Question 1

Below is test data from Code 1.1 referenced on page 6 using $p = 11$.

a	1	2	3	4	5	6	7	8	9	10
a^{-1}	1	6	4	3	9	2	8	7	5	10

Table 1: Test data for Code 1.1

To improve this procedure, we exploit the fact that each $2 \le a \le p-2$ has a unique distinct inverse. Therefore by storing `inverses(a)=b` and `inverses(b)=a` in one iteration, and assuming that each of the inverses are randomly distributed in $\mathbb{Z}_p$, there would be $(p+1)/2$ iterations of the code. For large p this would speed up the procedure by a factor of 2, however I appreciate the code would check if each entry had already been calculated, so this factor would be slightly less than 2. The modification is referenced as Code 1.2 on page 7. I ran both codes with $p = 7121$; the first took 0.92 seconds, the second took 0.58 seconds, confirming this idea.

Question 2

Suppose a_i was the inverse for i, so $i \cdot a_i \equiv 1 \mod p$. There are 3 operations to check if mod(a_i*b)==1 for some b, and this will iterate i times when finding the inverse for a_i. So there will be approximately

$$\sum_{i=1}^{p-1} (3i) = \frac{3}{2} p(p-1) \tag{1}$$

operations. Creating a zero vector at the start uses $p-1$ operations, so the complexity of the program is $O(p^2)$.

Question 3

The results in the second column of Table 2 were produced by Code 3.1, referenced on page 8.[1]

Matrix	Echelon Form	Rank	Basis for Row Space
$A_1 \text{ mod}(3)$	$\begin{pmatrix} 1 & 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 2 & 1 \\ 0 & 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$	3	$\left\{ \begin{matrix} \begin{pmatrix} 1 & 0 & 1 & 1 & 2 \end{pmatrix}, \\ \begin{pmatrix} 0 & 1 & 1 & 2 & 1 \end{pmatrix}, \\ \begin{pmatrix} 0 & 0 & 1 & 2 & 0 \end{pmatrix}. \end{matrix} \right\}$
$A_1 \text{ mod}(29)$	$\begin{pmatrix} 1 & 0 & 22 & 26 & 11 \\ 0 & 1 & 7 & 2 & 10 \\ 0 & 0 & 1 & 1 & 26 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$	4	$\left\{ \begin{matrix} \begin{pmatrix} 1 & 0 & 22 & 26 & 11 \end{pmatrix}, \\ \begin{pmatrix} 0 & 1 & 7 & 2 & 10 \end{pmatrix}, \\ \begin{pmatrix} 0 & 0 & 1 & 1 & 26 \end{pmatrix}, \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \end{matrix} \right\}$
$A_2 \text{ mod}(23)$	$\begin{pmatrix} 1 & 18 & 21 & 10 & 4 & 16 \\ 0 & 1 & 4 & 0 & 15 & 10 \\ 0 & 0 & 1 & 9 & 14 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$	3	$\left\{ \begin{matrix} \begin{pmatrix} 1 & 18 & 21 & 10 & 4 & 16 \end{pmatrix}, \\ \begin{pmatrix} 0 & 1 & 4 & 0 & 15 & 10 \end{pmatrix}, \\ \begin{pmatrix} 0 & 0 & 1 & 9 & 14 & 7 \end{pmatrix}. \end{matrix} \right\}$

Table 2: Matrices converted into echelon form using Gaussian elimination

The rows of a matrix in echelon form are linearly independent. Let r_i be the i-th row of the matrix. Suppose row r_i was a linear combination of other rows. By induction on j, where $j < i$, it cannot contain a combination of rows r_j since $r_j(l(j)) = 1$, whereas $r_i(l(j) = 0$. So it must only be a combination of rows r_k where $k > i$. However this is not the case, since $r_i(l(i)) = 1$ and $r_k(l(i)) = 0$ for all k. So r_i cannot

[1]Code 1.1 was used at the start of Code 3.1 to produce the `inverses` vector, it has not been included in the referencing.

be written as a combination of the other rows and therefore the rows must be linearly independent. It follows that the rows of a matrix in echelon form is a basis for its own row-space. Using the fact $\mathrm{Rank}(A) = \mathrm{Rank}(A^T)$ we deduce that $\mathrm{Rank}(A)$ is the value of r.

Question 4

Suppose

$$x_{l(r)} + \sum_{j=l(r)+1}^{n} a_{r,j} \cdot x_j = 0 \tag{2}$$

where $A = (a_{i,j})$. There exist $n - l(r)$ vectors v where $v_j = 1$ for some $j > l(r)$, $v_{l(r)} = -a_{r,j}$ and zero everywhere else. Let S be the set of these vectors. We now iterate from $i = r - 1$ up to $i = 1$, each time doing the following:
1. Create $l(i) - l(i+1) - 1$ vectors of the form $v_j = 1$ for some $l(i-1) < j < l(i)$ and 0 everywhere else, and add them to S.
2. For all vectors $v \in S$, set

$$v_{l(i)} = -\sum_{j=l(i)+1}^{n} a_{i,j} v_j \tag{3}$$

Once this process finishes, we will have $n - l(r) + \sum_{i=1}^{r-1}(l(i+1) - l(i) - 1) = n + 1 - r - l(1) = n - r$ vectors in S.[2] The vectors in S have been constructed so that they are linearly independent, since for each $j \neq l(i)$ for some i, there is a corresponding vector v with $v_j = 1$. All other vectors in S have their jth entry equal to 0, so each vector cannot be written as a linear combination of the others. By the rank-nullity theorem, we have a complete basis for the kernel of A. This is how Code 4.1 referenced on page 10 has computed the kernels of the matrices in Table 3.[3]

[2]This is only the case when $l(1) = 1$. However we may assume this true, since the matrices in this project don't have a column of 0's at their start. If, however, A had k 0 columns on the front of it, we could introduce k more kernel vectors of the form $v_i = 1$ where $i \leq k$ and zero everywhere else. We would combine these vectors with the vectors produced by the iteration in Question 4, however these vectors would now have k zero entries on the top of them.

[3]Code 1.1 was used at the start of Code 4.1 but has not been referenced. The function `Echelon`, as defined in Code 3.1, has also been used but also not included.

Matrix	Echelon Form	Basis for Kernel
$B_1 \bmod(2)$	$\begin{pmatrix}1&0&1&0&1&0\\0&1&0&1&1&0\\0&0&1&0&1&1\\0&0&0&1&1&1\\0&0&0&0&0&0\end{pmatrix}$	$\begin{pmatrix}0\\0\\1\\1\\1\\0\end{pmatrix},\begin{pmatrix}1\\1\\1\\1\\0\\1\end{pmatrix}$
$B_1 \bmod(11)$	$\begin{pmatrix}1&7&4&6&9&3\\0&1&3&4&0&2\\0&0&1&4&10&10\\0&0&0&0&1&3\\0&0&0&0&0&1\end{pmatrix}$	$\begin{pmatrix}9\\8\\7\\1\\0\\0\end{pmatrix}$
$B_1 \bmod(17)$	$\begin{pmatrix}1&10&14&9&5&13\\0&1&1&6&10&3\\0&0&1&6&3&10\\0&0&0&1&2&1\\0&0&0&0&1&16\end{pmatrix}$	$\begin{pmatrix}7\\0\\5\\14\\1\\1\end{pmatrix}$
$B_2 \bmod(23)$	$\begin{pmatrix}1&10&14&1&3&2\\0&1&20&3&16&11\\0&0&1&17&7&5\\0&0&0&1&20&18\\0&0&0&0&1&14\end{pmatrix}$	$\begin{pmatrix}6\\6\\9\\9\\9\\1\end{pmatrix}$

Table 3: Kernels produced by Code 4.1

Question 5

If $U \subseteq F^n$ then

$$\dim(U) + \dim(U^\circ) = n \tag{4}$$

Note that when U is the row space for a matrix A, we recover the rank-nullity theorem; $\text{Rank}(A) + \text{Nul}(A) = n$

Question 6

Inputting A_1 into Code 4.1 produced the middle two columns of Table 4. The last column was produced by inputting $(U^\circ)^T$ into Code 4.1 again.

U = Row space	Echelon Form	U°	$(U^\circ)^\circ$
A_1	$\begin{pmatrix}1&0&5&3&12\\0&1&7&2&10\\0&0&1&4&7\\0&0&0&1&1\end{pmatrix}$	$\begin{pmatrix}6\\13\\16\\18\\1\end{pmatrix}$	$\begin{pmatrix}1&1&0&0&0\\0&1&17&0&0\\0&0&1&16&0\\0&0&0&1&1\end{pmatrix}$

Table 4: Kernels produced by Code 4.1

The following operations in this order converts $(U^\circ)^\circ$ back to U: $S(1,1,2), S(2,10,3), S(2,9,4), S(3,12,4)$. This verifies that $(U^\circ)^\circ = U$.

Question 7

I will prove that $\dim(U) + \dim(W) = \dim(U + W) + \dim(U \cap W)$. Suppose $\dim(U \cap W) = q$ with basis $\{x_i\}$ where $1 \leq i \leq q$, and that $\dim(U) = a$ and $\dim(W) = b$. We may extend the basis $\{x_i\}$ so that the basis of U is $\{x_i, u_j\}$ where $q + 1 \leq j \leq a$ and that the basis for W is $\{x_i, w_k\}$ where $q + 1 \leq k \leq b$. Therefore $\{x_i, u_j, w_k\}$ spans $U + W$. Assume that

$$\sum_{i=1}^{q} \lambda_i x_i + \sum_{j=q+1}^{a} \mu_j u_j + \sum_{k=q+1}^{b} \eta_k w_k = 0 \tag{5}$$

for constants λ_i, μ_j, η_k. If at least one $\eta_k \neq 0$ then there is a linear combination of vectors w_k that is in $U \cap V$, a contradiction that $\{x_i, w_k\}$ was a basis. Therefore all $\eta_k = 0$. The fact that $\{x_i, u_j\}$ is a basis then implies $\lambda_i = \mu_j = 0$ for all i, j. We conclude that $\{x_i, u_j, w_k\}$ is a set of linearly independent vectors and so forms a basis for $U + W$. There are $a + b - q$ vectors in this basis and so $\dim(U + V) + \dim(U \cap V) = a + b = \dim(U) + \dim(V)$.

The following is an explanation of how the program works. Code 3.1 already calculates a basis for U and W from matrices A and B, and so this is used at the start. If U had basis $\{u_i\}$ and W had basis $\{w_j\}$, then $S = \text{Span}\{u_i, w_j\} = U + W$. This is because if $x \in U + W, x = u + w$ for some $u \in U, w \in W$ and so $U + W \subseteq S$. Equally if $x \in S$, $x = \sum_i \lambda_i u_i + \sum_j \mu_j v_j$, for constants λ_i, μ_j, which lies in $U + W$, so $S \subseteq U + W$, and therefore $S = U + W$. Inputting the matrix $\begin{pmatrix} A \\ B \end{pmatrix}$ into Code 3.1 will give a linearly independent set of vectors with the same span, i.e. a basis for $U + W$. To calculate $U \cap W$, we must first find a basis for $U^\circ + W^\circ$, by using Code 4.1 to find the annihilators and then inputting the matrix $(\ker(A) \;\; \ker(B))^T$ into Code 3.1. Then by using Code 4.1 again we may find $(U^\circ + W^\circ)^\circ = U \cap W$.

Code 7.1 described above and referenced on page 11 was used to produce the results in Tables 5-7.[4]

Basis for U	Basis for W
$\left\{\begin{matrix}(1 & 10 & 0 & 6 & 2 & 8)\\(0 & 1 & 9 & 5 & 8 & 2)\\(0 & 0 & 1 & 5 & 6 & 0)\\(0 & 0 & 0 & 1 & 0 & 9)\end{matrix}\right\}$	$\left\{\begin{matrix}(1 & 7 & 4 & 6 & 9 & 3)\\(0 & 1 & 3 & 4 & 0 & 2)\\(0 & 0 & 1 & 4 & 10 & 10)\\(0 & 0 & 0 & 0 & 1 & 3)\\(0 & 0 & 0 & 0 & 0 & 1)\end{matrix}\right\}$
Basis for $U + W$	**Basis for $U \cap W$**
$\left\{\begin{matrix}(1 & 10 & 0 & 6 & 2 & 8)\\(0 & 1 & 9 & 5 & 8 & 2)\\(0 & 0 & 1 & 5 & 6 & 0)\\(0 & 0 & 0 & 1 & 0 & 9)\\(0 & 0 & 0 & 0 & 1 & 4)\\(0 & 0 & 0 & 0 & 0 & 1)\end{matrix}\right\}$	$\left\{\begin{matrix}(1 & 7 & 6 & 3 & 0 & 0)\\(0 & 1 & 9 & 6 & 8 & 0)\\(0 & 0 & 1 & 4 & 6 & 2)\end{matrix}\right\}$

Table 5: $A = A_2$ and $B = B_1$ working modulo 11

[4]Code 1.1 was used at the start of Code 7.1, but has not been included in the referencing. The functions `Echelon` and `Ker`, defined in the previous codes, were also used and have also not been referenced.

