: **Implement on a data set of characters the CRC**

import java.util.\*;

// create CRCExample class to demonstrate the working of Cyclic Rnedundancy Check

class CRCExample {

// main() method start

public static void main(String args[]) {

// create scanner class object to take input from user

Scanner scan = new Scanner(System.in);

// declare n for the size of the data

int size;

// take the size of the data from the user

System.out.println("Enter the size of the data array: ");

size = scan.nextInt();

// declaration of the data array

int data[] = new int[size];

// take bits of the data from the user

System.out.println("Enter data bits in the array one by one: ");

for(int i = 0 ; i < size ; i++) {

System.out.println("Enter bit " + (size-i) + ":");

data[i] = scan.nextInt();

}

// take the size of the divisor from the user

System.out.println("Enter the size of the divisor array:");

size = scan.nextInt();

// declaration of the divisor array

int divisor[] = new int[size];

System.out.println("Enter divisor bits in the array one by one: ");

for(int i = 0 ; i < size ; i++) {

System.out.println("Enter bit " + (size-i) + ":");

divisor[i] = scan.nextInt();

}

// Divide the input data by the input divisor and store the result in the rem array

int rem[] = divideDataWithDivisor(data, divisor);

// iterate rem using for loop to print each bit

for(int i = 0; i < rem.length-1; i++) {

System.out.print(rem[i]);

}

System.out.println("\nGenerated CRC code is: ");

for(int i = 0; i < data.length; i++) {

System.out.print(data[i]);

}

for(int i = 0; i < rem.length-1; i++) {

System.out.print(rem[i]);

}

System.out.println();

// we create a new array that contains the original data with its CRC code

int sentData[] = new int[data.length + rem.length - 1];

System.out.println("Enter bits in the array which you want to send: ");

for(int i = 0; i < sentData.length; i++) {

System.out.println("Enter bit " +(sentData.length - 1)+ ":");

sentData[i] = scan.nextInt();

}

receiveData(sentData, divisor);

}

// create divideDataWithDivisor() method to get CRC

static int[] divideDataWithDivisor(int oldData[], int divisor[]) {

// declare rem[] array

int rem[] = new int[divisor.length];

int i;

int data[] = new int[oldData.length + divisor.length];

// use system's arraycopy() method for copying data into rem and data arrays

System.arraycopy(oldData, 0, data, 0, oldData.length);

System.arraycopy(data, 0, rem, 0, divisor.length);

// iterate the oldData and exor the bits of the remainder and the divisor

for(i = 0; i < oldData.length; i++) {

System.out.println((i+1) + ".) First data bit is : "+ rem[0]);

System.out.print("Remainder : ");

if(rem[0] == 1) {

// We have to exor the remainder bits with divisor bits

for(int j = 1; j < divisor.length; j++) {

rem[j-1] = exorOperation(rem[j], divisor[j]);

System.out.print(rem[j-1]);

}

}

else {

// We have to exor the remainder bits with 0

for(int j = 1; j < divisor.length; j++) {

rem[j-1] = exorOperation(rem[j], 0);

System.out.print(rem[j-1]);

}

}

// The last bit of the remainder will be taken from the data

// This is the 'carry' taken from the dividend after every step

// of division

rem[divisor.length-1] = data[i+divisor.length];

System.out.println(rem[divisor.length-1]);

}

return rem;

}

// create exorOperation() method to perform exor data

static int exorOperation(int x, int y) {

// This simple function returns the exor of two bits

if(x == y) {

return 0;

}

return 1;

}

// method to print received data

static void receiveData(int data[], int divisor[]) {

int rem[] = divideDataWithDivisor(data, divisor);

// Division is done

for(int i = 0; i < rem.length; i++) {

if(rem[i] != 0) {

// if the remainder is not equal to zero, data is currupted

System.out.println("Currupted data received...");

return;

}

}

System.out.println("Data received without any error.");

}

}

out put :





