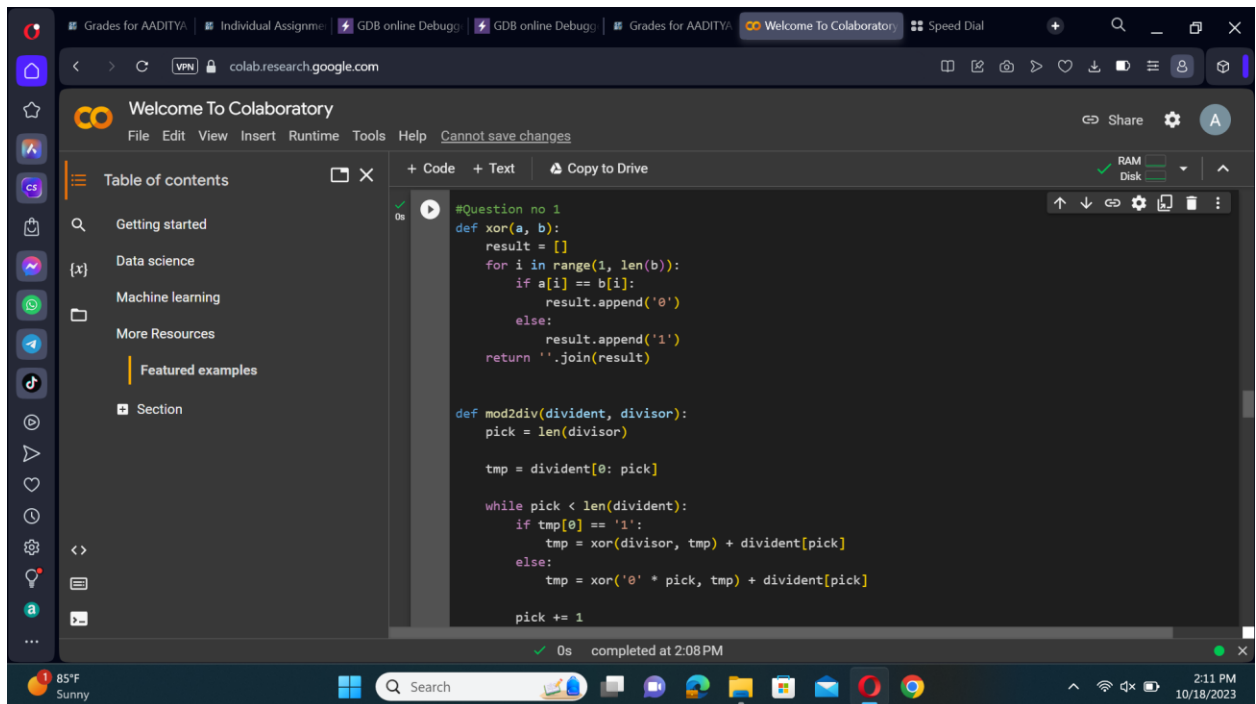


1. Cyclic Redundancy Check (CRC) is one of the popular coding and decoding techniques in the data transmitted over the network for error detection and correction. Given  $x^5 + x^2 + 1$  as a CRC generation polynomial from International Telegraph and Telephone Consultative Committee (CCITT), write the encoding and decoding def functions in Python for the only 4-bits original binary data. The examples and testcases of the encoding and decoding processes are shown as follows for your programming. After that, discuss how many bits errors CRC can detect



The screenshot shows a Google Colaboratory notebook interface. The left sidebar contains a 'Table of contents' with links to 'Getting started', 'Data science', 'Machine learning', 'More Resources', and 'Featured examples'. The main code editor displays the following Python code:

