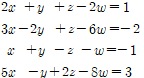
**학번:** (학번을 쓰시오)  **성명:** (이름을 쓰시오)

1. 다음 선형연립방정식을 Gauss-Jordan 소거법으로 풀어라.



**□ 풀이**

[Sage]

# 1) 예시  
# ------------------------  
A = matrix([  
  [2, 1,  1, -2],  
  [3, -2, 1, -6],  
  [1, 1, -1, -1],  
  [5, -1, 2, -8],  
])  
​  
# 상수항 벡터  
b = vector([1, -2, -1, 3])  
​  
# 첨가행렬  
aug\_mat = A.augment(b)  
​  
# Gauss-Jordan => RREF 로 변형하여 풀기  
mat\_RREF = aug\_mat.rref()  
​  
print(mat\_RREF)  
'''  
[     1     0     0 -17/11     0]  
[     0     1     0   9/11     0]  
[     0     0     1   3/11     0]  
[     0     0     0     0     1]  
>> 최하단에 선행성분이 없는데(0행렬), 선행성분이 1이므로 해가 존재하지 않는다. (False)  
'''  
​  
# 2) random\_case test  
# 행렬 생성  
A = random\_matrix(ZZ, 7, 7, x = -20, y = 20)   
​  
print(A)  
​  
# 상수항 벡터  
b = random\_vector(ZZ, 7, x = -20, y = 20)   
​  
# 첨가행렬  
aug\_mat = A.augment(b)  
​  
# Gauss-Jordan => RREF 로 변형하여 풀기  
mat\_RREF = aug\_mat.rref()  
print(mat\_RREF)  
​  
# 해  
sol = A.solve\_right(b)  
print(sol)

**□ 설명**

1) 예시의 4 x 4 행렬을 이용한 계산 => 출력 결과 마지막 선행성분이 없으므로 해가 없다.

2)

- 먼저 랜덤매트릭스, 랜덤벡터 함수를 사용하여 7 by 7 사이즈를 만든다.

- 마찬가지로 RREF -> solve\_right를 이용하여 값을 계산해준다.

- (선행항을 1로 만드는 과정을 거쳐 값을 구함)

- ex) R3 \* -1을 통하여 1로 만들고, R3을 제외한 나머지 줄의 앞을 다 소거

…

이러한 계산을 통해 최종적으로 A.solve\_right(b) 와 같은 해를 얻는다.

# matrix\_A

[ -4  -7   4  17 -19  11  -8]

[ 11   9   9 -19  19 -11 -10]

[ -1  19   2  -9   5   9  -8]

[  9 -17  -6  -7 -13 -15  -1]

[  8  -5  17 -13  15 -17   6]

[ -7 -18   3  13  19  15  -5]

[ 11  11  -1  11 -15  18   9]

​

# RREF -> Sol

# Gauss-Jordan => RREF 로 변형하여 풀기

mat\_RREF = aug\_mat.rref()

print(mat\_RREF)

​

# 해

sol = A.solve\_right(b)

print(sol)

​

# 아래와 같은 답이 나온다.

(-21504263/12890831, -11643789/12890831, 79737931/38672493, -89120698/38672493, -19240194/12890831, 16001972/12890831, 34224292/38672493)