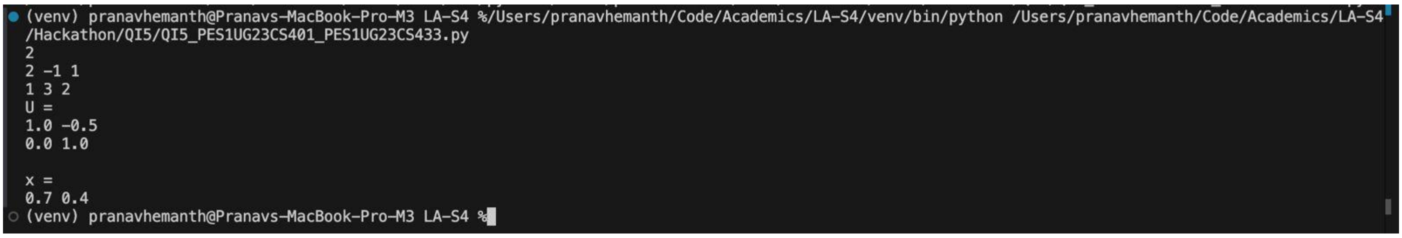
# Linear Algebra Hackathon 1
# PES University

Name: Nishant Holla and Pranav Hemanth
SRN: PES1UG23CS401 and PES1UG23CS433

### Implementation Question 5

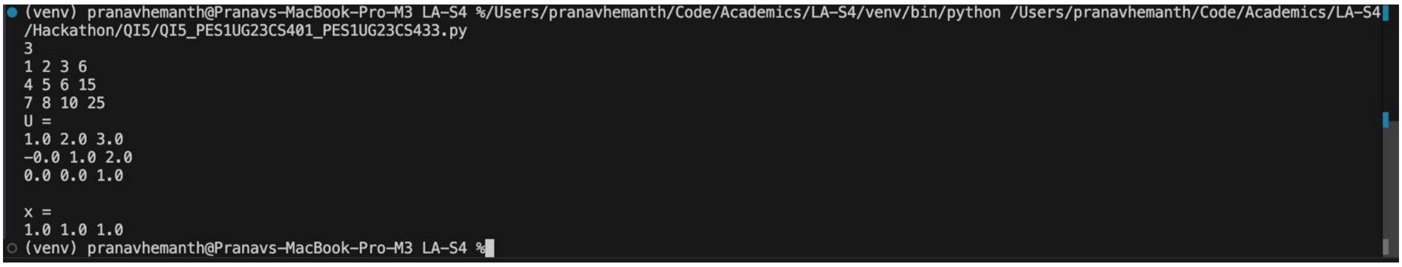
Test Case 1:



```
(venv) pranavhemanth@Pranavs-MacBook-Pro-M3 LA-S4 %/Users/pranavhemanth/Code/Academics/LA-S4/venv/bin/python /Users/pranavhemanth/Code/Academics/LA-S4
/Hackathon/QI5/QI5_PES1UG23CS401_PES1UG23CS433.py
2
2 -1 1
1 3 2
U =
1.0 -0.5
0.0 1.0

x =
0.7 0.4
(venv) pranavhemanth@Pranavs-MacBook-Pro-M3 LA-S4 %
```

Test Case 2:



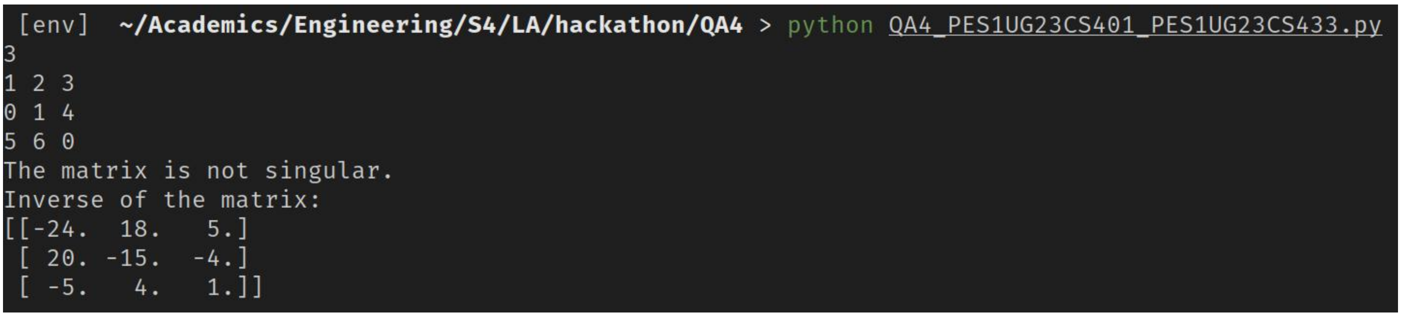
```
(venv) pranavhemanth@Pranavs-MacBook-Pro-M3 LA-S4 %/Users/pranavhemanth/Code/Academics/LA-S4/venv/bin/python /Users/pranavhemanth/Code/Academics/LA-S4
/Hackathon/QI5/QI5_PES1UG23CS401_PES1UG23CS433.py
3
1 2 3 6
4 5 6 15
7 8 10 25
U =
1.0 2.0 3.0
-0.0 1.0 2.0
0.0 0.0 1.0

x =
1.0 1.0 1.0
(venv) pranavhemanth@Pranavs-MacBook-Pro-M3 LA-S4 %
```

- First we are given a square matrix of size NxN and the column of the B matrix. So we can use the **np.hstack** function in numpy to reshape the NxN matrix to NxN+1 augmented matrix.
- We then **normalize** the row and then ensure all values below pivot are 0. Doing this for all rows gives us the Upper triangular matrix **U**
- We initialize the Vector **X** with 0s using **np.zeros** for solution vector
- Once we have **U** we do backward substitution to get the values of vector **X**

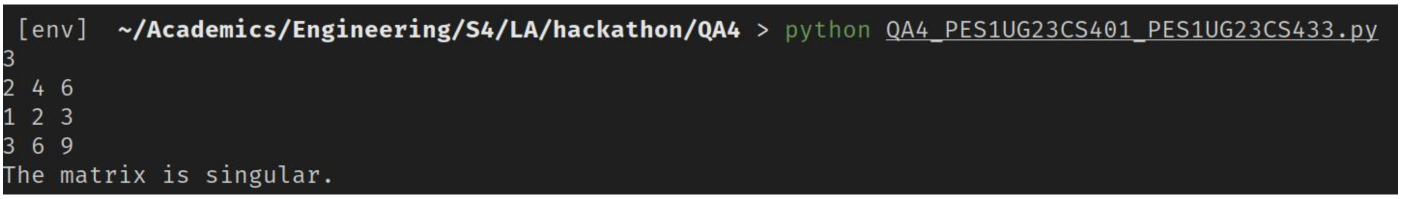
### Application Question 4

Test Case 1:



```
 [env]  ~/Academics/Engineering/S4/LA/hackathon/QA4 > python QA4_PES1UG23CS401_PES1UG23CS433.py
3
1 2 3
0 1 4
5 6 0
The matrix is not singular.
Inverse of the matrix:
[[-24.  18.   5.]
 [ 20. -15.  -4.]
 [ -5.   4.   1.]]
```

Test Case 2:



```
 [env]  ~/Academics/Engineering/S4/LA/hackathon/QA4 > python QA4_PES1UG23CS401_PES1UG23CS433.py
3
2 4 6
1 2 3
3 6 9
The matrix is singular.
```

- The input matrix is determined to be a square matrix if its **shape[0]** is equal to its **shape[1]** else the matrix is considered to be rectangular.
- If the matrix is a square matrix then **np.linalg.inv** function is used to calculate the inverse of the square matrix.
- If the matrix is a rectangular matrix then Moore-Penrose pseudo-inverse is calculated using **np.linalg.pinv** function.
- If the functions raise a **np.linalg.LinAlgError** then the input matrix is **singular** else it is **not singular** and inverse exists.