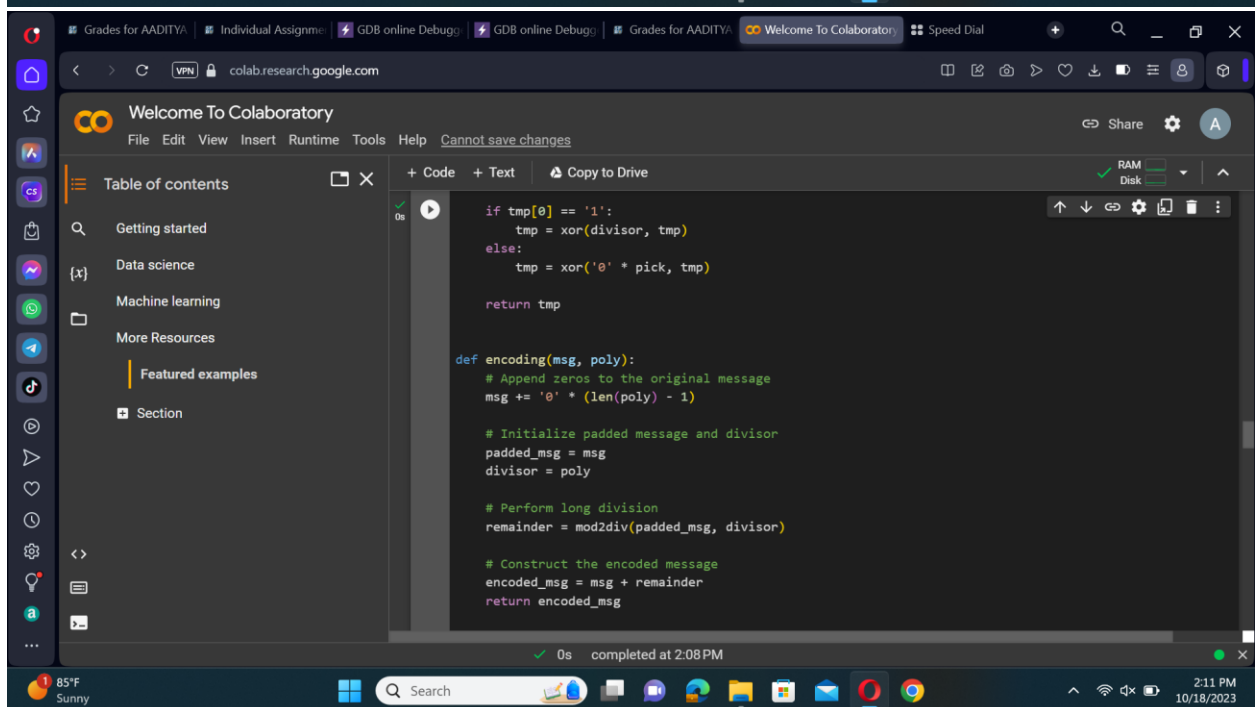
```
#Question no 1
def xor(a, b):
    result = []
    for i in range(1, len(b)):
        if a[i] == b[i]:
            result.append('0')
        else:
            result.append('1')
    return ''.join(result)

def mod2div(divident, divisor):
    pick = len(divisor)

    tmp = divident[0: pick]

    while pick < len(divident):
        if tmp[0] == '1':
            tmp = xor(divisor, tmp) + divident[pick]
        else:
            tmp = xor('0' * pick, tmp) + divident[pick]
        pick += 1
```

The status bar at the bottom indicates '0s completed at 2:08 PM'.



The screenshot shows the same Google Colaboratory notebook interface, displaying the continuation of the Python code for CRC encoding:

```
        tmp = xor(divisor, tmp)
    else:
        tmp = xor('0' * pick, tmp)

    return tmp

def encoding(msg, poly):
    # Append zeros to the original message
    msg += '0' * (len(poly) - 1)

    # Initialize padded message and divisor
    padded_msg = msg
    divisor = poly

    # Perform long division
    remainder = mod2div(padded_msg, divisor)

    # Construct the encoded message
    encoded_msg = msg + remainder
    return encoded_msg
```

The status bar at the bottom indicates '0s completed at 2:08 PM'.

