```
using LinearAlgebra, RowEchelon
```

Question 2

Describe all least-squares solutions of the equation Ax = b

The least-squares solutions are all vectors of the form xhat = ___ + x3 ___

```
A = 4×3 Matrix{Int64}:
 1  1  0
 1  1  0
 1  0  1
 1  0  1
```

```
A = [
    1 1 0;
    1 1 0;
    1 0 1;
    1 0 1
]
```

```
b =  [1, 3, 7, 5]
```

```
b = [
    1;
    3;
    7;
    5
]
```

```
AT = 3×4 transpose(::Matrix{Int64}) with eltype Int64:
 1  1  1  1
 1  1  0  0
 0  0  1  1
```

```
AT = transpose(A)
```

```
Aug = 3×4 Matrix{Int64}:
  4  2  2  16
  2  2  0   4
  2  0  2  12
```

```
Aug = [AT*A AT*b]
```

```
3×4 Matrix{Float64}:
 1.0  0.0   1.0   6.0
 0.0  1.0  -1.0  -4.0
 0.0  0.0   0.0   0.0
```

```
rref(Aug)
```

x1 + x3 = 6; x2 - x3 = -4

The least-squares solutions are all vectors of the form

a = [6, -4, 0]

```
a = [
    6;
    -4;
    0;
]
```

f = [-1, 1, 1]

```
f = [
    -1;
    1;
    1
]
```

Question 3

Compute the least-squares error associated with the solution

A_3 = 3×2 Matrix{Int64}:
 1 2
 1 -1
 1 1

```
A_3 = [
    1 2;
    1 -1;
    1 1
]
```

b_3 = [4, 6, 0]

```
b_3 = [
    4;
    6;
    0
]
```

xhat_3 = [4, -1]

```
xhat_3 = [
    4;
    -1
]
```

pred_b_3 = [2, 5, 3]

```
pred_b_3 = A_3 * xhat_3
```

```
error_3 = 3.7416573867739413
```

```
error_3 = sqrt(sum((b_3 - pred_b_3).^2))
```

Question 4

Use the factorization A = QR to find the least-squares solution of Ax = b

```
A_4 = 3×2 Matrix{Int64}:
 2  3
 2  4
 1  1
```

```
A_4 = [
    2 3;
    2 4;
    1 1
]
```

```
Q_4 = 3×2 Matrix{Float64}:
 0.666667  -0.333333
 0.666667   0.666667
 0.333333  -0.666667
```

```
Q_4 = [
    2/3 -1/3;
    2/3 2/3;
    1/3 -2/3
]
```

```
R_4 = 2×2 Matrix{Int64}:
 3  5
 0  1
```

```
R_4 = [
    3 5;
    0 1
]
```

```
b_4 =  [8, 7, 9]
```

```
b_4 = [
    8;
    7;
    9
]
```

```
A_4T = 2×3 transpose(::Matrix{Int64}) with eltype Int64:
 2  2  1
 3  4  1
```

```
A_4T = transpose(A_4)
```

```
2×3 Matrix{Float64}:
 2.0  2.0  1.0
 3.0  4.0  1.0
```

```
transpose(R_4) * transpose(Q_4)
```

```
2×2 Matrix{Int64}:
  9  15
 15  26
```

```
1 A_4T * A_4
```

```
[39, 61]
```

```
1 A_4T * b_4
```

```
xhat_4 =  [11.0, -4.0]
```

```
1 xhat_4 = inv(A_4T * A_4) * A_4T * b_4
```

We get xhat here, but I'm not sure how the idea that A = QR -> A^T = R^T Q^T would be helpful