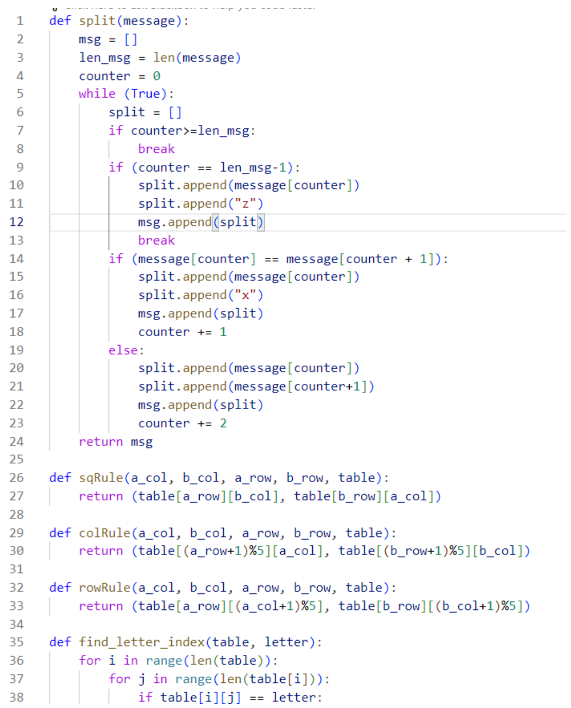
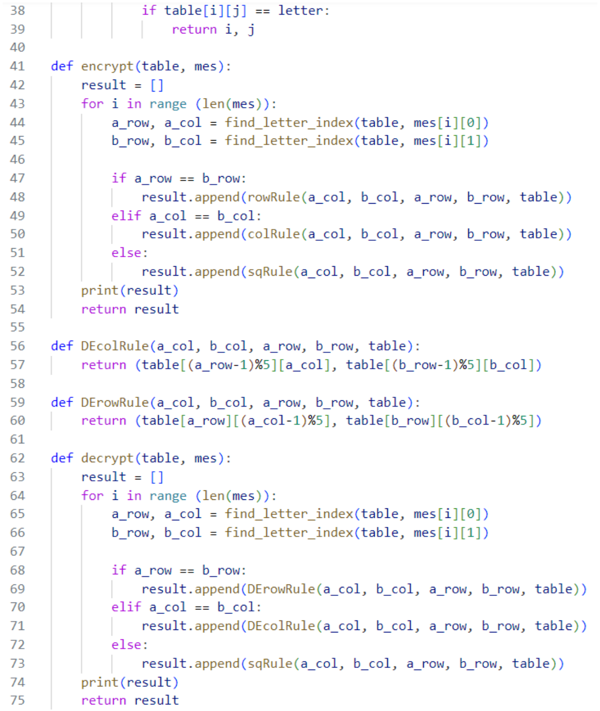
Syed Shaheer Ali Zaidi B20102168

Play Fair Cipher

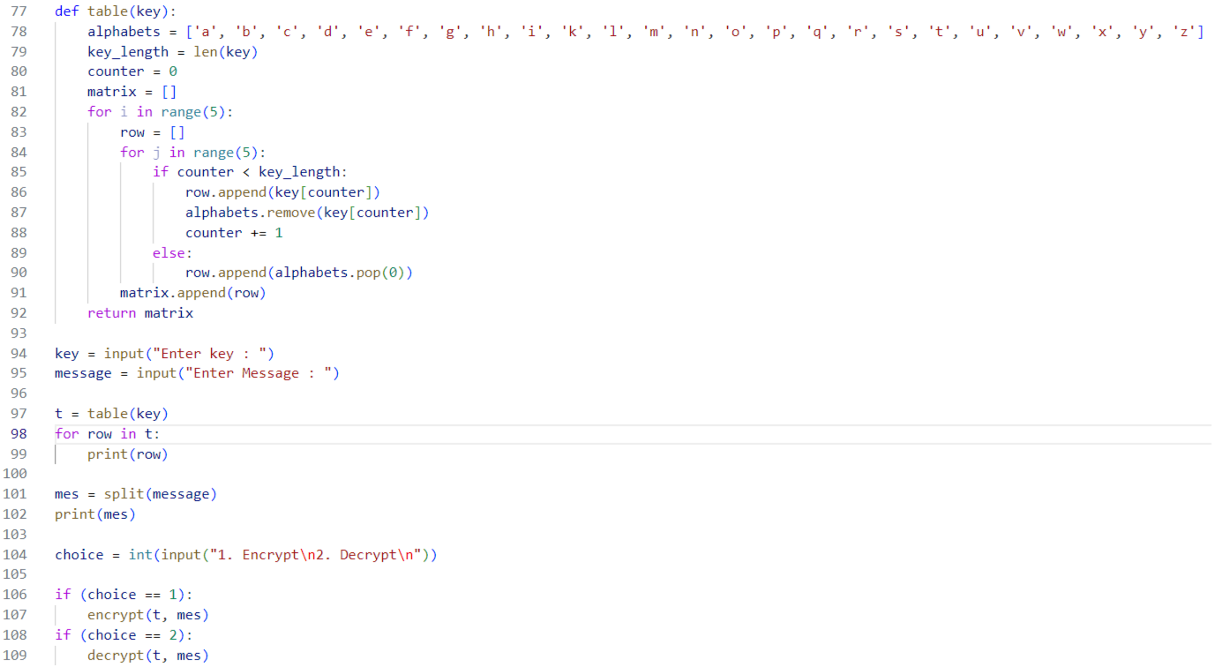
Code:



```python
def split(message):
    msg = []
    len_msg = len(message)
    counter = 0
    while (True):
        split = []
        if counter>=len_msg:
            break
        if (counter == len_msg-1):
            split.append(message[counter])
            split.append("z")
            msg.append(split)
            break
        if (message[counter] == message[counter + 1]):
            split.append(message[counter])
            split.append("x")
            msg.append(split)
            counter += 1
        else:
            split.append(message[counter])
            split.append(message[counter+1])
            msg.append(split)
            counter += 2
    return msg

def sqRule(a_col, b_col, a_row, b_row, table):
    return (table[a_row][b_col], table[b_row][a_col])

def colRule(a_col, b_col, a_row, b_row, table):
    return (table[(a_row+1)%5][a_col], table[(b_row+1)%5][b_col])

def rowRule(a_col, b_col, a_row, b_row, table):
    return (table[a_row][(a_col+1)%5], table[b_row][(b_col+1)%5])

def find_letter_index(table, letter):
    for i in range(len(table)):
        for j in range(len(table[i])):
            if table[i][j] == letter:
```



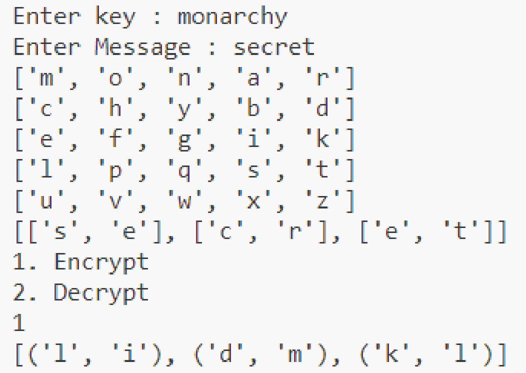
```python
            if table[i][j] == letter:
                return i, j

def encrypt(table, mes):
    result = []
    for i in range (len(mes)):
        a_row, a_col = find_letter_index(table, mes[i][0])
        b_row, b_col = find_letter_index(table, mes[i][1])

        if a_row == b_row:
            result.append(rowRule(a_col, b_col, a_row, b_row, table))
        elif a_col == b_col:
            result.append(colRule(a_col, b_col, a_row, b_row, table))
        else:
            result.append(sqRule(a_col, b_col, a_row, b_row, table))
    print(result)
    return result

def DEcolRule(a_col, b_col, a_row, b_row, table):
    return (table[(a_row-1)%5][a_col], table[(b_row-1)%5][b_col])

def DErowRule(a_col, b_col, a_row, b_row, table):
    return (table[a_row][(a_col-1)%5], table[b_row][(b_col-1)%5])

def decrypt(table, mes):
    result = []
    for i in range (len(mes)):
        a_row, a_col = find_letter_index(table, mes[i][0])
        b_row, b_col = find_letter_index(table, mes[i][1])

        if a_row == b_row:
            result.append(DErowRule(a_col, b_col, a_row, b_row, table))
        elif a_col == b_col:
            result.append(DEcolRule(a_col, b_col, a_row, b_row, table))
        else:
            result.append(sqRule(a_col, b_col, a_row, b_row, table))
    print(result)
    return result
```



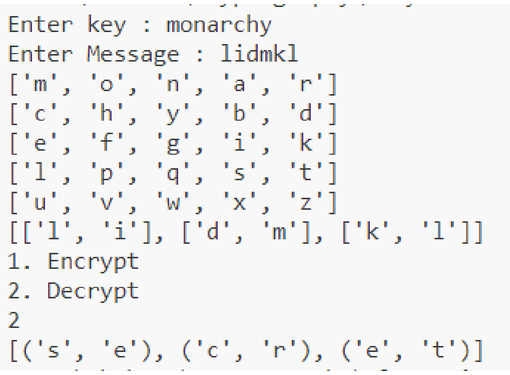
```python
def table(key):
    alphabets = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']
    key_length = len(key)
    counter = 0
    matrix = []
    for i in range(5):
        row = []
        for j in range(5):
            if counter < key_length:
                row.append(key[counter])
                alphabets.remove(key[counter])
                counter += 1
            else:
                row.append(alphabets.pop(0))
        matrix.append(row)
    return matrix

key = input("Enter key : ")
message = input("Enter Message : ")

t = table(key)
for row in t:
    print(row)

mes = split(message)
print(mes)

choice = int(input("1. Encrypt\n2. Decrypt\n"))

if (choice == 1):
    encrypt(t, mes)
if (choice == 2):
    decrypt(t, mes)
```

Syed Shaheer Ali Zaidi B20102168

Output:



```
Enter key : monarchy
Enter Message : secret
['m', 'o', 'n', 'a', 'r']
['c', 'h', 'y', 'b', 'd']
['e', 'f', 'g', 'i', 'k']
['l', 'p', 'q', 's', 't']
['u', 'v', 'w', 'x', 'z']
[['s', 'e'], ['c', 'r'], ['e', 't']]
1. Encrypt
2. Decrypt
1
[('l', 'i'), ('d', 'm'), ('k', 'l')]
```



```
Enter key : monarchy
Enter Message : lidmkl
['m', 'o', 'n', 'a', 'r']
['c', 'h', 'y', 'b', 'd']
['e', 'f', 'g', 'i', 'k']
['l', 'p', 'q', 's', 't']
['u', 'v', 'w', 'x', 'z']
[['l', 'i'], ['d', 'm'], ['k', 'l']]
1. Encrypt
2. Decrypt
2
[('s', 'e'), ('c', 'r'), ('e', 't')]
```