Grades for AADITYA | Individual Assignment | GDB online Debug | GDB online Debug | Grades for AADITYA | Welcome To Colaboratory | Speed Dial

colab.research.google.com

### Welcome To Colaboratory

File Edit View Insert Runtime Tools Help [Cannot save changes](#)

Table of contents

- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- Section

```
def decoding(rcv, poly):
    # Perform long division
    remainder = mod2div(rcv, poly)

    # Check if the remainder is zero
    if int(remainder) == 0:
        return 'No error'
    else:
        return 'Error'

# Test cases for encoding
org_sig1 = '1010'
poly = '100101'
print(encoding(org_sig1, poly))

org_sig2 = '1100'
poly = '100101'
print(encoding(org_sig2, poly))

# Test cases for decoding
received_sig1 = '101000111'
poly = '100101'
```

0s completed at 2:08 PM

85°F Sunny 2:11 PM 10/18/2023

Grades for AADITYA | Individual Assignment | GDB online Debug | GDB online Debug | Grades for AADITYA | Welcome To Colaboratory | Speed Dial

colab.research.google.com

### Welcome To Colaboratory

File Edit View Insert Runtime Tools Help [Cannot save changes](#)

Table of contents

- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- Section

```
# Test cases for decoding
received_sig1 = '101000111'
poly = '100101'
print(decoding(received_sig1, poly)) # Output: No error

received_sig2 = '101001111'
poly = '100101'
print(decoding(received_sig2, poly))

received_sig3 = '110011001'
poly = '100101'
print(decoding(received_sig3, poly)) # Output: No error

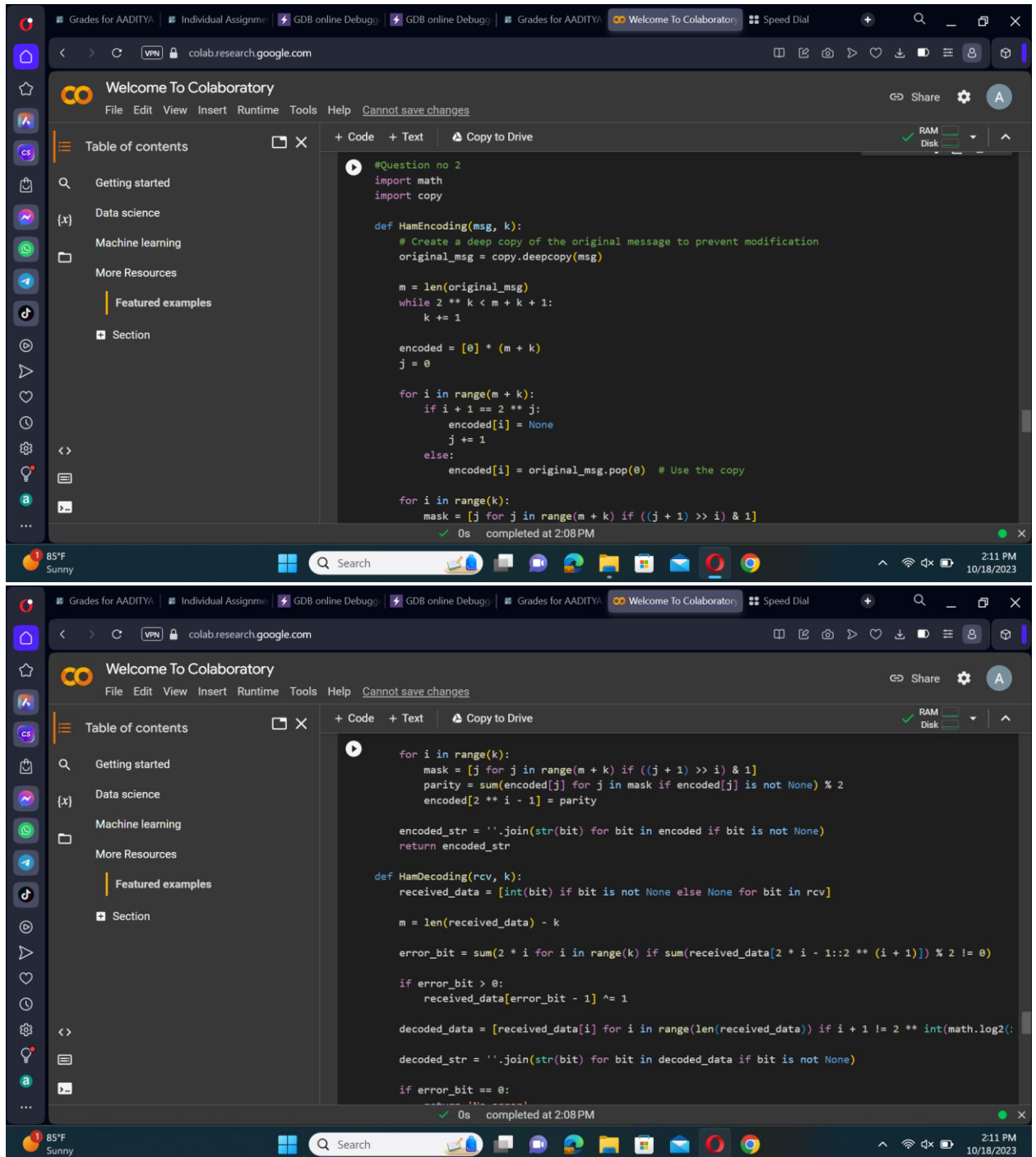
received_sig4 = '110011111'
poly = '100101'
print(decoding(received_sig4, poly)) # Output: Error
```

```
10100000000111
11000000011001
No error
Error
No error
Error
```

