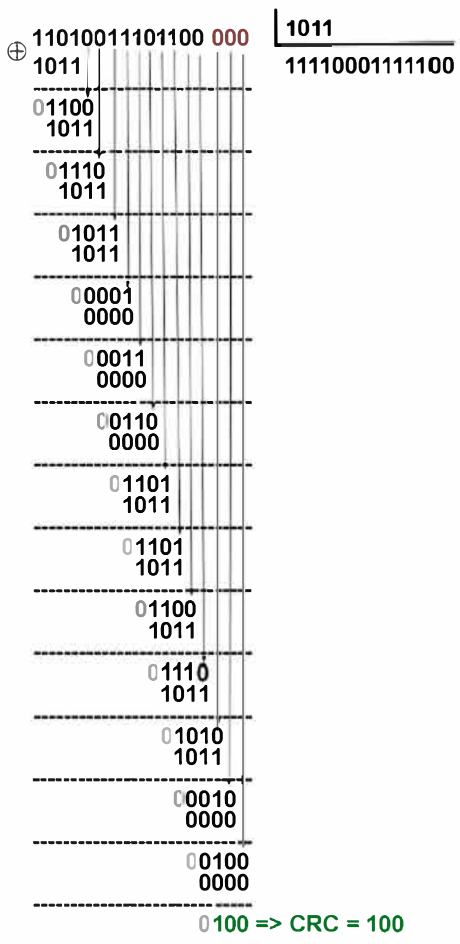
Assignment for VG (väl godkänt)

Cyclic Redundancy Check (CRC) is an error detection algorithm which is used in transmitting data and storing data on storage devices in order to check if data is corrupted or not. In this technique blocks of data are entered to the system and a checksum is calculated and appended to the original data. The checksum is calculated as the remainder of the XOR division of the data by a polynomial. It means that the data is divided by a predefined polynomial and instead of subtraction, XOR is used. The CRC and associated polynomial typically have a name of the form of CRC-N and N is the number of the bits in the remainder of the division which is appended to data and number of bits in the polynomial as the divisor is N + 1 bits. For example, CRC-15 in CAN and CRC-32 in Ethernet communication protocols. To calculate the CRC checksum of a message using CRC-N, first we append N zeros to the message and then we should do the **xor** division of the message with appended zeros by the polynomial as the divisor. The remainder of such a division is the CRC checksum and we need to replace the appended zeros with the calculated CRC checksum. Now if we transmit the result or store it on a storage device, when we receive/read such data we can detect if the data because of any reason has been altered or not. To verify the health of the data we should again do the polynomial division over the checksummed message using the same polynomial and if the remainder is zero, it means that the data is healthy; otherwise, it is corrupted.



```
  11010011101100 000   |1011
⊕ 1011                  11110001111100
  ------------------
  01100
   1011
  ------------------
   01110
    1011
  ------------------
    01011
     1011
  ------------------
     00001
      0000
  ------------------
      00011
       0000
  ------------------
       00110
        0000
  ------------------
        01101
         1011
  ------------------
         01101
          1011
  ------------------
          01100
           1011
  ------------------
           01110
            1011
  ------------------
            01010
             1011
  ------------------
             00010
              0000
  ------------------
              00100
               0000
  ------------------
               0100 => CRC = 100
```

An example of CRC-3:

- Message: 11010011101100
- Polynomial: 1011
- Checksummed message: 11010011101100**100**

Make a program to checksum a message and verify a checksummed message using CRC according to the requirements below.

Requirements

1. The message is an **N**-element array of type **uint8_t**.
2. The minimum length of the message is **1** byte.
3. The maximum length of the message is **14** bytes.
4. The CRC-15 polynomial is **0xC599** (1100010110011001)
5. The message is processed from the **first** element to the **last** element in the array
6. Each element of the message is processed from the **LSB** to the **MSB**
7. No magic numbers!

Some useful links:

1. Cyclic redundancy check
2. Cyclic Redundancy Check
3. How do CRCs work?