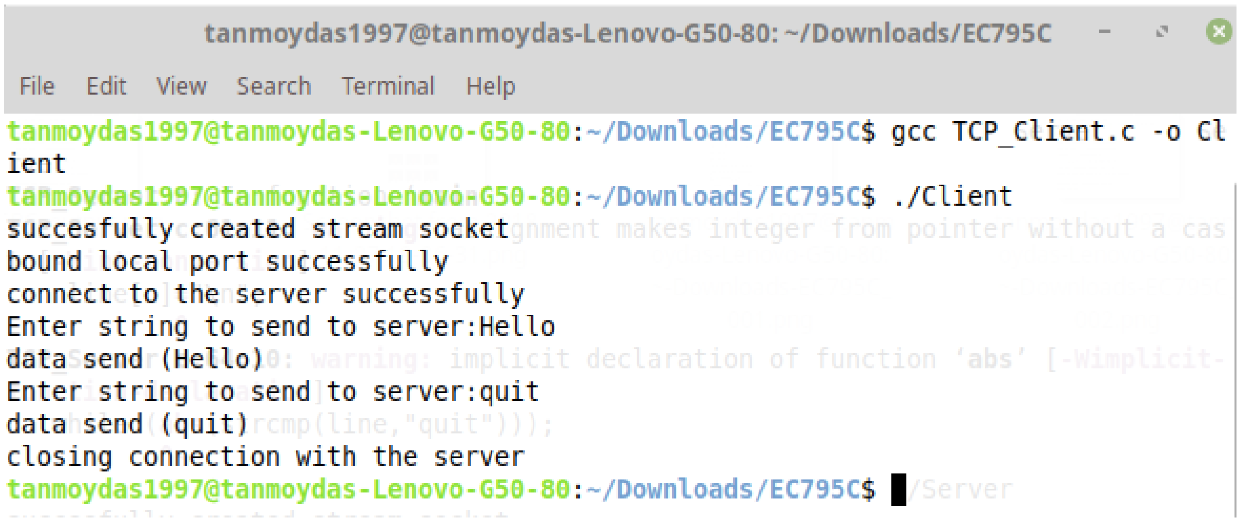
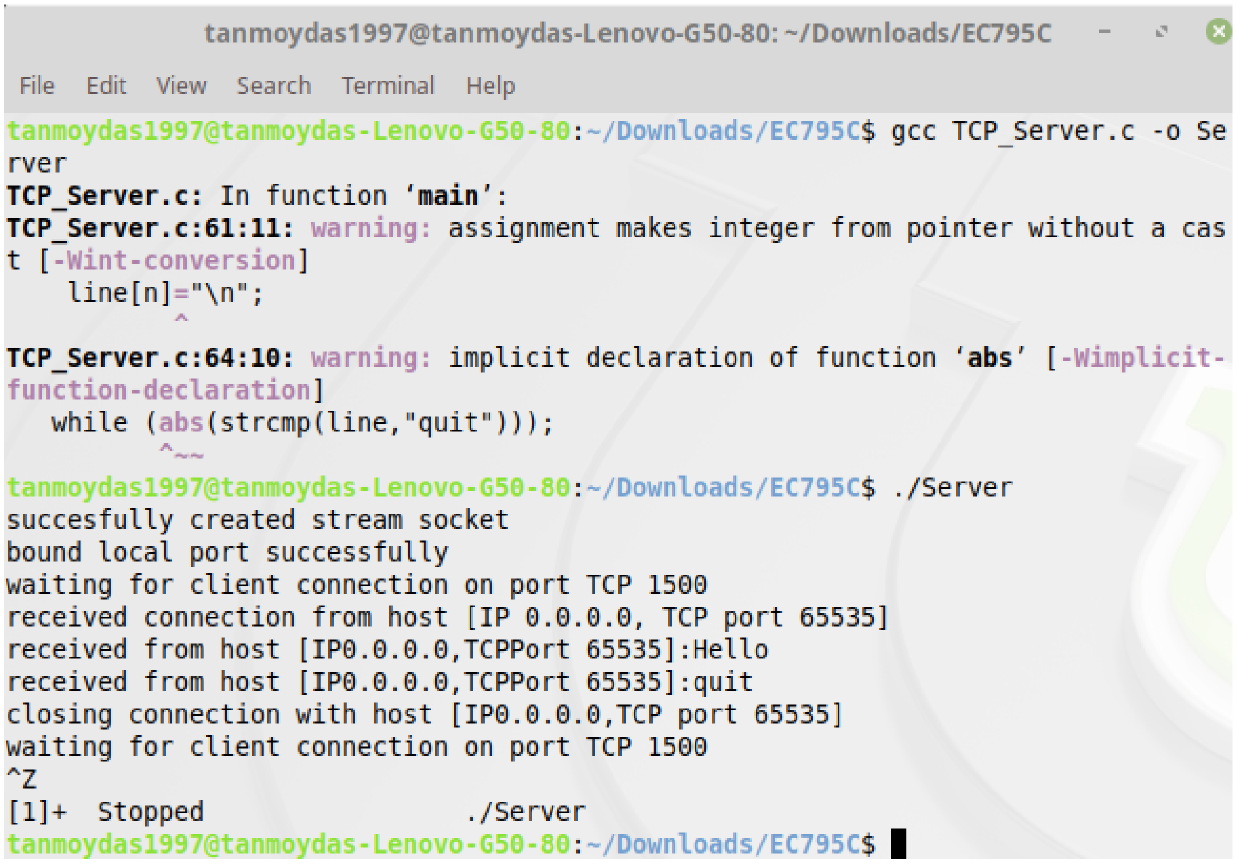
Output:

