COMP-8567
Advanced System Programming Project
Instructor: Dr. Boubakeur Boufama

Submitted by: Gundeep Singh
Student Id: 110008447

Solution:

According to the problem statement, I created two separate programs namely *Server.c* and *Client.c.* The program Server.c as by its names acts as the server and waits for both the players before starting the game. Once we get both of the clients , server forks and let the *child [servicePlayer method]* handle thse two players while it tends to wait for the next two players. A message is sent to the first player(Client.c) by the *servicePlayer* method that it can start the game now. Also to be noted that the first player joining the game is given a name "TOTO" by the Server and the second player joining the game is assigned a name "TITI" .

The Server.c program takes the argument "PORT NUMBER" while running , for our program we are taking the port number to be "9002". The Client.c class takes two arguments "IP address" and the "Port number".

I am taking *alpha* to act as my Server, which makes the *bravo* and the *charlie* to be the clients(players).

Both these players roll the dice turn by turn and send their respective scores to the server, the function servicePlayer acts as the referee and totals the score of both the players and validates if the score is equal to or greater than the value 100 before passing the control to the second player for its turn.

In case the score for a player is equal to or greater than the value 100, a win message is sent to the winner player and a lost message is sent to looser player hence closing the individual sockets of both the players(or clients).

The following is the working code of both the Server.c and the Client.c programs:

Server.c

```c
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<string.h>
#include<arpa/inet.h>
#include<stdint.h>
#include<inttypes.h>
#include<unistd.h>

void servicePlayers (int sd1,int sd2);
int main(int argc, char * argv[]){

    int sd, player1,player2, portNumber;

    socklen_t len;
    struct sockaddr_in servAdd;

    if(argc != 2){
        printf("call model: %s <PORT #>\n", argv[0]);
        exit(0);
    }

    if ((sd=socket(AF_INET, SOCK_STREAM,0))<0){

        fprintf(stderr, "cannot create socket \n");
        exit(1);
    }

    servAdd.sin_family = AF_INET;
    servAdd.sin_addr.s_addr = htonl(INADDR_ANY);
    sscanf( argv[1], "%d", &portNumber );
    servAdd.sin_port = htons((uint16_t)portNumber);
    bind(sd,(struct sockaddr*)&servAdd, sizeof(servAdd));
    listen(sd, 5);
```

```
while(1){

        player1 = accept(sd, (struct sockaddr*)NULL, NULL);
        printf("Got first player\n");
        player2 = accept(sd, (struct sockaddr*)NULL, NULL);
        printf("Got second player\n");
        printf("lets start the game\n");
        if(!fork()){
            servicePlayers(player1, player2);
        }
        close(player1);
        close(player2);
    }
}

void servicePlayers (int sd1, int sd2){

    int numeric1 = 0;
    int total1=0;
    int numeric2 = 0;
    int total2=0;
    char message[255];

    while(1){

        strcpy(message,"You can now play");
        write(sd1, message, strlen(message)+1);

        if(!read(sd1, message, 255)){
            close(sd1);
            fprintf(stderr, "BYE, player 1 dead, wait for a new Player\n");
            exit(0);
        }

        fprintf(stderr, "TOTO dice value: %s\n", message);
        //converting the string into integer
        sscanf(message,"%d",&numeric1);
        // printf("and in numeric integer its this %d\n",numeric1);
        total1 = total1+numeric1;
        printf("TOTO's total score so far is: %d\n",total1);
        printf("---------------------------------\n");
        sleep(2);
        if(total1 >= 100 ){
        printf("Game Over : TOTO Won the game.\n ");
```

