Network and Communication Lab Exercise-2

Error Check Methods

Joyeeta Dey

19BCE1243

Dr. Karmel A.

Q1. Assume Sender A is transmitting a packet made of four 16-bit words (A7A2)16, (CABF)16, (903A)16, (A123)16 to Receiver B. Apply the appropriate error detection mechanism that involves binary addition to compute the codewords at the receiver end.

Code (in Python):

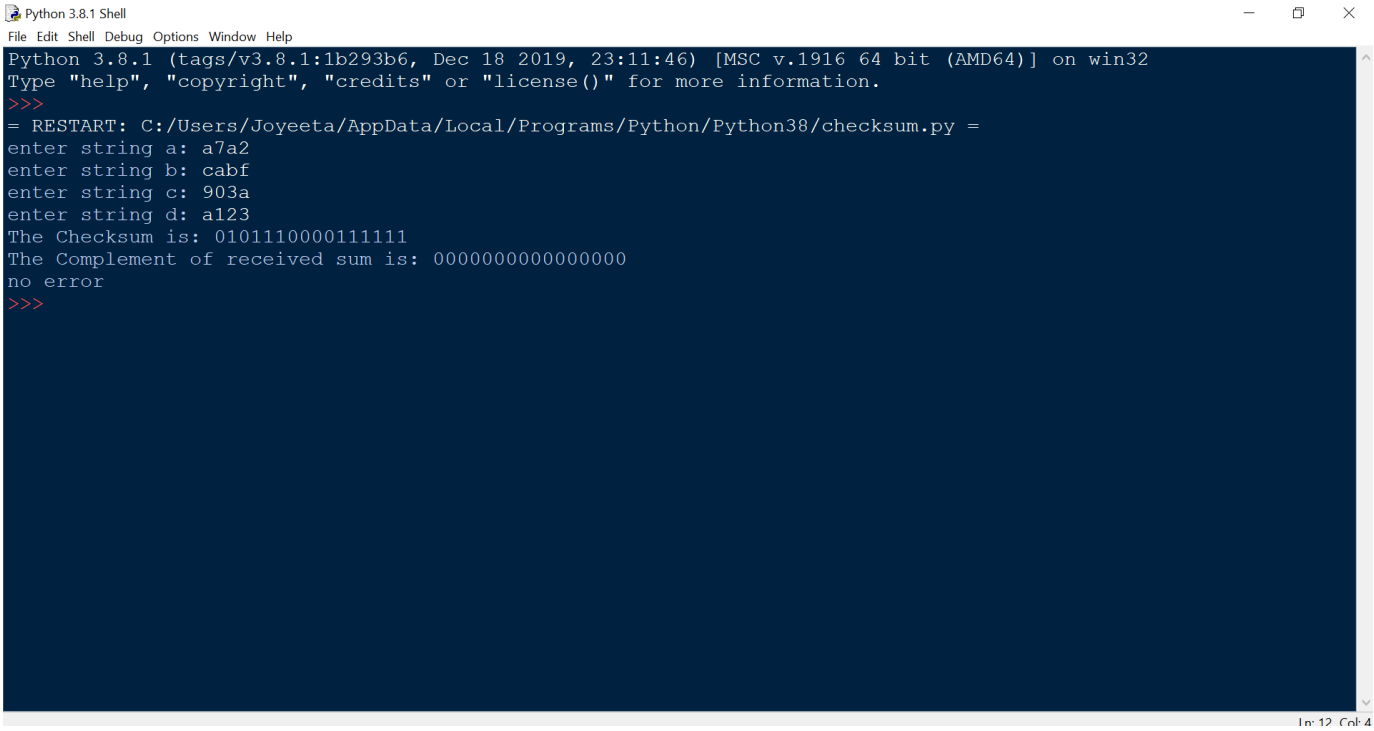
```
def findcarry(a):
    carry=[]
    for i in range(2):
        carry.append(a[i])
    empty_str=""
    carry1=empty_str.join(carry)
    return carry1


def without_carry(y):
    z=[]
    for i in range(2,18):
        z.append(y[i])
    empty_str=""
```

```
    z1=empty_str.join(z)
    return z1

def checker(a):
    for i in range(len(a)):
        if(a[i]=='1'):
            flag=1
            break
        else:
            flag=0
    if(flag==1):
        print("error")
    else:
        print("no error")

def listToString(x):
    empty_str=""
    s=empty_str.join(x)
    return s

def finalbin(n):
    return n.replace("0b", "")

def binary_sum(num1,num2):
```

```
    binsum = bin(int(num1,2) + int(num2,2))
    return binsum

def findComplement(x):
    complement=[]
    for i in range(0,len(x)):
        if(x[i]=='1'):
            complement.append('0')
        else:
            complement.append('1')
    empty=""
    complt=empty.join(complement)
    return complt

a=str(input("enter string a: "))
b=str(input("enter string b: "))
c=str(input("enter string c: "))
d=str(input("enter string d: "))
a1 = int(a, 16)
b1 = int(b, 16)
c1=int(c, 16)
d1=int(d, 16)
code=a1+b1+c1+d1
```

```
sum=bin(code)
raw_sum=finalbin(sum)

carry2=findcarry(raw_sum)

sum2=without_carry(raw_sum)

sum3=binary_sum(sum2,carry2)

sum4=finalbin(sum3)
chcksum=findComplement(sum4)

print("The Checksum is:",chcksum)

receiversum=binary_sum(sum4, chcksum)
rsum=finalbin(receiversum)
rsum_comp=findComplement(rsum)
print("The Complement of received sum is:", rsum_comp)
checker(rsum_comp)
```

Output:



Q2. Implement the appropriate error detection mechanism to compute the codeword at the sender side for the given dataword, 101001111 using the polynomial generator, x4+x2+x+1.

Code (in Python):

```
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

    if tmp[0] == '1':
        tmp = xor(divisor, tmp)
    else:
        tmp = xor('0'*pick, tmp)

    checkword = tmp
    return checkword

def encodeData(data, key):
```

```
    l_key = len(key)

    appended_data = data + '0'*(l_key-1)
    remainder = mod2div(appended_data, key)
    codeword = data + remainder
    return codeword

def checker(a):
    for i in range(len(a)):
        if(a[i]=='1'):
            flag=1
            break
        else:
            flag=0
    if(flag==1):
        print("error")
    else:
        print("no error")


data =str(input("enter data to be sent: "))
key = str(input("enter key: "))
e=encodeData(data, key)
rem=mod2div(e, key)
```

```
print ("reminder is: ",rem)

checker(rem)
```

Output:

```
Python 3.8.1 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/Joyeeta/AppData/Local/Programs/Python/Python38/crc.py ===
enter data to be sent: 101001111
enter key: 10111
reminder is:  0000
no error
>>>
```