

*Prepared by:*

*Ali Atwi*

*2021-2022*

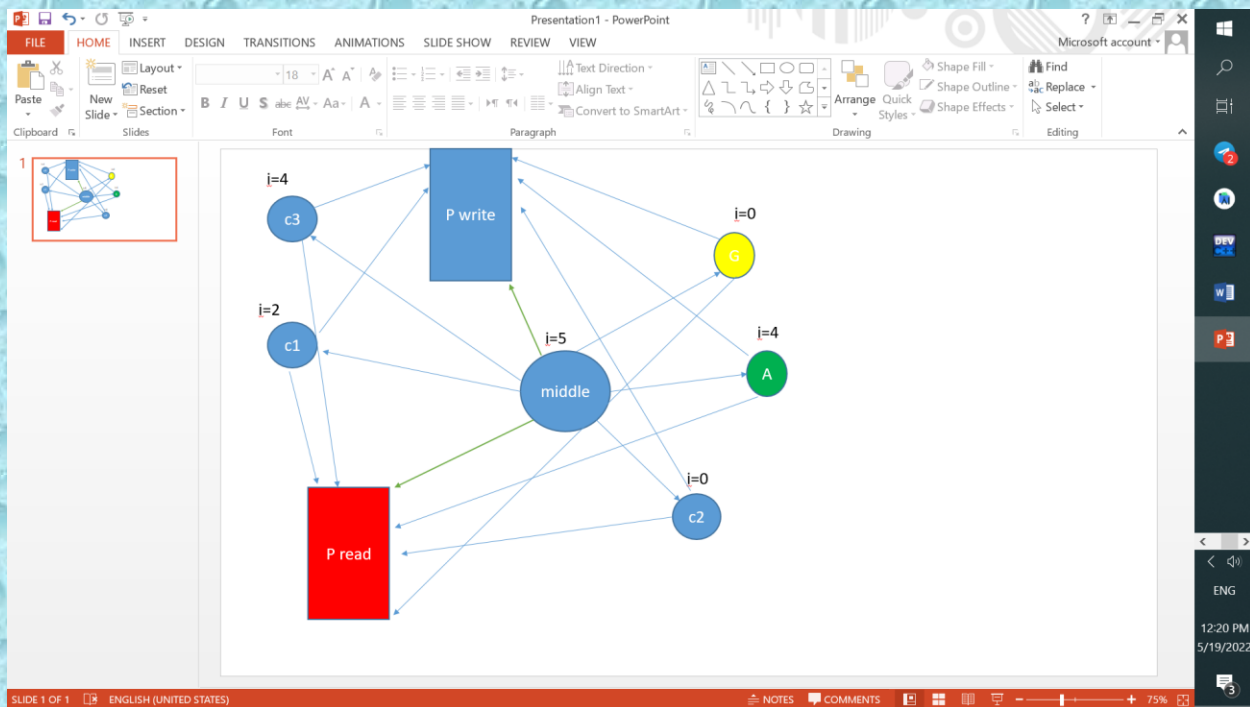
*Project submitted in the context of* **Operating System**

# *GAMING\_ SYSTEM*

## **Acknowledgements**

## **Abstract**

In general; to be clear, the communication between all processes is as follows:



There are two pipes(descriptor tables) one for read and the other for write. The parent(middle) creates 5 children processes, every child can read resp. write from Pread/Pwrite. However the parent reads from Pwrite and write in Pread. In fact, the pid of all children is stored in the array `pidclients[]`.

All the children will close unused pipes(`close(pw[0])` `close(pr[1])`) to avoid congestion, similarly for the parent he will close (`pw[1]` `close(pr[0])`)...

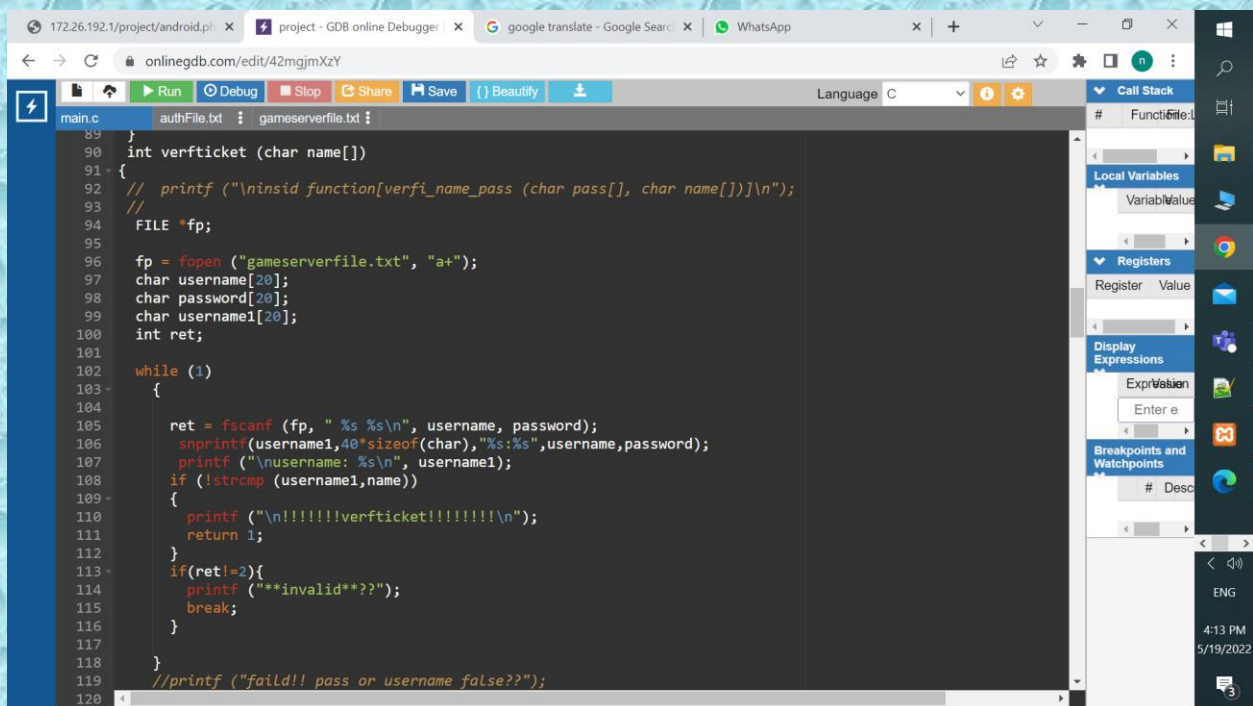
After finishing creation of processes, they will pause until receive a signal in order to perform specific task...

Sending messages process is done as follows:

for example if client1 wants to send his information to A to authenticate, he will first put them in `pwwrite` using `clientwritehandler` function and send a signal to the middle (I wrote some messages in the pipe `pwwrite` go and read...). Here the `middle` read handler function will be called, so it will be read by him and then he will put the message in `pread` in order to let A read it without confusion...



**GAMING\_SYSTEM**: this application is used to remotely manage gaming process between clients and servers through Authentication, where the client sends the Auth-server an Auth-message containing information about him(username:password). Here is the role of Auth-server in which he verifies whether the information entered are correct or not by comparing the input with the data stored for the client in a text file containing username:password using verfticket function(message1)...

A screenshot of a web-based code editor interface. The browser's address bar shows 'onlinegdb.com/edit/42mgjmXzY'. The editor has a dark theme and shows C code for a function named 'verfticket'. The code includes file operations to read and write to 'gameserverfile.txt'. It uses 'fscanf' to read a username and password, and 'strcmp' to compare the input with the stored data. If the data matches, it returns 1; otherwise, it prints an invalid message and breaks the loop. The right sidebar contains panels for 'Call Stack', 'Local Variables', 'Registers', 'Display Expressions', 'Expression', and 'Breakpoints and Watchpoints'. The bottom right corner shows the system clock as 4:13 PM on 5/19/2022.

```
main.c
authFile.txt
gameserverfile.txt

89 }
90 int verfticket (char name[])
91 {
92     // printf ("\ninsid function[verfi_name_pass (char pass[], char name[])\n");
93     //
94     FILE *fp;
95
96     fp = fopen ("gameserverfile.txt", "a+");
97     char username[20];
98     char password[20];
99     char username1[20];
100     int ret;
101
102     while (1)
103     {
104
105         ret = fscanf (fp, " %s %s\n", username, password);
106         snprintf(username1,40*sizeof(char),"%s:%s",username,password);
107         printf ("\nusername: %s\n", username1);
108         if (!strcmp (username1,name))
109         {
110             printf ("\n!!!!!!verfticket!!!!!!\n");
111             return 1;
112         }
113         if(ret!=2){
114             printf ("***invalid***?");
115             break;
116         }
117     }
118     //printf ("faield!! pass or username false??");
119
120 }
```