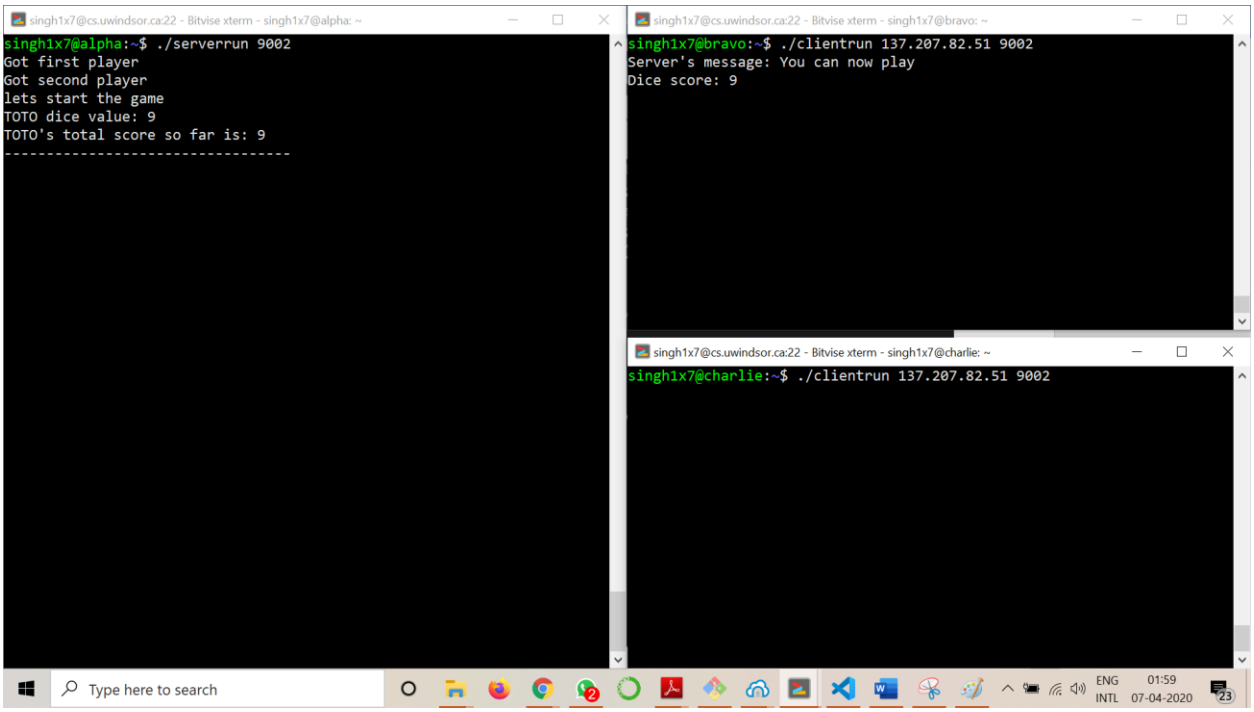
```
            strcpy(message,"Game over: you won the game");
            write(sd1, message, strlen(message)+1);

            strcpy(message,"Game over: you lost the game");
            write(sd2, message, strlen(message)+1);

            close(sd1);
            close(sd2);
            exit(0);
        }

        //for second player

        strcpy(message,"You can now play");
        write(sd2, message, strlen(message)+1);

        if(!read(sd2, message, 255)){
            close(sd2);
            fprintf(stderr, "BYE, player 2 dead, wait for a new player\n");
            exit(0);
        }

        fprintf(stderr, "TITI dice value: %s\n", message);
        sscanf(message,"%d",&numeric2);
        // printf("and in numeric integer its this %d\n",numeric2);
        total2 = total2+numeric2;
        printf("TITI's total score so far is: %d\n",total2);
        printf("---------------------------------\n");
    sleep(2);

        if(total2 >= 100 ){
        printf("Game Over : TITI Won the game.\n ");
        strcpy(message,"Game over: you won the game");
            write(sd2, message, strlen(message)+1);
            strcpy(message,"Game over: you lost the game");
            write(sd1, message, strlen(message)+1);

            close(sd1);
            close(sd2);
            exit(0);
        }
    }
}
```

Client.c :

