**CRC CODE**

**Program:**

#include<stdio.h>

#include<string.h>

int main() {

int i,j,keylen,msglen;

char input[100], key[30],temp[30],quot[100],rem[30],key1[30];

//clrscr();

printf("Enter Data: ");

gets(input);

printf("Enter Key: ");

gets(key);

keylen=strlen(key);

msglen=strlen(input);

strcpy(key1,key);

for (i=0;i<keylen-1;i++) {

input[msglen+i]='0';

}

for (i=0;i<keylen;i++)

temp[i]=input[i];

for (i=0;i<msglen;i++) {

quot[i]=temp[0];

if(quot[i]=='0')

for (j=0;j<keylen;j++)

key[j]='0'; else

for (j=0;j<keylen;j++)

key[j]=key1[j];

for (j=keylen-1;j>0;j--) {

if(temp[j]==key[j])

rem[j-1]='0'; else

rem[j-1]='1';

}

rem[keylen-1]=input[i+keylen];

strcpy(temp,rem);

}

strcpy(rem,temp);

printf("\nQuotient is ");

for (i=0;i<msglen;i++)

printf("%c",quot[i]);

printf("\nRemainder is ");

for (i=0;i<keylen-1;i++)

printf("%c",rem[i]);

printf("\nFinal data is: ");

for (i=0;i<msglen;i++)

printf("%c",input[i]);

for (i=0;i<keylen-1;i++)

printf("%c",rem[i]);

//if (rem ==0)

//printf("Error in the code");

//else

//printf("\nNo Error detected%d",rem);

//getch();

return 0;

}

**OUTPUT:**

root@ACA8FE61:~# cd Downloads

root@ACA8FE61:~/Downloads# **gcc -o crc crc.c**

crc.c: In function ‘main’:

crc.c:8:2: warning: implicit declaration of function ‘gets’ [-Wimplicit-function-declaration]

gets(input);

^

/tmp/cc85BTst.o: In function `main':

crc.c:(.text+0x27): warning: the `gets' function is dangerous and should not be used.

root@ACA8FE61:~/Downloads# **./crc**

Enter Data: 100100

Enter Key: 1101

Quotient is 111101

Remainder is 001

Final data is: 100100001root@ACA8FE61:~/Downloads# **./crc**

Enter Data: 100100001

Enter Key: 1101

Quotient is 111101000

Remainder is 000

Final data is: 100100001000root@ACA8FE61:~/Downloads#

**Hamming code**

**Program:**

#include<stdio.h>

//#include <curses.h>

//#include<conio.h>

int main() {

int data[10];

int dataatrec[10],c,c1,c2,c3,i;

//clrscr();

printf("Enter 4 bits of data one by one\n");

scanf("%d",&data[0]);

scanf("%d",&data[1]);

scanf("%d",&data[2]);

scanf("%d",&data[4]);

//Calculation of even parity

data[6]=data[0]^data[2]^data[4];

data[5]=data[0]^data[1]^data[4];

data[3]=data[0]^data[1]^data[2];

printf("\nEncoded data is\n");

for(i=0;i<7;i++)

printf("%d",data[i]);

printf("\n\nEnter received data bits one by one\n");

for(i=0;i<7;i++)

scanf("%d",&dataatrec[i]);

c1=dataatrec[6]^dataatrec[4]^dataatrec[2]^dataatrec[0];

c2=dataatrec[5]^dataatrec[4]^dataatrec[1]^dataatrec[0];

c3=dataatrec[3]^dataatrec[2]^dataatrec[1]^dataatrec[0];

c=c3\*4+c2\*2+c1 ;

if(c==0) {

printf("\nNo error while transmission of data\n");

}

else {

printf("\nError on position %d",c);

printf("\nData sent : ");

for(i=0;i<7;i++)

printf("%d",data[i]);

printf("\nData received : ");

for(i=0;i<7;i++)

printf("%d",dataatrec[i]);

printf("\nCorrect message is\n");

//if errorneous bit is 0 we complement it else vice versa

if(dataatrec[7-c]==0)

dataatrec[7-c]=1;

else

dataatrec[7-c]=0;

for (i=0;i<7;i++)

{

printf("%d ",dataatrec[i]);

}

printf("\n");

return 0;

}

getchar ();

}

**Output:**

root@ACA8FE61:~/Downloads# **gcc -o HAMMING HAMMING.C**

root@ACA8FE61:~/Downloads# **./HAMMING**

Enter 4 bits of data one by one

1111

0000

1111

0000

Encoded data is

1111011110011110

Enter received data bits one by one

1111

0111

1001

1110

0000

1111

111

Error on position 5835

Data sent : 1111011110011110

Data received : 11111111001111001111111

Correct message is

1111 111 1001 1110 0 1111 111 \*/