorithm.