In [ ]:
```
Q1 and Q2:
```

In [ ]:
```
A = matrix(FiniteField(5), 3, [1,1,2,4, 0, 4,3,2,   -2,2,-1,4]);
```

In [ ]:
```
A
```

In [ ]:
```
A1 = matrix(QQ, 3, [1,1,2,4, 0, 4,3,2,   -2,2,-1,4]);
A1
```

In [ ]:
```
A1.rref()   # note this is rref over R, not correct!
```

In [ ]:
```
A.rref()   # correct answer for Q1
```

In [ ]:
```
C = matrix(FiniteField(5), 3, [1,2,3, 0,1,3, 0,2,1]);
```

In [ ]:
```
C
```

In [ ]:
```
C.rref()
```

Q 2: basis of kernel:[1,0,0] and [2,1,2] basis of image: [2,3,-1] or multiple of it.

In [ ]:
```

```

In [ ]:
```

```

In [ ]:
```

```

In [ ]:
```

```

In [ ]:
```

```

Q8: KerHbig= Im Gbig = Vhamming

In [ ]:
```
string1 = "1 1 0 0 1 1 0 0 0 1 1 1 0 0 0 1 0 0 1 0 1 1 0 0 0 1 0 1 0 0 1 1 0 0 0
 1"
print (string1)
print (string1.lstrip())
```

In [ ]:
```
Hold = [string1.replace(" ", "")]
Hold
Hlist=list([int(y)  for x in Hold for y in x])
Hlist
```

In [111]:
```
Hbig =  matrix(FiniteField(2), 4, Hlist);
Hbig
```

Out[111]:
```
[1 1 0 0 1 1 0 0 0]
[1 1 1 0 0 0 1 0 0]
[1 0 1 1 0 0 0 1 0]
[1 0 0 1 1 0 0 0 1]
```

In [ ]:
```
Q9:  only v1 is in image Gbig.
```

In [ ]:
```
v1= matrix(FiniteField(2), 9,  [1,0,0,1,1,0,1,0,1]);
```

In [112]:
```
Hbig*v1
```

Out[112]:
```
[0]
[0]
[0]
[0]
```

In [ ]:
```
v2= matrix(FiniteField(2), 9, [1,0,1,1,0,0,0,1,0]);
```

In [113]:
```
Hbig*v2
```

Out[113]:
```
[1]
[0]
[0]
[0]
```

In [114]:
```
v3=  matrix(FiniteField(2), 9,  [1,1,0,1,1,1,1,0,1])
Hbig*v3
```

Out[114]:
```
[0]
[1]
[0]
[0]
```

Q10: x is the first 5 digits (element) of y.

Q11: Read the error message Hbig·y, find which column of Hbig matches this vector, then the corresponding digit of y is incorrect

In [ ]:
```
Q12: code see below:
```

In [ ]:
```
filename="C:/Users/Jun Li/Desktop/214/code.txt"
with open(filename) as f:
    mylist = f.read().splitlines()
D=[int(y)  for x in mylist for y in x]
D
```

In [ ]:
```
S = matrix(FiniteField(2), 66, D);
S
```

In [ ]:
```
S.str()
```

In [115]:
```
Hbig
```

Out[115]:
```
[1 1 0 0 1 1 0 0 0]
[1 1 1 0 0 0 1 0 0]
[1 0 1 1 0 0 0 1 0]
[1 0 0 1 1 0 0 0 1]
```

In [116]:
```
M = Hbig*S.T
```

In [117]:
```
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:77% !important; }</style>"))
M.str()
```

Out[117]:
```
'[0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]\n[0 1 0 1 0 1 0 0 0 0 0
0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]\n[1 1 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 1 0 0 0 0 0 0
1 0 0 0 1 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0
1]\n[1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0 1 0 0
0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 1 1 0 0 0 0 0 0 0]'
```

In [ ]:
```
#a=np.array(M); M
```

In [ ]:
```
 w = zero_vector(SR, 66*9) ; w
```

In [ ]:
```
Cr= matrix(FiniteField(2), 66,  w); Cr
```

In [ ]:
```
Cr[0]
```

In [ ]:
```
import numpy as np
a=np.array(M)
b=np.array(Hbig)
def crlist(M):
    results = []
    i = 0
    while i <= 65:
        ci=0
        for j in range(9):
            if  np.array_equal(a[:,i],  b[:,j]) :
                results.append(j+1)
                ci=1
                break
        if (ci==0):
            results.append(0)
       # print(i)
       # print(results)
        i += 1
    return results
#print(crlist)
print(crlist(M))
```

In [ ]:
```
a[:,0]
```

In [121]:
```
c=np.array(crlist(M))
d=np.array(Cr)
i = 0
while i <= 65:
    if  c[i] != 0:
        d[i, c[i]-1 ]=1
    i= i+1
print(d)  # find the correction code for each message
```

```
[[0 0 0 1 0 0 0 0 0]
 [0 0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0]
 [0 0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]
 [0 0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0]
 [0 0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [1 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0]]
```

In [122]:
```
F=d+S; F   # Add the correction to the original message Bob received.
```

Out[122]:
```
array([[1, 0, 1, 0, 0, 1, 0, 0, 1],
       [0, 1, 0, 0, 0, 1, 1, 0, 0],
       [0, 0, 1, 0, 1, 1, 1, 1, 1],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 1, 1, 1, 1, 0, 0, 0, 0],
       [0, 1, 1, 1, 0, 1, 0, 0, 1],
       [0, 1, 1, 0, 0, 1, 0, 1, 0],
       [1, 1, 0, 0, 1, 1, 0, 1, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [1, 0, 1, 1, 1, 0, 0, 1, 1],
       [0, 0, 0, 0, 1, 1, 0, 0, 1],
       [1, 1, 0, 0, 1, 1, 0, 1, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [1, 0, 1, 0, 0, 1, 0, 0, 1],
       [0, 1, 1, 1, 1, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 1, 1, 0, 0, 1, 0, 1, 0],
       [0, 0, 1, 0, 1, 1, 1, 1, 1],
       [0, 0, 0, 0, 1, 1, 0, 0, 1],
       [1, 0, 0, 1, 0, 1, 1, 0, 0],
       [0, 1, 1, 1, 0, 1, 0, 0, 1],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 1, 1, 0, 1, 0, 0, 1, 1],
       [0, 0, 0, 0, 1, 1, 0, 0, 1],
       [1, 0, 1, 0, 0, 1, 0, 0, 1],
       [0, 1, 0, 0, 0, 1, 1, 0, 0],
       [0, 0, 1, 0, 1, 1, 1, 1, 1],
       [0, 1, 1, 0, 1, 0, 0, 1, 1],
       [0, 0, 0, 0, 1, 1, 0, 0, 1],
       [1, 0, 1, 0, 0, 1, 0, 0, 1],
       [0, 1, 0, 0, 1, 0, 1, 0, 1],
       [0, 0, 0, 1, 1, 1, 0, 1, 0],
       [1, 0, 0, 1, 1, 0, 1, 0, 1],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 1, 0, 0, 1, 0, 1, 0, 1],
       [1, 0, 0, 1, 1, 0, 1, 0, 1],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [1, 0, 1, 0, 0, 1, 0, 0, 1],
       [0, 1, 1, 1, 1, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 1, 0, 0, 0, 1, 1, 0],
       [0, 1, 1, 1, 1, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 1, 1, 0, 1, 0, 0, 1, 1],
       [0, 0, 0, 0, 1, 1, 0, 0, 1],
       [1, 0, 1, 0, 0, 1, 0, 0, 1],
       [0, 1, 0, 0, 0, 1, 1, 0, 0],
       [0, 0, 1, 0, 1, 1, 1, 1, 1],
       [0, 1, 1, 0, 1, 0, 0, 1, 1],
       [0, 0, 0, 0, 1, 1, 0, 0, 1],
       [1, 0, 1, 0, 0, 1, 0, 0, 1],
       [0, 1, 0, 0, 1, 0, 1, 0, 1],
       [0, 0, 0, 1, 1, 1, 0, 1, 0],
       [1, 0, 0, 1, 1, 0, 1, 0, 1],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [1, 0, 0, 0, 0, 1, 1, 1, 1],
       [0, 0, 0, 0, 1, 1, 0, 0, 1],
       [1, 0, 1, 0, 1, 0, 0, 0, 0],
       [0, 1, 1, 0, 0, 1, 0, 1, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 1, 0, 0, 0, 1, 1, 0, 0],
       [0, 0, 0, 0, 1, 1, 0, 0, 1],
       [0, 1, 1, 0, 0, 1, 0, 1, 0],
       [0, 1, 1, 0, 1, 0, 0, 1, 1],
       [0, 1, 1, 1, 1, 0, 0, 0, 0],
       [1, 0, 0, 1, 1, 0, 1, 0, 1]], dtype=object)
```

In [123]:
```
F[:,:5] # read the first five digit as the original code Alice sent.
```

Out[123]:
```
array([[1, 0, 1, 0, 0],
       [0, 1, 0, 0, 0],
       [0, 0, 1, 0, 1],
       [0, 0, 0, 0, 0],
       [0, 1, 1, 1, 1],
       [0, 1, 1, 1, 0],
       [0, 1, 1, 0, 0],
       [1, 1, 0, 0, 1],
       [0, 0, 0, 0, 0],
       [1, 0, 1, 1, 1],
       [0, 0, 0, 0, 1],
       [1, 1, 0, 0, 1],
       [0, 0, 0, 0, 0],
       [1, 0, 1, 0, 0],
       [0, 1, 1, 1, 1],
       [0, 0, 0, 0, 0],
       [0, 1, 1, 0, 0],
       [0, 0, 1, 0, 1],
       [0, 0, 0, 0, 1],
       [1, 0, 0, 1, 0],
       [0, 1, 1, 1, 0],
       [0, 0, 0, 0, 0],
       [0, 1, 1, 0, 1],
       [0, 0, 0, 0, 1],
       [1, 0, 1, 0, 0],
       [0, 1, 0, 0, 0],
       [0, 0, 1, 0, 1],
       [0, 1, 1, 0, 1],
       [0, 0, 0, 0, 1],
       [1, 0, 1, 0, 0],
       [0, 1, 0, 0, 1],
       [0, 0, 0, 1, 1],
       [1, 0, 0, 1, 1],
       [0, 0, 0, 0, 0],
       [0, 1, 0, 0, 1],
       [1, 0, 0, 1, 1],
       [0, 0, 0, 0, 0],
       [1, 0, 1, 0, 0],
       [0, 1, 1, 1, 1],
       [0, 0, 0, 0, 0],
       [0, 0, 1, 0, 0],
       [0, 1, 1, 1, 1],
       [0, 0, 0, 0, 0],
       [0, 1, 1, 0, 1],
       [0, 0, 0, 0, 1],
       [1, 0, 1, 0, 0],
       [0, 1, 0, 0, 0],
       [0, 0, 1, 0, 1],
       [0, 1, 1, 0, 1],
       [0, 0, 0, 0, 1],
       [1, 0, 1, 0, 0],
       [0, 1, 0, 0, 1],
       [0, 0, 0, 1, 1],
       [1, 0, 0, 1, 1],
       [0, 0, 0, 0, 0],
       [1, 0, 0, 0, 0],
       [0, 0, 0, 0, 1],
       [1, 0, 1, 0, 1],
       [0, 1, 1, 0, 0],
       [0, 0, 0, 0, 0],
       [0, 1, 0, 0, 0],
       [0, 0, 0, 0, 1],
       [0, 1, 1, 0, 0],
       [0, 1, 1, 0, 1],
       [0, 1, 1, 1, 1],
       [1, 0, 0, 1, 1]], dtype=object)
```

In [ ]:
```
F[0,:5]
```

In [119]:
```
type(F)
```

Out[119]:
```
<class 'numpy.ndarray'>
```

In [120]:
```
letter_ind=[int(m,2) for m in [''.join([str(i) for i in x]) for x in F[:,:5]]]
```

In [124]:
```
import string
letters=[' ']
letters.extend(list(string.ascii_lowercase))
letters=np.array(letters)
letters   # match the vector x with letters.
```

Out[124]:
```
array([' ', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l',
       'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y',
       'z'], dtype='<U1')
```

In [127]:
```
''.join(letters[letter_ind])
```

Out[127]:
```
'the only way to learn mathematics is to do mathematics paul halmos'
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [126]:
```
b[:,0]
```

Out[126]:
```
array([1, 1, 1, 1], dtype=object)
```

In [ ]: