Netcom LAB FAT

Name: Kulvir Singh
Reg. No. : 19BCE2074

Question Set = 7

Question 7a)

a)Identify the tool is used with the netstat command to view a system's routing table? Run the identified command and highlight the output of the command with an appropriate result and discussion.

Aim, Algorithm, Results and Discussions

Kulvir Singh
19BCE2094

Question 7

a) **Aim**: Identify the tool which is used to view a system's routing table using the netstat command and display output.

Algorithm:

Use the 'netstat -r' command in the terminal to get the system's routing table.

Results and Discussions:

→ Netstat is a command line utility for Linux that prints the network connections, routing tables, interface statistics, multicast memberships etc.

→ Netstat can be used to diagnose network issues and service problems

→ "Destination" column shows the destination network.

→ "Gateway" column shows the router through which packets are forwarded.

→ "U" flag indicates that route is up
"G" flag indicates that route is a gateway.

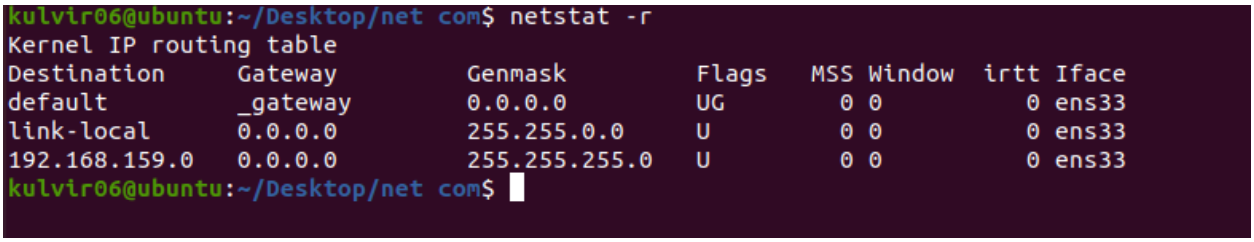
Code :

To install netstat : sudo apt install net-tools

To get routing table : netstat -r

Output Screenshot :

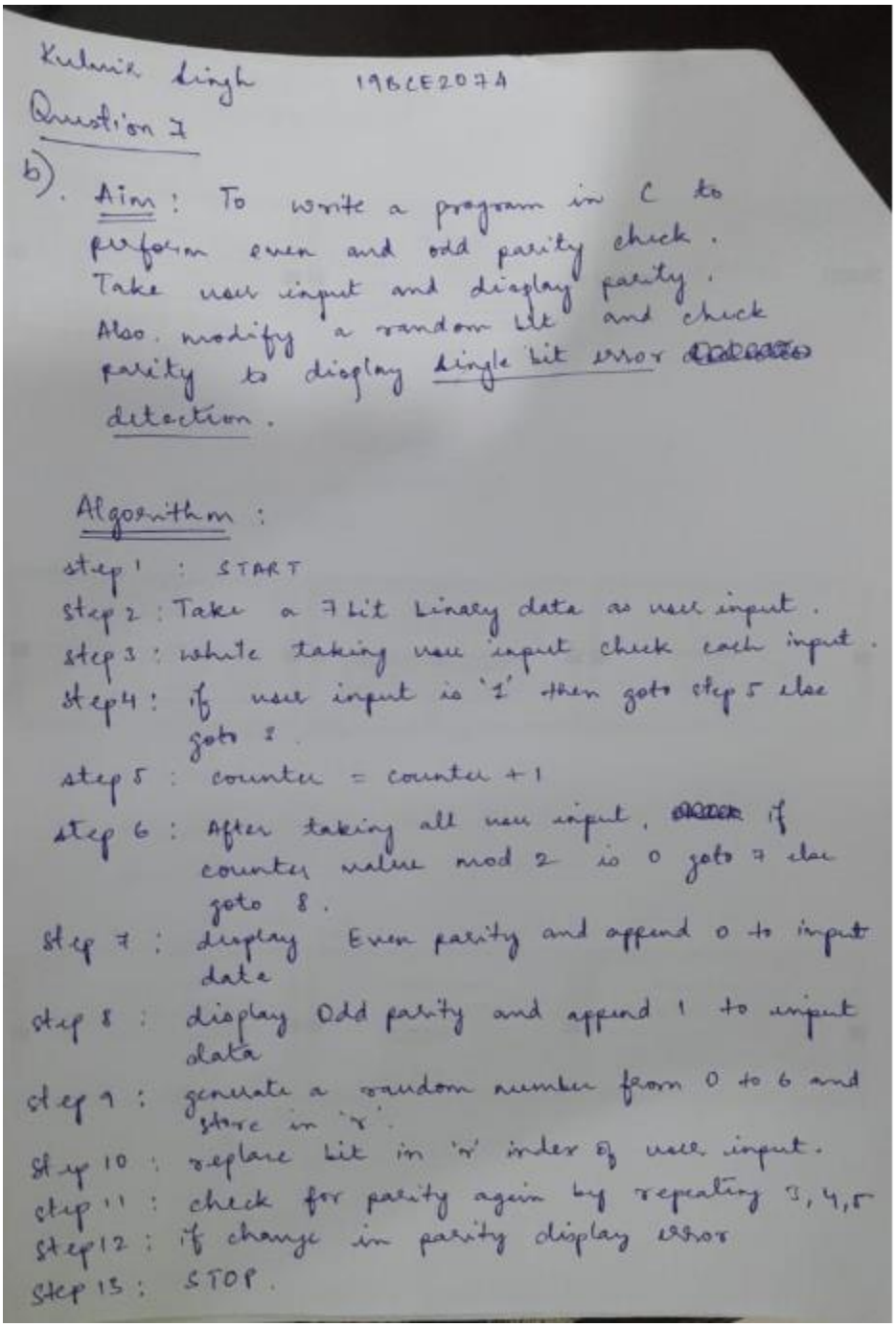
Output result is explained in results and discussion (handwritten above)