But before that, at the beginning of this app, the user should choose whether he wants to login(his info will be directly stored in the file auth.txt,by choosing the login option..),chat(in the case he wants to chat with his brothers),or game...

note:if the player is no longer login will not be able to play so enter login will be redisplayed for him

Moreover, at the beginning, the user of the app must give the keyboard to one of the players (first we consider all the clients are not connected  $connect=0$ , so if it will be 1 if any one is connecting to the game...)

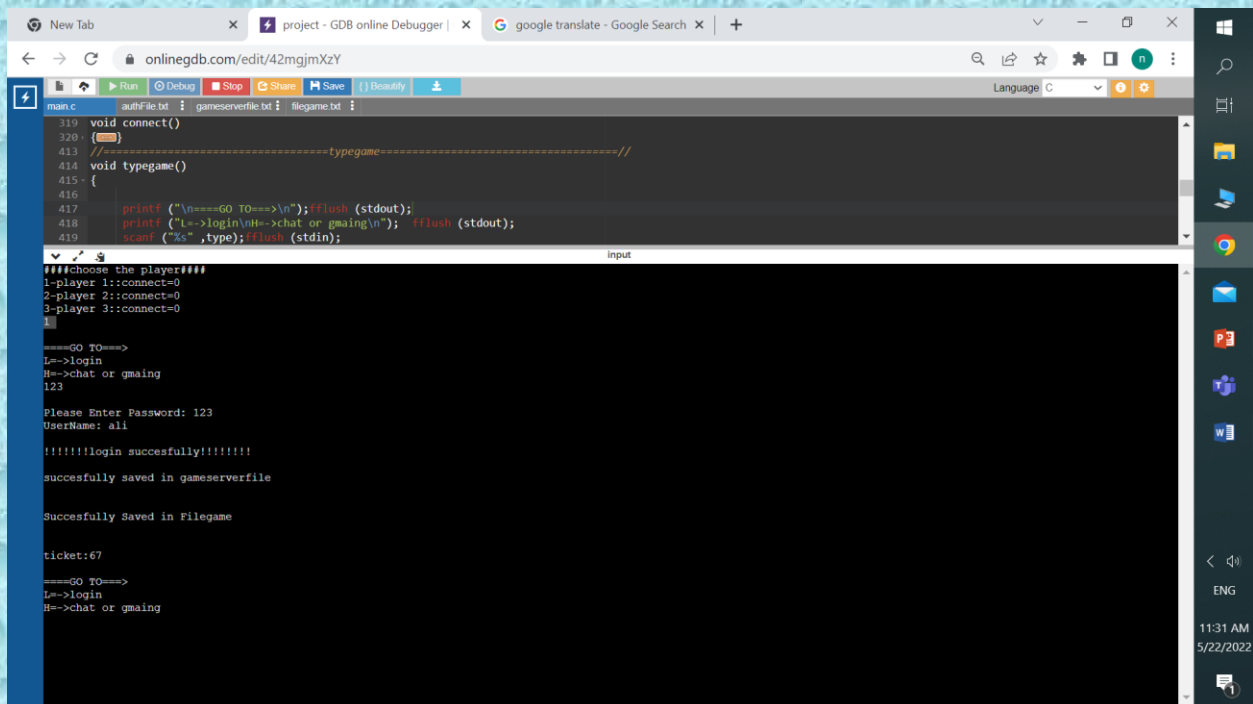
So, message1 is being handled using this simple function

After the first run of the program, a prompt message popped out to let the user choose a player to let him connect to the game. When this is done,  $connect$  will be ( $connect=1$ ). That will help us know the number of connected players in order to initialize the game or not..

In the below scenario, player 1 has been chosen to play, he can't play without login.

I entered here a valid username and password saved in my file authFile. So as we can see here, a ticket random number has been generated by the auth and displayed (67).

Again is a confirm display to complete playing...



```
319 void connect()
320 {
413 //=====typegame=====
414 void typegame()
415 {
416     printf ("\n====GO TO====\n"); fflush (stdout);
417     printf ("L->login\n"); fflush (stdout);
418     scanf ("%s", type); fflush (stdin);
419 }

####choose the player####
1-player 1::connect=0
2-player 2::connect=0
3-player 3::connect=0
1

====GO TO====
L->login
H->chat or gmaing
123

Please Enter Password: 123
UserName: ali

!!!!!!login succesfully!!!!!!

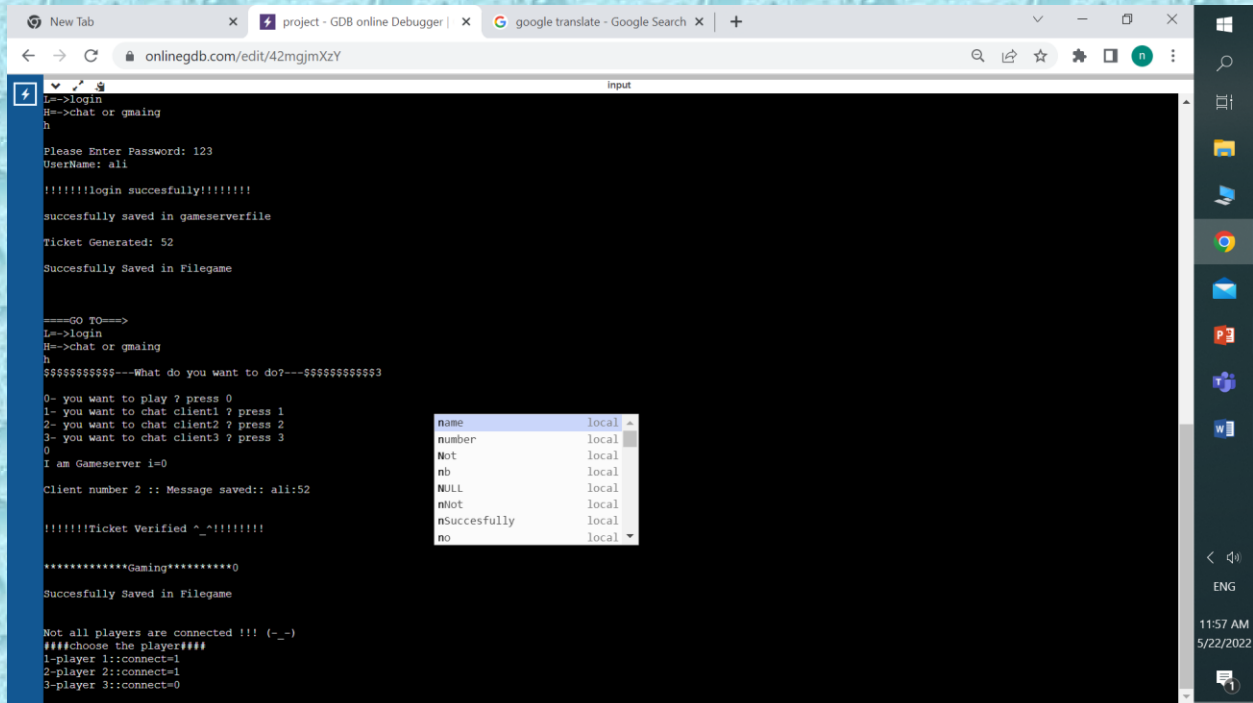
succesfully saved in gameserverfile

Succesfully Saved in Filegame

ticket:67

====GO TO====
L->login
H->chat or gmaing
```





```
L->login
H->chat or gmaing
h
Please Enter Password: 123
UserName: ali
!!!!!!login succesfully!!!!!!
succesfully saved in gameserverfile
Ticket Generated: 52
Succesfully Saved in Filegame

====GO TO====
L->login
H->chat or gmaing
h
$$$$$$$$$---What do you want to do?---$$$$$$$$$
0- you want to play ? press 0
1- you want to chat client1 ? press 1
2- you want to chat client2 ? press 2
3- you want to chat client3 ? press 3
0
I am Gameserver i=0
Client number 2 :: Message saved:: ali:52

!!!!!!Ticket Verified ^_!!!!!!

*****Gaming*****0
Succesfully Saved in Filegame

Not all players are connected !!! (-_-)
***choose the player****
1-player 1::connect=1
2-player 2::connect=1
3-player 3::connect=0
```

