PRACTICAL - 6 : HAMMING CODE

Write a program to implement error detection and correction using Hamming code concept. Howe a test run to input data stream and varity error correction feature.

Hamming cook is a set of error-correction code that ext can be used to dotect and correct the error that can occur when the date is transmitted from the sender to receiver. It is a technique developed by R.N Hamming for error correction.

CODE:

det string-to-binary (input-string):

return ".join (format (ord cc), '086') for c in input-string)

dof binary-to-string (binary-data):

chars = []

for i in range (0, len (binary data), e)

byte = binary-data [i:i+e]

chars. append (chr (int (byte, 2)))

def calculate - parity-bite (dato):

return ''. join (chans)

n = Lon (data)

T=0

while (2** 7) < (n+ (++++)):

T+=1

rotion or

```
def incort-parity-bito (data, r)
                  n = len (data)
                 j=0 tr-id about prisoned and due arthoughts
                               and a continue of
                  m = n+r man to man to many and to
                  hamming-code = 67
                 for i in range (1, m+1):
           homming - codi - appoind (D)
                 prior (" serected each: 1"; into the prior " sting
                   hamming - code. appoind cint (date [x])
trom
                   k+=1 (aba-gramma (1927) spent (6) - 1 (1)
oped
                  return 11. join (map (str, hamming-code))
                   the spinner primer of many told started the
               day calculate-parity-value chamming-code (T):
                   hamming-code = list (map (int, hamming-code))
                   n= lin (hamming-code) (1) 11)
                   for i in range (r):
                    parity-pos = 2** in the box a- old
                    parity-vol = 0
                     for j in range (1, n+1):
                      if i & partity pos and it partity pos:
                         parity - val ~ = hamming - coole [j-1]
                     hamming-code [ parity-pos-1)] = parity-vol
                   return 1 join (mop ( etr, harming-code)?
                dog detect - and - correct correct chamming - code, or?:
                    hamming-code = list (map cint, hamming-code))
                    n = lan chamming - code
                  omor-position = 0
                     pri in range on:
                   parity- pos = 2 ** i
                      parity-val =0
```

```
for j in range (1, n+1):
      it is parity-pos:
         parity-val ~= horming-code [j-1]
     if parity-vol 1=0
         error - position += parity-pos
 it comor-position:
     print ( " error at := ? error-position?")
     print ( "Binary pos: { bin (error- Position)[2:], 2till (1)2")
    hamming - cool [ ernor - position - 7 1 =1
    print ("corrected coole: &". join (map ( str., herming - code 1) 3").
 print ("No arror")
return ". sign (map (str, hamming-cools))
det extract-date- from-hamming chamming-cool !
    data = []
    for i in range (11 Len ( hamming - code ? +1):
        it i!= 2 ** j
          date-appoind Chamming-code [ i-1]
        0100
       j+=1
    return 11- join ( map ( str. doto))
det main:
   input - string = input ("enter the string :")
   binary-dato = string - to - binary cinput- string
   print (" Binory : { binory - datoz")
   r= calculate - parity-bite (binary-date)
   hamming-code = incert - parity - bits ( binary - data, r)
   hamming-code = calculate - posity- value chamming-code;
   print (+ " Homming code: & homming-code 3")
  redundant bito: 32 ** i for i in ronge croz
```

print ("In Flip a bit for orror ...") Endict:3 Binory pos:00011

error-bit = int (input (f " Fil Flip Bit (1- 91an chamming-wal)?")) if error bit in redundant - bits: print ("Redundant bit choose another position"). alse: 300000 or star on man 200000 homming- code : homming-code [error. bit-17 + Ci' if hamming-code (orror _ bit -1) = '0'. alse 'o') + homming - vod (7. Ornor Lit] prints ('y"Homming code with error: & hamming code y") hamming-code = detect-and-correct-error (hamming-code, +) corrected-data = extract-date = form-horming (horming-code, r) corrected-string: binary- to-string (corrected - binary date) print ('t' Final output: 'f connected - etring 3'") it -- name -- = " -- main -= " -- m (avail a main (), OUTPUT: Entor the string: Hi Binary : 0100100001101001 Homming code: 01001001100001101001 Flip a bit for ormor... Flip bit (1-21): 2 pedundent bit 2 chaose another option Fup bit for ornor !... Flip bit (1-21):3 Surar ! Hamming code with error: 0110100110000 1100100 corrected cook: 010010011000011001001 RESULT in thus program for Hamming code for implementing orner

detection and correction is successfully executed and output is varified

while True:

かいしてつないう

g- wode n y").

- adi in

alcolly