TCP Client



```
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ gcc TCP_Client.c -o Cl
ient
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ ./Client
succesfully created stream socket
bound local port successfully
connect to the server successfully
Enter string to send to server:Hello
data send (Hello)
Enter string to send to server:quit
data send (quit)
closing connection with the server
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ 
```

TCP Server

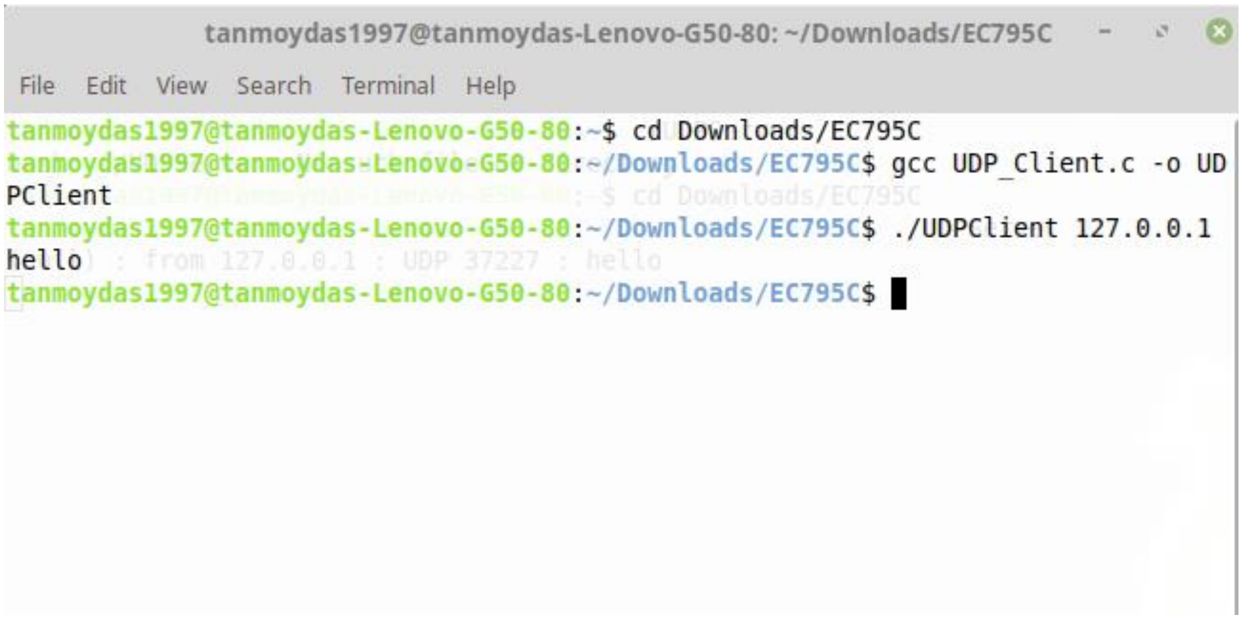


```
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ gcc TCP_Server.c -o Se
rver
TCP_Server.c: In function ‘main’:
TCP_Server.c:61:11: warning: assignment makes integer from pointer without a cas
t [-Wint-conversion]
    line[n]="\n";
           ^

TCP_Server.c:64:10: warning: implicit declaration of function ‘abs’ [-Wimplicit-
function-declaration]
   while (abs(strcmp(line,"quit")));
          ^~~
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ ./Server
succesfully created stream socket
bound local port successfully
waiting for client connection on port TCP 1500
received connection from host [IP 0.0.0.0, TCP port 65535]
received from host [IP0.0.0.0,TCPPort 65535]:Hello
received from host [IP0.0.0.0,TCPPort 65535]:quit
closing connection with host [IP0.0.0.0,TCP port 65535]
waiting for client connection on port TCP 1500
^Z
[1]+  Stopped                 ./Server
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ 
```