Basis for U	Basis for W
$\left\{\begin{array}{ccccccc}(1 & 0 & 0 & 0 & 3 & 0 & 0)\\(0 & 1 & 0 & 4 & 5 & 12 & 0)\\(0 & 0 & 1 & 0 & 8 & 0 & 0)\\(0 & 0 & 0 & 1 & 4 & 11 & 13)\\(0 & 0 & 0 & 0 & 1 & 17 & 6)\end{array}\right\}$	$\left\{\begin{array}{ccccccc}(1 & 10 & 9 & 0 & 6 & 3 & 0)\\(0 & 1 & 0 & 2 & 0 & 4 & 14)\end{array}\right\}$
Basis for $U+W$	**Basis for $U\cap W$**
$\left\{\begin{array}{ccccccc}(1 & 0 & 0 & 0 & 3 & 0 & 0)\\(0 & 1 & 0 & 4 & 5 & 12 & 0)\\(0 & 0 & 1 & 0 & 8 & 0 & 0)\\(0 & 0 & 0 & 1 & 4 & 11 & 13)\\(0 & 0 & 0 & 0 & 1 & 17 & 6)\\(0 & 0 & 0 & 0 & 0 & 1 & 14)\\(0 & 0 & 0 & 0 & 0 & 0 & 1)\end{array}\right\}$	$\left\{\begin{array}{ccccccc}(0 & 0 & 0 & 0 & 0 & 0 & 0)\end{array}\right\}$

Table 6: $A = A_3$ and $B = \mathrm{Ker}(A_3)$ working modulo 19

Basis for U	Basis for W
$\left\{\begin{array}{ccccccc}(1 & 0 & 0 & 0 & 3 & 0 & 0)\\(0 & 1 & 0 & 3 & 4 & 2 & 0)\\(0 & 0 & 1 & 0 & 6 & 0 & 0)\\(0 & 0 & 0 & 1 & 3 & 2 & 4)\\(0 & 0 & 0 & 0 & 1 & 0 & 0)\end{array}\right\}$	$\left\{\begin{array}{ccccccc}(0 & 1 & 0 & 3 & 0 & 2 & 0)\\(0 & 0 & 0 & 1 & 0 & 2 & 4)\end{array}\right\}$
Basis for $U+W$	**Basis for $U\cap W$**
$\left\{\begin{array}{ccccccc}(1 & 0 & 0 & 0 & 3 & 0 & 0)\\(0 & 1 & 0 & 3 & 4 & 2 & 0)\\(0 & 0 & 1 & 0 & 6 & 0 & 0)\\(0 & 0 & 0 & 1 & 3 & 2 & 4)\\(0 & 0 & 0 & 0 & 1 & 0 & 0)\end{array}\right\}$	$\left\{\begin{array}{ccccccc}(0 & 1 & 0 & 3 & 0 & 2 & 0)\\(0 & 0 & 0 & 1 & 0 & 2 & 4)\end{array}\right\}$

Table 7: $A = A_3$ and $B = \mathrm{Ker}(A_3)$ working modulo 7

Question 8

In the last part of Question 7 the intersection of the kernel and the row space is non trivial. If we were working over the real numbers, this would be impossible. Suppose that a non zero vector x lay in the kernel and row space of A, then $Ax^T = 0$. Therefore the dot product of x^T with any of the rows of A would equal zero. There also exists a linear combination of rows that equals x, so we conclude that $xx^T = 0$. Working over $\mathbb{R}$, this implies that x is the zero vector, and so the intersection must be trivial. However we cannot conclude that $x = 0$ if we work over $\mathrm{GF}(p)$. To give an example, take $p = 5$ and $x = (2, 4)$.

Code

Code 1.1

```
%Initial data
p=11;
a=1;
b=1;

tic
inverses=zeros(p-1,1);
while a<p
    while 1
        %This checks if b is the inverse of a, and if so, stores it then
        %breaks the loop.
        if mod(a*b,p)==1
            inverses(a)=b;
            b=1;
            break
        end
        b=b+1;
    end
    a=a+1;
end
disp(inverses)
toc
```

Code 1.2

```
%Initial data
p=11;
a=1;
b=1;
tic
inverses=zeros(p-1,1);
while a<p
    if inverses(a)==0
        while 1
            %This checks if b is the inverse of a, and if so, stores it then
            %breaks the loop.
            if mod(a*b,p)==1
                inverses(a)=b;
                inverses(b)=a;
                b=1;
                break
            end
            b=b+1;
        end
    end
    a=a+1;
end
disp(inverses)
toc
```