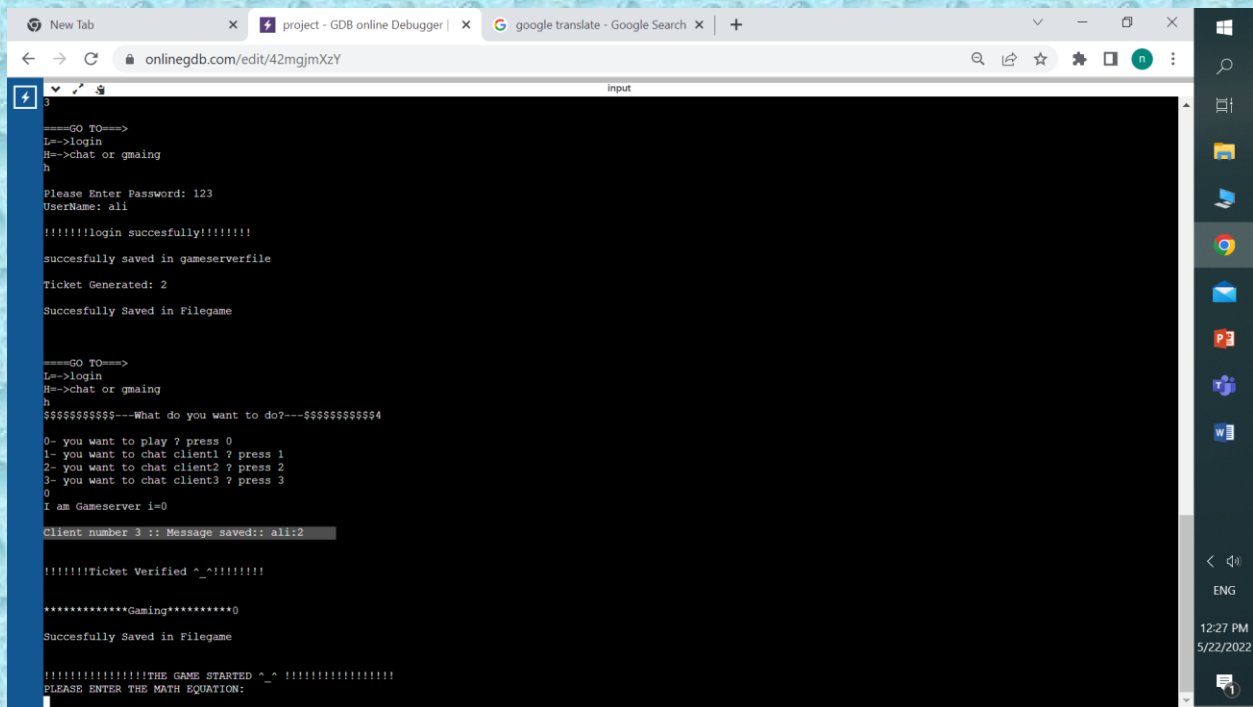
name	local
number	local
Not	local
nb	local
NULL	local
nNot	local
nSuccesfully	local
no	local

Here what happens is that another player connected to the game ,so new unique ticket has been generated ,thus saved in the gameserverFile.txt. infact the new connected player must login as seen

Here because the 3 clients are connected so the game can start .

As you can see that the game server (\*\*\*\*\*0) of I=0 is putting the question and sent it to the players,also the answer is provided by him, but here answer is not sent to the clients

