Group Members:

1)	**NIRINGIYIMANA Reverien**	**221012906**
2)	**IRIHOSE Eric**	**221001221**
3)	**MUGISHA Remy**	**221008577**
4)	**SHEJA Kevin**	**220006864**
5)	**MASEZERANO Esther Safina**	**221008118**

1) Describe Playfair Cipher

The **Playfair Cipher** is a digraph substitution cipher that encrypts pairs of letters instead of individual letters. It was invented in 1854 by **Charles Wheatstone** but is named after **Lord Playfair**, who promoted its use.

Encryption Steps:

1. **Construct Key Square matrix:**
    - A 5×5 matrix is created using a keyword and repeating letters are removed.
    - The remaining alphabet letters are added (I and J are considered the same and are combined).
2. **Dividing the Plaintext**:
    - The message is split into digraphs (pairs of 2 letters).
    - If a pair has the same letter (e.g., "HELLO" → "HE", "LX", "LO"), an **X** is inserted between them.
3. **Encryption Rules**:
    - **Same Row**: Replace each letter with the one to its **right** and wrap around if at the end.
    - **Same Column**: Replace each letter with the one **below** (wrap around if at the bottom).
    - **Different Row & Column**: Replace letters by forming a rectangle; each letter is swapped with the one in the same row but in the other column.

Example:

Using “PLAYFAIR” as a **Key encrypt “HELLO” using the Playfair cipher**

Step 1: Construct **the 5×5 Key Matrix**

Let’s use the key: "PLAYFAIR" to create the matrix. The remaining alphabet is filled after removing duplicate letters and treating I and J as the same.

Key Matrix:

P L A Y F

I/J R B C D

E G H K M

N O Q S T

U V W X Z

Step 2: Break the plain text into digraphs (pairs of two letters):

- "HE", "LX", "LO"
- If a duplicate letter appears (e.g., "LL"), insert X to separate them.

Step 3: Encrypt Each Pair Using Playfair Rules

Pair 1: "HE"

- Same row → Replace with next right letters
- H → K (right of H)
- E → G (right of E)
- Encrypted: "KG"

Pair 2: "LX"

- Different row & column → Form a rectangle
- L → Y (same row as L, column of X)
- X → V (same row as X, column of L)
- Encrypted: "YV"

Pair 3: "LO"

- Same column → Replace with below letters
- L → R
- O → V
- Encrypted: "RV"

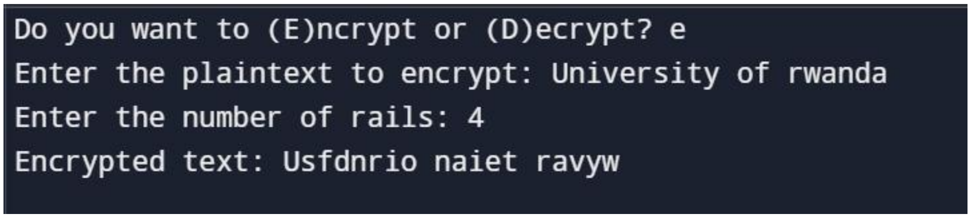
Encrypted Text:

Plain text: "HELLO"
Encrypted: "KG YV RV"

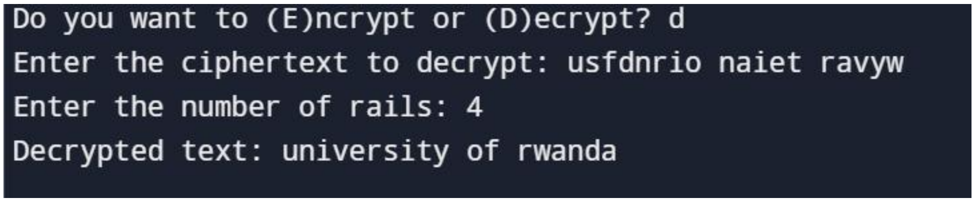
2) write a program to implement Railfence cipher using Python programming language

```
def encrypt_rail_fence(plaintext, num_rails):
    # Create a grid to represent the rails
    rails = [[''] * len(plaintext) for _ in range(num_rails)]
    # Fill the grid following the zigzag pattern
    rail = 0
    direction = 1
    for i in range(len(plaintext)):
        rails[rail][i] = plaintext[i]
        rail += direction
        if rail == 0 or rail == num_rails - 1:
            direction = -direction
    # Join the characters in each rail to form the ciphertext
    ciphertext = ''.join(''.join(rail) for rail in rails)
    return ciphertext
def decrypt_rail_fence(ciphertext, num_rails):
```

```
# Create a grid to represent the rails
n = len(ciphertext)
rails = [[''] * n for _ in range(num_rails)]
# Find the pattern of the rail fence
rail = 0
direction = 1
for i in range(n):
    rails[rail][i] = '*'
    rail += direction
    if rail == 0 or rail == num_rails - 1:
        direction = -direction
# Now fill the grid with the ciphertext
index = 0
for r in range(num_rails):
    for c in range(n):
        if rails[r][c] == '*' and index < n:
            rails[r][c] = ciphertext[index]
            index += 1
# Read the grid in a zigzag manner to get the decrypted text
decrypted_text = []
rail = 0
direction = 1
for i in range(n):
        decrypted_text.append(rails[rail][i])
        rail += direction
        if rail == 0 or rail == num_rails - 1:
            direction = -direction
    return ''.join(decrypted_text)
# Main function to interact with the user
def main():
    choice = input("Do you want to (E)ncrypt or (D)ecrypt? ").strip().lower()
    if choice == 'e':
        plaintext = input("Enter the plaintext to encrypt: ")
        num_rails = int(input("Enter the number of rails: "))
        encrypted_text = encrypt_rail_fence(plaintext, num_rails)
        print(f"Encrypted text: {encrypted_text}")
    elif choice == 'd':
        ciphertext = input("Enter the ciphertext to decrypt: ")
        num_rails = int(input("Enter the number of rails: "))
        decrypted_text = decrypt_rail_fence(ciphertext, num_rails)
        print(f"Decrypted text: {decrypted_text}")
    else:
        print("Invalid choice! Please select either 'E' for encryption or 'D' for decryption.")
# Run the program
if __name__ == "__main__":
    main()
```



```
Do you want to (E)ncrypt or (D)ecrypt? e
Enter the plaintext to encrypt: University of rwanda
Enter the number of rails: 4
Encrypted text: Usfdnrio naiet ravyw
```



```
Do you want to (E)ncrypt or (D)ecrypt? d
Enter the ciphertext to decrypt: usfdnrio naiet ravyw
Enter the number of rails: 4
Decrypted text: university of rwanda
```