```
function[L,U,P] = luFactor1(A)
%luFactor
[m,n]=size(A);
If m~=n
%Checking to see if A is the right size.
disp("Error, A needs to be a square matrix.");
    P=eye(size(A));
    %Making the pivot matrix which starts out as an identity matrix.
    L=P;
    %Making it so L starts as the same identity matrix as P.
    U=A;
    %Starting out by making the U matrix as the A matrix
    for x=1:n;
        x=1;
        %x is the counter that will show what row and column we are in.
        max(abs(U(x:n,x)))=themax;
        rownumber=x;
        for rownumber=x:n;
            if U(rownumber,x) == themax
                U=linalg::swapRow(U, x, rownumber);
                %This finds what row has the highest absolute value and
                %swaps it with the top row from the U matrix.
                P=linalg::swapRow(P,x,rownumber);
                %After swapping the rows in the U matrix we need to swap
                %the same rows for the pivot matrix and we do that here.
                for x=2:n
                p=L(x,1:x-1);
                %Here I am getting the values of the top of of the one that
                %needs to be switched in the L matrix and I am setting it
                %equal to something so I can substitute it into the new L
                %matrix.
                q=L(rownumber,1:x-1);
                L(x,1:x-1)=q;
                %Here I am substituting the values into the L matrix
                %because I could not find a swap function that would only
                %swap a couple parts of the matrix.
                L(rownumber,1:x-1)=p;
            else
                rownumber=rownumber + 1;
            end
        end
        y=x+1;
        for y=x:n;
            c=(U(y,x)/U(x,x));
            %These values are the ones that are going to be stored into the
            %L matrix.
            L(y,x)=c
            d=c*U(x,x);
            U(y,:)=U(y,:)-d;
            y=y+1;
        end
        x=x+1;
        end
    end
```