Question 7b)

Design a C code to perform odd or even parity

Aim and Algorithm



Kulwin Singh 19BCE2074

Question 7

b).

Aim: To write a program in C to perform even and odd parity check. Take user input and display parity. Also, modify a random bit and check parity to display single bit error detection.

Algorithm:

step 1 : START
step 2 : Take a 7 bit binary data as user input.
step 3 : while taking user input check each input.
step 4 : if user input is '1' then goto step 5 else goto 6.
step 5 : counter = counter + 1
step 6 : After taking all user input, if counter value mod 2 is 0 goto 7 else goto 8.
step 7 : display Even parity and append 0 to input data
step 8 : display Odd parity and append 1 to input data
step 9 : generate a random number from 0 to 6 and store in 'r'.
step 10 : replace bit in 'r' index of user input.
step 11 : check for parity again by repeating 3, 4, 5
step 12 : if change in parity display error
step 13 : STOP.

Code :

```
#include<iostream>
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
int main()
{
int data[8];
int i=0,c=0;
printf("Kulvir Singh - 19BCE2074\n");
printf("Enter the 7 bits\n");
for(i=0;i<7;i++)
scanf("%d",&data[i]);
for(i=0;i<7;i++)
{
if(data[i]==1)
c=c+1;
}
printf("\nData Bits Entered : ");
for(i=0;i<7;i++)
printf("%d",data[i]);
if(c%2==0)
{
printf("\nEven parity\n");
data[7]=0;
}
else
{
printf("\nOdd parity\n");
data[7]=1;
}
printf("original data:");
for(i=0;i<7;i++)
printf("%d",data[i]);
srand(time(0));
int r=0,j=0;
for(j=0;j<5;j++)
r=rand()%((8-0)+1);
printf("\nBit changed randomly at position:%d\n",r+1);
if(data[r]==1)
data[r]=0;
else
data[r]=1;
printf("\nData after adding the parity bits:");
for(i=0;i<8;i++)
printf("%d",data[i]);
```

```
int cn=0;
for(i=0;i<8;i++)
{
if(data[i]==1)
cn=cn+1;
}
if(cn%2==0)
printf("\nno error\n");
else
printf("\nerror\n");
return 0;
}
```

Output Screenshot :

The first output shows ODD PARITY. The bit at fourth position is changed and error is caught.

The second output shows EVEN PARITY. The bit at sixth position is changed and error is caught.

```
kulvir06@ubuntu:~/Desktop/net com/da 1$ ./a.out
Kulvir Singh - 19BCE2074
Enter the 7 bits
1
0
0
1
1
0
0

Data Bits Entered : 1001100
Odd parity
original data:1001100
Bit changed randomly at position:4

Data after adding the parity bits:10001001
error
kulvir06@ubuntu:~/Desktop/net com/da 1$ ./a.out
Kulvir Singh - 19BCE2074
Enter the 7 bits
1
0
0
1
0
1
1

Data Bits Entered : 1001011
Even parity
original data:1001011
Bit changed randomly at position:6

Data after adding the parity bits:10010010
error
kulvir06@ubuntu:~/Desktop/net com/da 1$
```