Code 3.1

```
%Initial data
m=[0,1,7,2,10;8,0,2,5,1;2,1,2,5,5;7,4,5,3,0];
p=3;

[m,rank]=Echelon(m,p,inverses)

function [m,rank] = Echelon(m,p,inverses)
%First this makes sure all elements of the matrix lie within mod p
m=mod(m,p);
%The dimensions of the matrix are used for the limits of the while loop
dimensions=size(m);
columns=1;
rows=1;
%rank tracks how many rows have already changed to echelon form
rank=0;
while columns<=dimensions(2)
    %For each column, we search downwards until we find a non-zero element.
    %If we are successful, we convert this column and the top most possible
    %row into echelon form, before converting the 'lower-right' matrix into
    %echelon form. Otherwise we move onto the next column.
    while rows<=dimensions(1)
        if m(rows,columns)~=0
            m=T(m,rows,rank+1);
            m=D(m,rank+1,m(rank+1,columns),inverses,p);
            rows=rank+2;
            while rows<=dimensions(1)
                m=S(m,rows,m(rows,columns),rank+1,p);
                rows=rows+1;
            end
            rank=rank+1;
            break
        end
        rows=rows+1;
    end
    %We continue to convert the 'lower-right' matrix into echelon form
    rows=rank+1;
    columns=columns+1;
end
end

function m = T(m,i,j)
    v=m(i,:);
    m(i,:)=m(j,:);
    m(j,:)=v;
end

%This divides a row i by a by indexing its inverse from 'inverses'
function m = D(m,i,a,inverses,p)
    v=m(i,:);
    v=mod(inverses(a)*v,p);
    m(i,:)=v;
end

%This takes a multiple a of row j and minuses this from row i
```

```
function m = S(m,i,a,j,p)
    if i==j
        error("i should not equal j")
    else
        m(i,:)=m(i,:)-a*m(j,:);
    end
    %This makes sure that the matrix still lies in mod(p)
    m=mod(m,p);
end
```

Code 4.1

```
m=[4,6,5,2,3,1;5,0,3,0,1,0;1,5,7,1,0,12;5,5,0,3,1,7;2,1,2,4,0,5];
p=2;

[m,rank]=Echelon(m,p,inverses)
ker=Ker(m,p,rank)

function basis = Ker(m,p,rank)
%This creates a blank array to be filled with vectors for the basis
basis=zeros(size(m,2),size(m,2)-rank);

%This resets the rows,columns and noBasis (which represents the number of
%basis vectors created) counters.
rows=size(m,1);
columns=1;
noBasis=1;
lastColumn=size(m,2)+1;

%This iterates through until it finds a non-zero element
while rows >= 1
    while columns <= size(m,2)
        if m(rows,columns)~=0
            %If there is a 1 in the far right of the matrix, this ensures
            %the last element of the vector is 0
            if columns==size(m,2)
                lastColumn=columns;
            else
                %This iterates between the column where the non-zero
                %element is, and the last column that caused a vector to be
                %produced. It creates new linearly-independant vectors for
                %the basis, before ensuring that the vector lies in the
                %kernal
                thisColumn=columns;
                columns=columns+1;
                while columns<lastColumn
                    basis(columns,noBasis)=1;
                    noBasis=noBasis+1;
                    columns=columns+1;
                end
                basis(thisColumn,:)=-m(rows,:)*basis;
                lastColumn=thisColumn;
            end
            break
        end
        columns=columns+1;
    end
    columns=1;
    rows=rows-1;
end
%This ensures that the vector lies within mod(p)
basis=mod(basis,p);
end
```

Code 7.1

```
A=[6,16,11,14,1,4;7,9,1,1,21,0;8,2,9,12,17,7;2,19,2,19,7,12];
B=[4,6,5,2,3,1;5,0,3,0,1,0;1,5,7,1,0,12;5,5,0,3,1,7;2,1,2,4,0,5];
p=11;

%This calculates the basis for U and W, as well as the kernels of A and B
[U,rank]=Echelon(A,p,inverses);
disp('U')
disp(U)
KerU=Ker(U,p,rank);
[W,rank]=Echelon(B,p,inverses);
disp('W')
disp(W)
KerW=Ker(W,p,rank);

%This calculates the basis for U+W, using the theory described in Q7
V=[U;W];
V=Echelon(V,p,inverses);
disp('U+W')
disp(V)

%This calculates a basis for U^o+W^o, and then calculates the kernel of
%this for the basis of U intersect W.
X=[KerU KerW]';
[X,rank]=Echelon(X,p,inverses);
X=Ker(X,p,rank)';
X=Echelon(X,p,inverses);
disp('U intersect W')
disp(X)
```