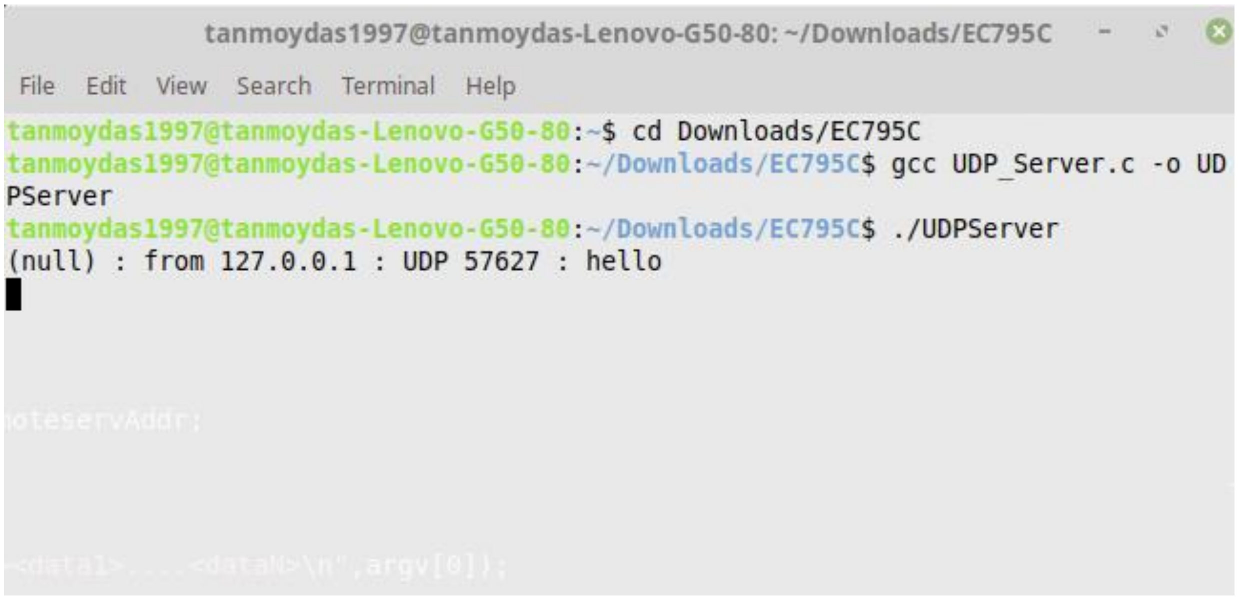
Output:

UDP Client


tanmoydas1997@tanmoydas-Lenovo-G50-80: ~/Downloads/EC795C
File Edit View Search Terminal Help

