Name – Aryan Sharma

Enrollment No – BT21GCS161

Computer Network (CS – 261)

Assignment -07

Activity – Hamming Code

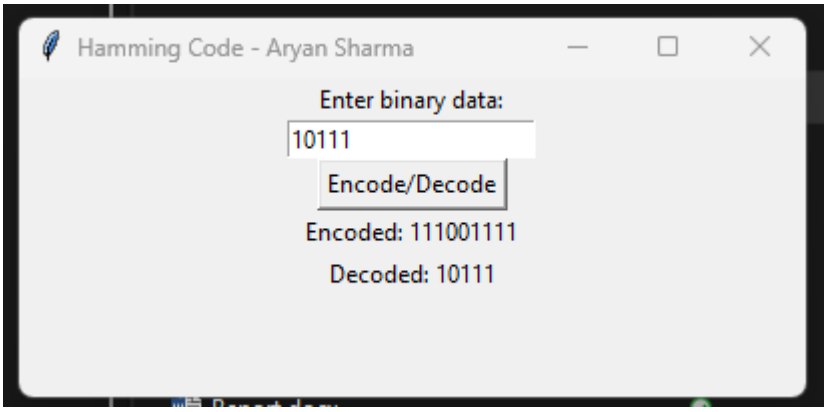
Code-


```python
import tkinter as tk
from tkinter import messagebox

def encode(data):
    r = 0
    while 2**r < len(data) + r + 1:
        r += 1

    encoded_message = [0] * (len(data) + r)
    i, j = 0, 0

    for pos in range(1, len(data) + r + 1):
        if pos == 2**i:
            encoded_message[pos - 1] = 0
            i += 1
        else:
            encoded_message[pos - 1] = data[j]
            j += 1

    for i in range(r):
        parity_bit_pos = 2**i
        parity_bit_value = 0
        for j in range(parity_bit_pos - 1, len(encoded_message), 2*parity_bit_pos):
            for k in range(j, j + parity_bit_pos):
                if k < len(encoded_message):
                    parity_bit_value ^= encoded_message[k]
        encoded_message[parity_bit_pos - 1] = parity_bit_value

    return encoded_message
```

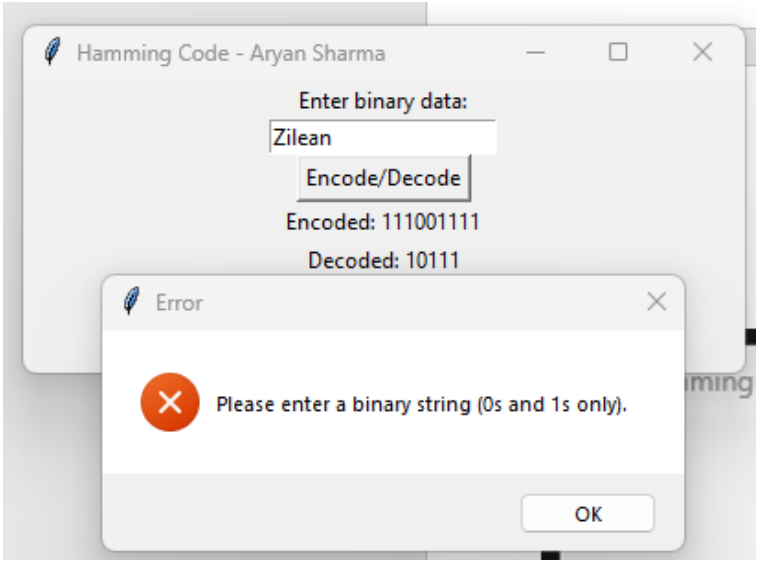
```python
def decode(encoded_message):
    r = 0
    while 2**r < len(encoded_message):
        r += 1

    error_position = 0

    for i in range(r):
        parity_bit_pos = 2**i
        parity_bit_value = 0
        for j in range(parity_bit_pos - 1, len(encoded_message), 2*parity_bit_pos):
            for k in range(j, j + parity_bit_pos):
                if k < len(encoded_message):
                    parity_bit_value ^= encoded_message[k]
        if parity_bit_value != 0:
            error_position += parity_bit_pos

    if error_position > 0:
        encoded_message[error_position - 1] ^= 1

    decoded_data = []
    j = 0
    for i in range(len(encoded_message)):
        if i == 2**j - 1:
            j += 1
        else:
            decoded_data.append(encoded_message[i])

    return decoded_data
```

```python
def encode_decode():
    input_data = entry_data.get()

    if set(input_data) <= {'0', '1'}:
        data = list(map(int, input_data))
        encoded = encode(data)
        decoded = decode(encoded)

        label_encoded.config(text="Encoded: " + ''.join(map(str, encoded)))
        label_decoded.config(text="Decoded: " + ''.join(map(str, decoded)))
    else:
        messagebox.showerror("Error", "Please enter a binary string (0s and 1s only).")

root = tk.Tk()
root.title("Hamming Code")

label_input = tk.Label(root, text="Enter binary data:")
label_input.pack()

entry_data = tk.Entry(root)
entry_data.pack()

button_encode_decode = tk.Button(root, text="Encode/Decode", command=encode_decode)
button_encode_decode.pack()

label_encoded = tk.Label(root, text="Encoded:")
label_encoded.pack()

label_decoded = tk.Label(root, text="Decoded:")
label_decoded.pack()

root.mainloop()
```

Output –


Hamming Code - Aryan Sharma
Enter binary data:
10111
Encode/Decode
Encoded: 111001111
Decoded: 10111

Error –


Hamming Code - Aryan Sharma
Enter binary data:
Zilean
Encode/Decode
Encoded: 111001111
Decoded: 10111
Error
Please enter a binary string (0s and 1s only).
OK