Orion Gregory
This is all my work, I don't really know what it counts for but here's my digital signature, OG.
Math 237-007
12160343
Project 1
1.

a. a = 15; 15
2.2 * a -3; 30
B = exp(4); 54.5982
Pi; 2.1416
C = b / a; 3.6399

the input and output for the following commands are shown here, we are setting a = 15, then multiplying 2.2 by a, then taking that product and subtracting 3. We are then setting b to e^4, as denoted by the e ex(4) notation, we define pi, then set variable c equal to the dividend of b and a.
a = b + 16*pi; 104.8636
We then set a equal to the variable b, defined by (e xp(4)) and add it to 16 times pi. Which results in a being equal to an irrational number, which exact is e^4 + 16pi, but is approximate to 104.8636

b.
1. a = [2 2 3 4 5; 6 1 2 0 7; -4 2 -1 3 4]; the result of inputting c = [a b] is that it conjoins matrix b onto the end of matrix a, this could be useful for combining a coefficient matrix to a system of equations solutions, or right-hand side.
2. D = c(3,5) This extracts the specific value that is in row 3, column 5 of matrix C

3. e = c(2, :) This returns the entire 2nd row of matrix C

4. f = c(:,2) This returns the entire 2nd column of matrix C

5. g=c(:,[1 3 5]) This returns the entire 1st, 3rd, and 5th column of matrix C

6.c([1 3],:) = c([3 1],:) This is the start of something, this is a elementary row operation, we have switched the 1st and 3rd rows in matrix C

7. c(3,:) = 5 * c(3,:); This multiplies the third row by a scalar of 5, this is how scalar multiplication is done in matlab.

8. c(3, :) = c(3, :) + (−c(3, 1)/c(1, 1)) ∗ c(1, :); This one is another row operation, although with more calculations. This adds the third row of matrix C to a scalar multiplied by row 1. The scalar is found by finding the element in the 3rd row and 1st column, and finding the negative of that, then dividing it by the element in the 1st row and 1st column in matrix C.

9. zeros(5, 2) This creates a 5x2 zero matrix

10. eye(6) This creates an identity matrix, 6x6

c.

1. a = amatrix(2,1) this simply displays the element that is in row 2, column 1, The first argument given in the parentheses is the row, and the second given is the column.

2. to display a, all that must be done is to simply write ‘a’, without quotation marks

3. temp1 = a, this creates another variable called temp1, and the equal sign copies the data from the variable a, to temp1

4. amatrix(3,:), this displays the third row of matrix a, the first argument is the row that wants to be displayed, and the colon acts as a marker saying, “hey, im talking about every single column in this array”.

5. amatrix([1 3],:) = amatrix([3 1],:), the arguments in this are a bit more tricky, the first element of the argument is a set of numbers [1 3] or [3 1]. These numbers signify which rows are being operated on, and in what order should they be switched.

6. temp2 = c does the same exact thing as 3.

7. c(:,3) same idea as 4. Except the colon is placed in the row spot, as to signify each element in the row shall be displayed if it is in the third column

8. c(:,4) = c(:,4) * 2 this is simply scalar multiplication, we are stating that column 4 is equal to itself, except each element in the column is multiplied by 2

9. c(:, [2 5]) = c(:, [5 2]) same idea as 5. Except we are doing it with columns instead

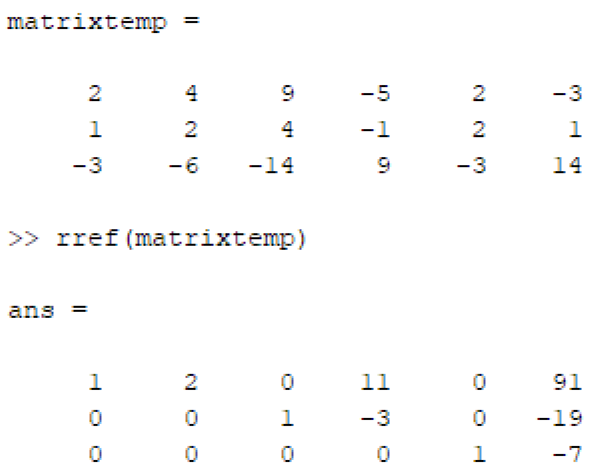
10. amatrix(2, :) = amatrix(2, :) + (-amatrix(2, 1)/amatrix(1, 1)) * amatrix(1, :) this basically does R2 = -3R1 + R2. But this is a generalized formula, for any element in 2,1 this should eliminate 1,1.

11. h = c(:,[1 2 5 6]) This just copies the information from specific rows in C to a new matrix, h.

2.

X5 = -7
X4 = S1
X3 = -19+3(S1)
X2=S2
X1= S2 + S1 -91

The system of equations is turned into an augmented matrix, using the coefficients each variable and the right-hand side at the end. X2 and X4 are free variables, as they do not contain pivots in their columns, X4 and X2 will be denoted as S, and the system Is solved using them in place of X4 and X2. The system has infinitely many solutions because Of the presence of free variables.



```
matrixtemp =

     2     4     9    -5     2    -3
     1     2     4    -1     2     1
    -3    -6   -14     9    -3    14

>> rref(matrixtemp)

ans =

     1     2     0    11     0    91
     0     0     1    -3     0   -19
     0     0     0     0     1    -7
```