Computer Networks Lab

Assessment – 3

Name: Tanuj Katyal

Reg No: 22BCE0598

1. CRC

```
#include <iostream>
#include <vector>
using namespace std;

void xorOperation(vector<int>& dividend, const vector<int>& divisor, int start) {
   for (int i = 0; i < divisor.size(); i++) {
      dividend[start + i] ^= divisor[i];
   }
}

vector<int> crcRemainder(vector<int> dividend, const vector<int>& divisor) {
   int n = divisor.size();
   for (int i = 0; i <= dividend.size() - n; i++) {
      if (dividend[i] == 1) {
         xorOperation(dividend, divisor, i);
      }
   }
   vector<int> remainder(dividend.end() - (n - 1), dividend.end());
   return remainder;
}
```

```
vector<int> generateCRCCode(const vector<int>& data, const vector<int>& remainder)
{
   vector<int> crcCode = data;
   crcCode.insert(crcCode.end(), remainder.begin(), remainder.end());
   return crcCode;
}

bool checkForErrors(vector<int> receivedData, const vector<int>& divisor) {
   vector<int> remainder = crcRemainder(receivedData, divisor);
   for (int bit : remainder) {
      if (bit == 1) return true;
   }
   return false;
}

int main() {
   int dataSize, divisorSize;
   cout << "Enter the size of the data array: ";
   cin >> dataSize;

   vector<int> data(dataSize);
   cout << "Enter data bits in the array one by one: ";
   for (int i = 0; i < dataSize; i++) {
      cin >> data[i];
   }

   cout << "Enter the size of the divisor array: ";
   cin >> divisorSize;
```

```
vector<int> divisor(divisorSize);
cout << "Enter divisor bits in the array one by one: ";
for (int i = 0; i < divisorSize; i++) {
   cin >> divisor[i];
}

vector<int> paddedData = data;
paddedData.insert(paddedData.end(), divisorSize - 1, 0);

vector<int> remainder = crcRemainder(paddedData, divisor);
vector<int> crcCode = generateCRCCode(data, remainder);

cout << "CRC Remainder: ";
for (int bit : remainder) cout << bit;
cout << "\nGenerated CRC code is: ";
for (int bit : crcCode) cout << bit;
cout << endl;

cout << "Case1:\nEnter bits in the array which you want to send: ";
vector<int> receivedData(crcCode.size());
for (int i = 0; i < crcCode.size(); i++) {
   cin >> receivedData[i];
}

if (checkForErrors(receivedData, divisor)) {
   cout << "Data received with error." << endl;
} else {
```

```
    cout << "Data received without any error." << endl;
  }

  return 0;
}
```

Output-

```
Enter the size of the data array: 8
Enter data bits in the array one by one: 1 0 1 0 1 0 1 0
Enter the size of the divisor array: 4
Ent f5f5f5 bits in the array one by one: 1 1 0 0
CRC Remainder: 000
Generated CRC code is: 10101010000
Case1:
Enter bits in the array which you want to send: 10101010000
Data received without any error.


=== Code Execution Successful ===
```

2. Hamming Code

```
import java.util.Scanner;


public class HammingCode {


  public static String encode(String data) {
    int dataLength = data.length();
    int parityBits = 0;
    while (Math.pow(2, parityBits) < (dataLength + parityBits + 1)) {
      parityBits++;
    }
```

```
    char[] encoded = new char[dataLength + parityBits];
    int j = 0;
    for (int i = 0; i < encoded.length; i++) {
        if (Math.pow(2, j) == (i + 1)) {
            encoded[i] = '0';
            j++;
        } else {
            encoded[i] = data.charAt(i - j);
        }
    }

    for (int i = 0; i < parityBits; i++) {
        int parity = 0;
        for (int k = 0; k < encoded.length; k++) {
            if (((k + 1) & (1 << i)) != 0) {
                parity ^= encoded[k] - '0';
            }
        }
        encoded[(1 << i) - 1] = (char) (parity + '0');
    }

    return new String(encoded);
}

public static String decode(String encoded) {
    int length = encoded.length();
    int parityBits = 0;
```

```
while (Math.pow(2, parityBits) < length) {
   parityBits++;
}

int errorPosition = 0;
for (int i = 0; i < parityBits; i++) {
   int parity = 0;
   for (int k = 0; k < length; k++) {
      if (((k + 1) & (1 << i)) != 0) {
         parity ^= encoded.charAt(k) - '0';
      }
   }
   if (parity != 0) {
      errorPosition += 1 << i;
   }
}

if (errorPosition != 0) {
   char[] corrected = encoded.toCharArray();
   corrected[errorPosition - 1] = (encoded.charAt(errorPosition - 1) == '0') ? '1' : '0';
   encoded = new String(corrected);
   System.out.println("Error detected at position: " + errorPosition);
} else {
   System.out.println("No error detected.");
}

StringBuilder data = new StringBuilder();
for (int i = 0; i < length; i++) {
```

```
        if (Math.pow(2, parityBits - 1) != (i + 1)) {
            data.append(encoded.charAt(i));
        } else {
            parityBits--;
        }
    }

    return data.toString();
}

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    System.out.print("Enter data to encode (binary): ");
    String data = scanner.nextLine();

    String encoded = encode(data);
    System.out.println("Encoded Data: " + encoded);

    System.out.print("Introduce an error (enter position to flip a bit or 0 for no error): ");
    int errorPosition = scanner.nextInt();
    scanner.close();

    char[] erroneousData = encoded.toCharArray();
    if (errorPosition > 0 && errorPosition <= encoded.length()) {
        erroneousData[errorPosition - 1] = (encoded.charAt(errorPosition - 1) == '0') ? '1' :
'0';
    }
```

```
        String erroneousDataStr = new String(erroneousData);
        System.out.println("Erroneous Data: " + erroneousDataStr);

        String decoded = decode(erroneousDataStr);
        System.out.println("Decoded Data: " + decoded);
    }
}
```

Output-

```
Enter data to encode (binary): 10010111
Encoded Data: 101000110111
Introduce an error (enter position to flip a bit or 0 for no error): 4
Erroneous Data: 101100110111
Error detected at position: 4
Decoded Data: 10100010111

=== Code Execution Successful ===
```

```
Enter data to encode (binary): 11111111
Encoded Data: 111011101111
Introduce an error (enter position to flip a bit or 0 for no error): 0
Erroneous Data: 111011101111
No error detected.
Decoded Data: 11101111111

=== Code Execution Successful ===
```

3. Checksum

```
import java.util.Scanner;

public class checksum {

    public static int calculateChecksum(String data) {
        int checksum = 0;
        for (char ch : data.toCharArray()) {
            checksum += ch;
        }
        return checksum;
    }
```

```
    public static boolean verifyChecksum(String originalData, String receivedData)
{
        int originalChecksum = calculateChecksum(originalData);
        int receivedChecksum = calculateChecksum(receivedData);
        return originalChecksum == receivedChecksum;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the string: ");
        String originalData = scanner.nextLine();

        int checksum = calculateChecksum(originalData);
        System.out.println("The checksum generated is = " + checksum);

        System.out.print("Enter the data to be sent: ");
        String receivedData = scanner.nextLine();

        if (verifyChecksum(originalData, receivedData)) {
            System.out.println("Data is received without error.");
        } else {
            System.out.println("There is an error in the received data.");
        }

        scanner.close();
    }
}
```

Output-

```
Enter the string: monogatari
The checksum generated is = 1073
Enter the data to be sent: bakemonogatari
There is an error in the received data.
```

```
Enter the string: jjk
The checksum generated is = 319
Enter the data to be sent: jjk
Data is received without error.
```