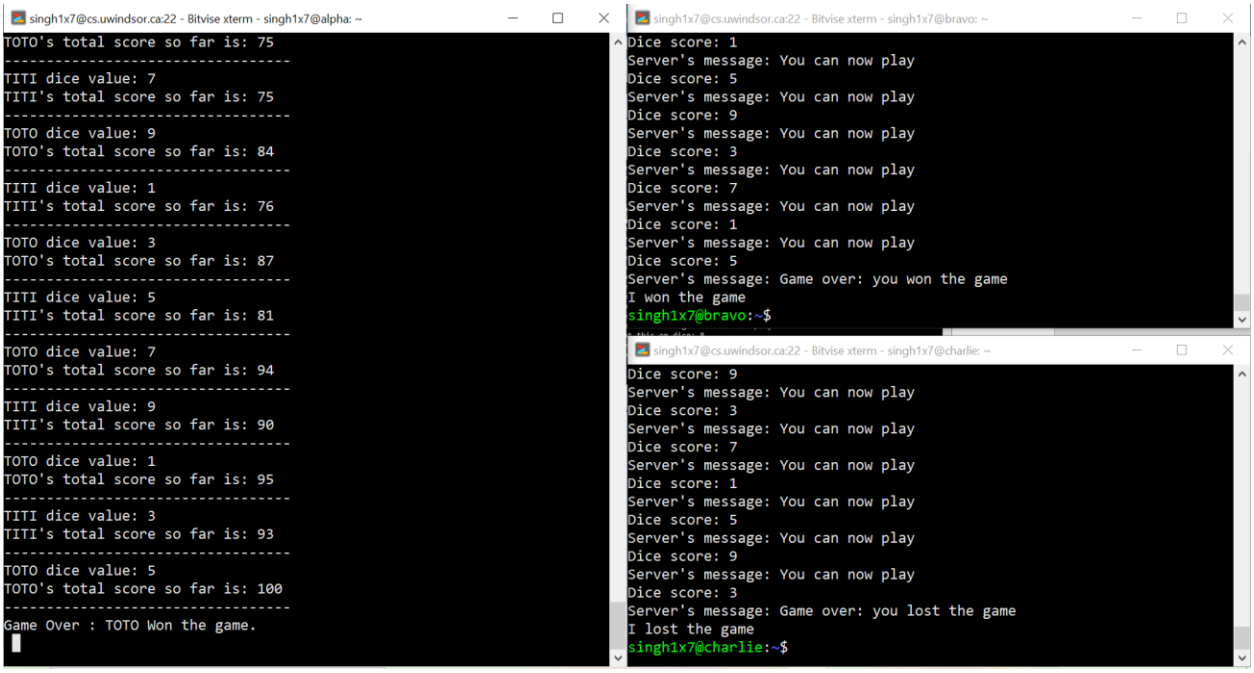
```c
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<string.h>
#include<arpa/inet.h>
#include<stdint.h>
#include<inttypes.h>
#include<unistd.h>

int main(int argc, char *argv[]){

    char message[255];
    int server, portNumber, dice;
    long int ss=0;
    socklen_t len;

    struct sockaddr_in servAdd;

    if(argc != 3){

        printf("Call model: %s <IP> <PORT#>\n",argv[0]);
        exit(0);
    }

    if((server =socket(AF_INET, SOCK_STREAM, 0)) < 0 ){
        fprintf(stderr, "Cannot create a socket\n");
        exit(1);
    }
    servAdd.sin_family = AF_INET;
    sscanf(argv[2], "%d", &portNumber);
    servAdd.sin_port = htons((uint16_t)portNumber);


    if(inet_pton(AF_INET,argv[1], &servAdd.sin_addr)<0){
        fprintf(stderr, "inet_pton() has failed\n");
    }
    if(connect(server,(struct sockaddr *)&servAdd, sizeof(servAdd))<0){
        fprintf(stderr, "connect() has failed, exiting\n");
        exit(3);
    }
```

```
    while(1){

        if(read(server, message, 255)<0){
            fprintf(stderr, "read() error");
            exit(3);
        }

        fprintf(stderr, "Server's message: %s\n", message);

        if(! (strcmp(message,"You can now play"))){
            // printf("Its my chance to play now\n");
            dice = (int)(time(&ss))%10 +1;
            printf("Dice score: %d\n",dice);
            sprintf(message, "%d", dice );
            write(server, message, strlen(message)+1);

        }
        if(! (strcmp(message,"Game over: you won the game"))){
            printf("I won the game\n");
            close(server);
            exit(0);
        }
        if(! (strcmp(message,"Game over: you lost the game"))){
            printf("I lost the game\n");
            close(server);
            exit(0);

        }

    }

}
```

Output:



This above output shows both clients/players connected and TOTO going for first turn.



This above output shows the end of the game where TOTO wins and server again waits for 2 another player.