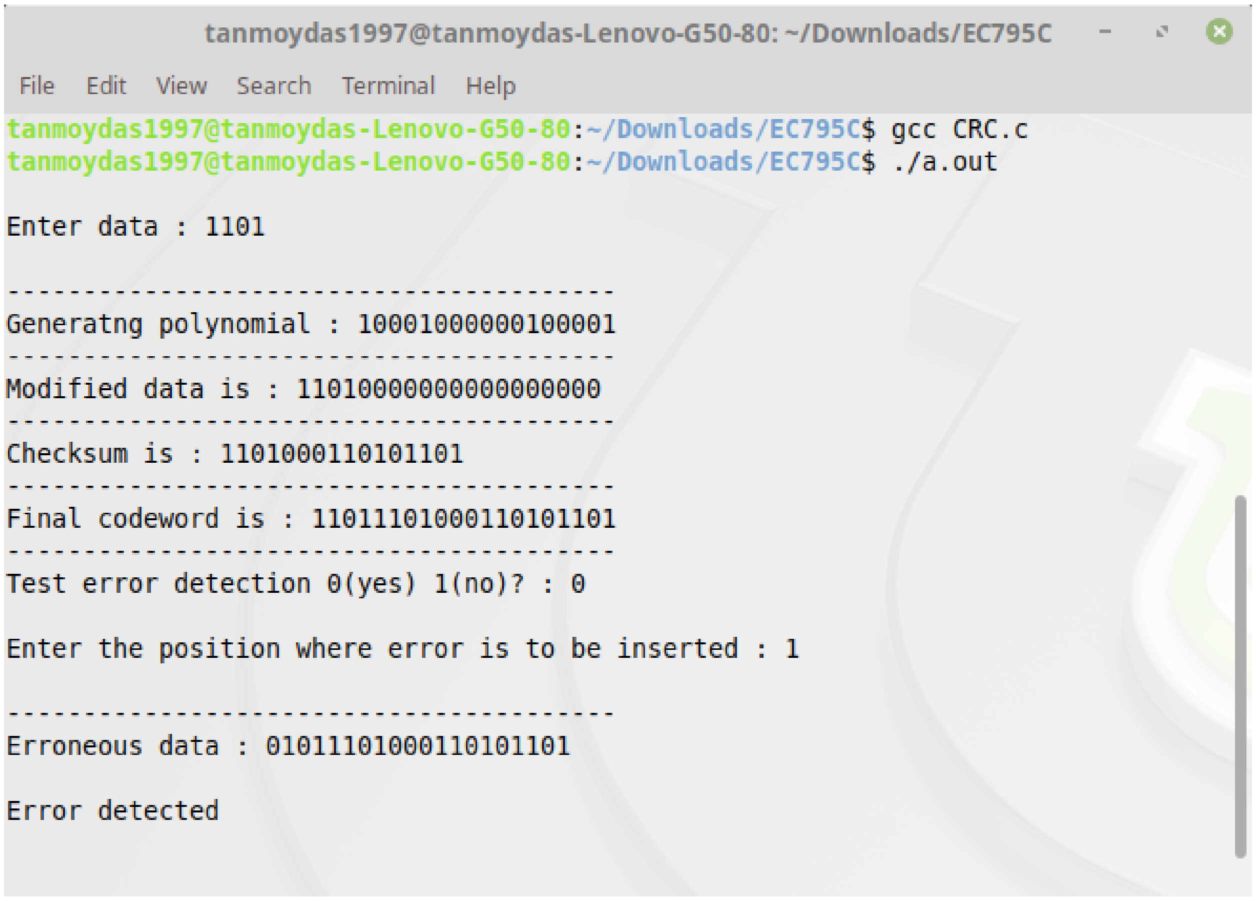
```
tanmoydas1997@tanmoydas-Lenovo-G50-80:~$ cd Downloads/EC795C
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ gcc UDP_Client.c -o UD
PClient
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ ./UDPClient 127.0.0.1
hello
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$
```

UDP Server


tanmoydas1997@tanmoydas-Lenovo-G50-80: ~/Downloads/EC795C
File Edit View Search Terminal Help

```
tanmoydas1997@tanmoydas-Lenovo-G50-80:~$ cd Downloads/EC795C
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ gcc UDP_Server.c -o UD
PServer
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ ./UDPServer
(null) : from 127.0.0.1 : UDP 57627 : hello
```

Output:


tanmoydas1997@tanmoydas-Lenovo-G50-80: ~/Downloads/EC795C
File Edit View Search Terminal Help

```
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ gcc CRC.c
tanmoydas1997@tanmoydas-Lenovo-G50-80:~/Downloads/EC795C$ ./a.out

Enter data : 1101

----------------------------------------
Generatng polynomial : 10001000000100001
----------------------------------------
Modified data is : 11010000000000000000
----------------------------------------
Checksum is : 1101000110101101
----------------------------------------
Final codeword is : 11011101000110101101
----------------------------------------
Test error detection 0(yes) 1(no)? : 0

Enter the position where error is to be inserted : 1

----------------------------------------
Erroneous data : 01011101000110101101

Error detected
```