0s completed at 2:08 PM

85°F Sunny 2:11 PM 10/18/2023

2. Hamming code is one important error correcting code in computer science and telecommunication as well. Standard Hamming code can only detect and correct a single bit error. Below is my solution for the problem



The image displays two screenshots of a Google Colaboratory notebook. The top screenshot shows the 'HamEncoding' function, which takes a message 'msg' and a parameter 'k'. It creates a deep copy of the message, calculates its length 'm', and then iterates through the message to encode it. The bottom screenshot shows the 'HamDecoding' function, which takes received data 'rcv' and parameter 'k'. It calculates the parity for each bit and then decodes the message back to its original form. Both functions use a mask to identify and correct errors.

```
#Question no 2
import math
import copy

def HamEncoding(msg, k):
    # Create a deep copy of the original message to prevent modification
    original_msg = copy.deepcopy(msg)

    m = len(original_msg)
    while 2 ** k < m + k + 1:
        k += 1

    encoded = [0] * (m + k)
    j = 0

    for i in range(m + k):
        if i + 1 == 2 ** j:
            encoded[i] = None
            j += 1
        else:
            encoded[i] = original_msg.pop(0) # Use the copy

    for i in range(k):
        mask = [j for j in range(m + k) if ((j + 1) >> i) & 1]
        parity = sum(encoded[j] for j in mask if encoded[j] is not None) % 2
        encoded[2 ** i - 1] = parity

    encoded_str = ''.join(str(bit) for bit in encoded if bit is not None)
    return encoded_str

def HamDecoding(rcv, k):
    received_data = [int(bit) if bit is not None else None for bit in rcv]

    m = len(received_data) - k

    error_bit = sum(2 * i for i in range(k) if sum(received_data[2 * i - 1::2 * (i + 1)]) % 2 != 0)

    if error_bit > 0:
        received_data[error_bit - 1] ^= 1

    decoded_data = [received_data[i] for i in range(len(received_data)) if i + 1 != 2 ** int(math.log2(i + 1))]

    decoded_str = ''.join(str(bit) for bit in decoded_data if bit is not None)

    if error_bit == 0:
        return decoded_str
    else:
        return decoded_str
```

