EX NO: 6

30/09/2021

**CONCURRENT SERVER**

**AIM:**

To create a TCP/IP concurrent server program to find the winner based on maximum points obtained in the lucky number game and display the winner.

**ALGORITHM:**

**SERVER:**

1. Include header files, initialize the required variables and specify the family, protocol, IP address and port number.
2. Create a socket using socket() function.
3. Bind the IP address and port number and listen and accept the different client’s request for connection.
4. Deploy the lucky number game between the various clients concurrently.
5. Display the winner based on the maximum points obtained.

**CLIENT:**

1. Include header files, initialize the required variables
2. specify the family, protocol, IP address and port number.
3. Create a socket using socket() function.
4. Call the connect() function.
5. Connect with the server for various clients to decide the winner.

**PROGRAM:**

**SERVER:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <time.h>

#define PORT 4545

struct LIST{

int item[10];

int flag[10];

}list,list1;

int main(){

int sockfd, val, i;

struct sockaddr\_in serverAddr;

int cSocket;

struct sockaddr\_in newAddr;

list.item[0]=1;

list.item[1]=3;

list.item[2]=5;

list.item[3]=10;

list.item[4]=15;

list.item[5]=25;

list.item[6]=50;

list.item[7]=100;

list.item[8]=110;

list.item[9]=150;

for(i=0;i<10;i++)

list.flag[i]=0;

list1=list;

socklen\_t addr\_size;

char buffer[1024],buff[1024];

pid\_t childpid;

sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

if(sockfd < 0){

printf("Error in connection...\n");

exit(1);

}

printf("Socket created successfully...\n");

memset(&serverAddr, '\0', sizeof(serverAddr));

serverAddr.sin\_family = AF\_INET;

serverAddr.sin\_port = htons(PORT);

serverAddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

val = bind(sockfd, (struct sockaddr\*)&serverAddr, sizeof(serverAddr));

if(val < 0){

printf("Binding error...\n");

exit(1);

}

printf("Bind to port %d...\n", 4545);

if(listen(sockfd, 10) == 0){

printf("Listening...\n");

}else{

printf("Binding error...\n");}

int iteration=0,client=0,client\_port[5],value=0,reward[5]={0},max\_reward[5]={0};

while(1){

cSocket = accept(sockfd, (struct sockaddr\*)&newAddr, &addr\_size);

if(cSocket < 0)

exit(1);

client\_port[client]=cSocket;

client=client+1;

printf("Connection accepted from %s:%d\n", inet\_ntoa(newAddr.sin\_addr) , ntohs(newAddr.sin\_port));

if((childpid = fork()) == 0){

close(sockfd);

if(client==5){

while(iteration<5){

int client\_value[5]={0};

printf("\n\nIteration%d:",iteration+1);

printf("\n\nServer:");

for(int i=0;i<10;i++)

printf("%d ",list.item[i]);

for(int j=0;j<5;j++){

send(client\_port[j], &list1, sizeof(list1), 0);

recv(client\_port[j],&list1,sizeof(list1),0);

recv(client\_port[j],&value,sizeof(value),0);

client\_value[j]=ntohl(value);

}

int lucky\_number=list.item[rand()%10];

printf("\nLucky number: %d",lucky\_number);

int c=-1;

for(int k=0;k<5;k++){

printf("\nclient%d: %d\t",k+1,client\_value[k]);

if(lucky\_number==client\_value[k]){

reward[k]=reward[k]+100;

c=k;

}

}

if(c==-1)

printf("\nAll clients get $0 as reward points...");

else

printf("\nClient%d gets $100 as reward points...",c+1);

for(int m=0;m<10;m++)

list.flag[m]=0;

list1=list;

iteration++;

}

if(iteration==5){

int max1=reward[0],c1=0;

for(int p=1;p<5;p++){

if(max1<reward[p]){

max1=reward[p];

c1=p;

}

}

printf("\n\nThe winner is:");

for(int i=0;i<5;i++){

if(max1==reward[i])

printf("\nClient%d: %d points",i+1,reward[i]);

}}}}}

close(cSocket);

return 0;

}

**CLIENT:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <time.h>

#define PORT 4545

struct LIST{

int item[10];

int flag[10];

}list1;

int main(){

int connfd, val, iteration=0;

struct sockaddr\_in serverAddr;

char buffer[1024];

connfd = socket(AF\_INET, SOCK\_STREAM, 0);

if(connfd < 0){

printf("Error in connection...\n");

exit(1);}

printf("Socket created successfully...\n");

memset(&serverAddr, '\0', sizeof(serverAddr));

serverAddr.sin\_family = AF\_INET;

serverAddr.sin\_port = htons(PORT);

serverAddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

val = connect(connfd, (struct sockaddr\*)&serverAddr, sizeof(serverAddr));

if(val < 0){

printf("Error in connection...\n");

exit(1);

}

printf("Connected to Server...\n");

while(iteration<5){

recv(connfd,&list1,sizeof(list1),0);

int renum=0,s;

while(1){

srand(time(NULL));

s=rand()%10;

if(list1.flag[s]!=1){

renum=list1.item[s];

list1.flag[s]=1;

break;

}

else

continue;

}

renum=htonl(renum);

send(connfd,&list1,sizeof(list1),0);

send(connfd,&renum,sizeof(renum),0);

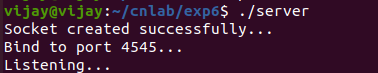
iteration++;

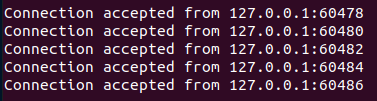
}

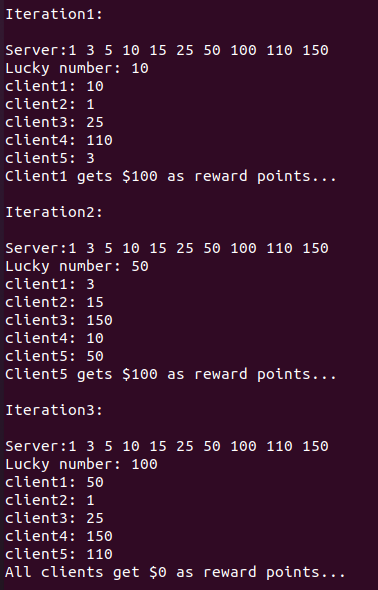
return 0;

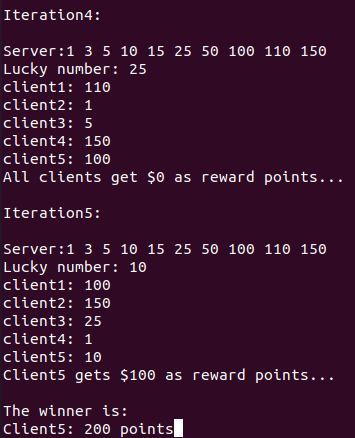
}

**SAMPLE OUTPUT:**









**RESULT:**

Hence the TCP/IP concurrent server program was created to find the winner based on maximum points obtained in the lucky number game and the output was verified.