Program:

TCP Client

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <unistd.h>
#include <math.h>
#include <string.h>
#include <stdio.h>
#define MAX_MSG 100
#define SERVER_ADDR "127.0.0.1"
#define CLIENT_ADDR "127.0.0.1"
#define SERVER_PORT 1500
#define CLIENT_PORT 1500

int main()
{
    int sd, rc, i;
    struct sockaddr_in clientAddr, servAddr;
    char line[MAX_MSG];

    bzero((char *)&servAddr, sizeof(servAddr));
    servAddr.sin_family = AF_INET;
    servAddr.sin_addr.s_addr = inet_addr(SERVER_ADDR);
    servAddr.sin_port = htons(SERVER_PORT);

    bzero((char *)&clientAddr, sizeof(clientAddr));
    clientAddr.sin_family = AF_INET;
    clientAddr.sin_addr.s_addr = INADDR_ANY;
    clientAddr.sin_port = htons(0);

    sd = socket(AF_INET, SOCK_STREAM, 0);
    printf("succesfully created stream socket \n");

    bind(sd, (struct sockaddr *)&clientAddr, sizeof(clientAddr));
    printf("bound local port successfully \n");

    connect(sd, (struct sockaddr *)&servAddr, sizeof(servAddr));
    printf("connect to the server successfully\n");

    do
    {
        printf("Enter string to send to server:");
        scanf("%s", line);
        send(sd, line, strlen(line) + 1, 0);
        printf("data send (%s)\n", line);
    } while (strcmp(line, "quit"));
    printf("closing connection with the server \n");
    close(sd);
}
```

TCP Server

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <unistd.h>
#include <math.h>
#include <string.h>
#include <stdio.h>
#define MAX_MSG 100
#define SERVER_ADDR "127.0.0.1"
#define SERVER_PORT 1500
int main()
{
    int sd, newsd, clilen, n;
    struct sockaddr_in cliAddr, servAddr;
    char line[MAX_MSG];

    bzero((char *)&servAddr, sizeof(servAddr));
    servAddr.sin_family = AF_INET;
    servAddr.sin_addr.s_addr = inet_addr(SERVER_ADDR);
    servAddr.sin_port = htons(SERVER_PORT);

    sd = socket(AF_INET, SOCK_STREAM, 0);
    printf("succesfully created stream socket \n");

    bind(sd, (struct sockaddr *)&servAddr, sizeof(servAddr));
    printf("bound local port successfully \n");

    listen(sd, 5);
    while (1)
    {
        printf("waiting for client connection on port TCP %u\n", SERVER_PORT);
        newsd = accept(sd, (struct sockaddr *)&cliAddr, &clilen);

        printf("received connection from host [IP %s, TCP port %d]\n",
inet_ntoa(cliAddr.sin_addr), ntohs(cliAddr.sin_port));

        do
        {
            memset(line, 0x0, MAX_MSG);
            n = recv(newsd, line, MAX_MSG, 0);
            line[n] = "\n";
            printf("received from host [IP%s,TCPPort %d]:%s\n",
inet_ntoa(cliAddr.sin_addr), ntohs(cliAddr.sin_port), line);
        } while (abs(strcmp(line, "quit")));
        printf("closing connection with host [IP%s,TCP port %d]\n",
inet_ntoa(cliAddr.sin_addr), ntohs(cliAddr.sin_port));

        close(newsd);
    }
}
```

Program:

UDP Client

```
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <unistd.h>
#include <math.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#define MAX_MSG 100
#define REMOTE_SERVER_PORT 1500

int main(int argc, char *argv[])
{
    int sd, rc, i;
    struct sockaddr_in cliAddr, remoteservAddr;
    struct hostent *h;
    if (argc < 3)
    {
        printf("Usage : %s <server><data1>....<dataN>\n", argv[0]);
        exit(1);
    }
    h = gethostbyname(argv[1]);
    if (h == NULL)
    {
        printf("%s : unknown host '%s \n", argv[0], argv[1]);
        exit(1);
    }

    remoteservAddr.sin_family = h->h_addrtype;
    memcpy((char *)&remoteservAddr.sin_addr.s_addr, h->h_addr_list[0], h-
>h_length);
    remoteservAddr.sin_port = htons(REMOTE_SERVER_PORT);
    sd = socket(AF_INET, SOCK_DGRAM, 0);

    if (sd < 0)
    {
        printf("%s : cannot open socket \n", argv[0]);
        exit(1);
    }

    cliAddr.sin_family = AF_INET;
    cliAddr.sin_addr.s_addr = htonl(INADDR_ANY);
    cliAddr.sin_port = htons(0);

    rc = bind(sd, (struct sockaddr *)&cliAddr, sizeof(cliAddr));

    if (rc < 0)
    {
        printf("%s : cannot bind port \n", argv[0]);
        exit(1);
    }

    for (i = 2; i < argc; i++)
    {
        rc = sendto(sd, argv[i], strlen(argv[1]) + 1, 0, (struct
sockaddr*)&remoteservAddr, sizeof(remoteservAddr));
        if (rc < 0)
        {
            printf("%s : cannot send data %d \n", argv[0], i - 1);
            close(sd);
            exit(1);
        }
    }
    return 1;
}
```

UDP Server

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <unistd.h>
#include <math.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#define MAX_MSG 100

int main(int argc, char *argv[])
{
    int sd, rc, n, clilen;
    struct sockaddr_in cliAddr, servAddr;
    char msg[MAX_MSG];
    sd = socket(AF_INET, SOCK_DGRAM, 0);
    if (sd < 0)
    {
        printf("%s : Cannot open socket \n", argv[0]);
        exit(1);
    }

    servAddr.sin_family = AF_INET;
    servAddr.sin_addr.s_addr = ntohl(INADDR_ANY);
    servAddr.sin_port = htons(1500);

    rc = bind(sd, (struct sockaddr *)&servAddr, sizeof(servAddr));

    if (rc < 0)
    {
        printf("Cannot bind \n");
        exit(1);
    }
    while (1)
    {
        memset(msg, 0x0, MAX_MSG);
        clilen = sizeof(cliAddr);
        n = recvfrom(sd, msg, MAX_MSG, 0, (struct sockaddr *)&cliAddr, &clilen);

        if (n < 0)
        {
            printf("%s : Cannot receive data \n", argv[0]);
            continue;
        }
        printf("%s : from %s : UDP %u : %s \n", argv[1],
inet_ntoa(cliAddr.sin_addr), ntohs(cliAddr.sin_port), msg);
    }
    return 0;
}
```

Program:

```c
#include <stdio.h>
#include <string.h>
#define N strlen(g)

char t[28], cs[28], g[] = "10001000000100001";
int a, e, c;

void xor () {
    for (c = 1; c < N; c++)
        cs[c] = ((cs[c] == g[c]) ? '0' : '1');
}

    void crc()
{
    for (e = 0; e < N; e++)
        cs[e] = t[e];
    do
    {
        if (cs[0] == '1')
            xor();
        for (c = 0; c < N - 1; c++)
            cs[c] = cs[c + 1];
        cs[c] = t[e++];
    } while (e <= a + N - 1);
}

int main()
{
    printf("\nEnter data : ");
    scanf("%s", t);
    printf("\n----------------------------------------");
    printf("\nGeneratng polynomial : %s", g);
    a = strlen(t);
    for (e = a; e < a + N - 1; e++)
        t[e] = '0';
    printf("\n----------------------------------------");
    printf("\nModified data is : %s", t);
    printf("\n----------------------------------------");
    crc();
    printf("\nChecksum is : %s", cs);
    for (e = a; e < a + N - 1; e++)
        t[e] = cs[e - a];
    printf("\n----------------------------------------");
    printf("\nFinal codeword is : %s", t);
    printf("\n----------------------------------------");
    printf("\nTest error detection 0(yes) 1(no)? : ");
    scanf("%d", &e);
    if (e == 0)
    {
        do
        {
            printf("\nEnter the position where error is to be inserted : ");
            scanf("%d", &e);
        } while (e == 0 || e > a + N - 1);
        t[e - 1] = (t[e - 1] == '0') ? '1' : '0';
        printf("\n----------------------------------------");
        printf("\nErroneous data : %s\n", t);
    }
    crc();
    for (e = 0; (e < N - 1) && (cs[e] != '1'); e++)
        ;
    if (e < N - 1)
        printf("\nError detected\n\n");
    else
        printf("\nNo error detected\n\n");
    printf("\n----------------------------------------\n");
    return 0;
}
```