Grades for AADITYA | Individual Assignment | GDB online Debug | GDB online Debug | Grades for AADITYA | Welcome To Colaboratory | Speed Dial

colab.research.google.com

# Welcome To Colaboratory

File Edit View Insert Runtime Tools Help Cannot save changes

Table of contents

- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- Section

```
return 'No error'
else:
    return f'Error at Position {error_bit}, and correct data: {decoded_str}'

# Test cases
org_sig1 = [1, 1, 0, 1]
k1 = 3
encoded_sig1 = HamEncoding(org_sig1, k1)
print(f"Original Data: {org_sig1}, Encoded Data: {encoded_sig1}")

received_sig1 = list(encoded_sig1)
k1 = 3
result1 = HamDecoding(received_sig1, k1)
print(result1)

org_sig2 = [1, 0, 0, 1, 0, 1, 1]
k2 = 4
encoded_sig2 = HamEncoding(org_sig2, k2)
print(f"Original Data: {org_sig2}, Encoded Data: {encoded_sig2}")

received_sig2 = list(encoded_sig2)
k2 = 4
result2 = HamDecoding(received_sig2, k2)
print(result2)
```

0s completed at 2:08 PM

85°F Sunny 2:11 PM 10/18/2023

Grades for AADITYA | Individual Assignment | GDB online Debug | GDB online Debug | Grades for AADITYA | Welcome To Colaboratory | Speed Dial

colab.research.google.com

# Welcome To Colaboratory

File Edit View Insert Runtime Tools Help Cannot save changes

Table of contents

- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- Section

```
result2 = HamDecoding(received_sig2, k2)
print(result2)

org_sig3 = [1, 0, 1, 0, 1]
k3 = 3
encoded_sig3 = HamEncoding(org_sig3, k3)
print(f"Original Data: {org_sig3}, Encoded Data: {encoded_sig3}")

received_sig3 = list(encoded_sig3)
k3 = 3
result3 = HamDecoding(received_sig3, k3)
print(result3)

org_sig4 = [0, 1, 0, 1, 1, 0]
k4 = 4
encoded_sig4 = HamEncoding(org_sig4, k4)
print(f"Original Data: {org_sig4}, Encoded Data: {encoded_sig4}")

received_sig4 = list(encoded_sig4)
k4 = 4
result4 = HamDecoding(received_sig4, k4)
print(result4)
```

Original Data: [1, 1, 0, 1], Encoded Data: 1010101

0s completed at 2:08 PM

85°F Sunny 2:11 PM 10/18/2023

Grades for AADITYAIndividual AssignmentGDB online DebugGDB online DebugGrades for AADITYAWelcome To ColaboratorySpeed Dial

colab.research.google.com

Welcome To Colaboratory

File Edit View Insert Runtime Tools Help Cannot save changes

Share

Table of contents

Getting started

Data science

Machine learning

More Resources

Featured examples

Section

+ Code + Text Copy to Drive

```
org_sig4 = [0, 1, 0, 1, 1, 0]
k4 = 4
encoded_sig4 = HamEncoding(org_sig4, k4)
print(f"Original Data: {org_sig4}, Encoded Data: {encoded_sig4}")

received_sig4 = list(encoded_sig4)
k4 = 4
result4 = HamDecoding(received_sig4, k4)
print(result4)
```

Original Data: [1, 1, 0, 1], Encoded Data: 1010101

No error

Original Data: [1, 0, 0, 1, 0, 1, 1], Encoded Data: 10110010011

Error at Position 6, and correct data: 10110110011

Original Data: [1, 0, 1, 0, 1], Encoded Data: 001101011

Error at Position 6, and correct data: 001100011

Original Data: [0, 1, 0, 1, 1, 0], Encoded Data: 1100101110

Error at Position 2, and correct data: 1000101110

0s completed at 2:08 PM

85°F Sunny

Search

2:11 PM 10/18/2023