It is saved in an array called answer see the figure next the below



```
3
====GO TO====
L->login
H->chat or gmaing
h
Please Enter Password: 123
UserName: ali
!!!!!!login succesfully!!!!!!
succesfully saved in gameserverfile
Ticket Generated: 2
Succesfully Saved in Filegame

====GO TO====
L->login
H->chat or gmaing
h
$$$$$$$$$---What do you want to do?---$$$$$$$$$4
0- you want to play ? press 0
1- you want to chat client1 ? press 1
2- you want to chat client2 ? press 2
3- you want to chat client3 ? press 3
0
I am Gameserver i=0
Client number 3 :: Message saved:: ali:2

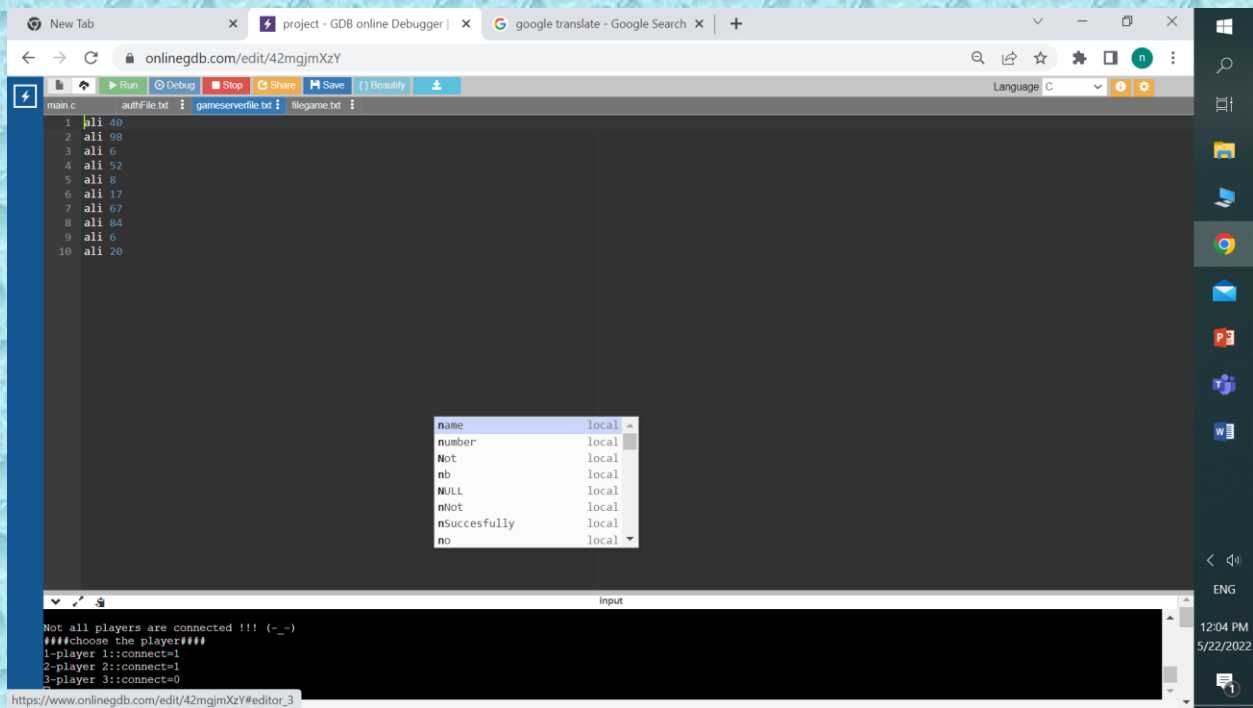
!!!!!!ticket Verified ^_^!!!!!!

*****Gaming*****0
Succesfully Saved in Filegame

!!!!!!!!!!!!!!THE GAME STARTED ^_^ !!!!!!!!!!!!!!!
PLEASE ENTER THE MATH EQUATION:
```

above his information is saved in authfile.txt to be done in thenext login as indicated before.here the game can't start since the number of connected players is <3

## GameserverFile



```
main.c
authFile.txt
gameserverfile.txt
filegame.txt
Language: C

1 pli 40
2 ali 98
3 ali 6
4 ali 52
5 ali 8
6 ali 17
7 ali 67
8 ali 84
9 ali 6
10 ali 20

name local
number local
Not local
nb local
NULL local
nlot local
nSuccesfully local
no local

Not all players are connected !!! (-_-)
###choose the player###
1-player 1::connect=1
2-player 2::connect=1
3-player 3::connect=0
```

Authfile:

The screenshot shows a web browser window with the URL `onlinegdb.com/edit/42mgjmXzY`. The browser has several tabs open: "New Tab", "project - GDB online Debugger", and "google translate - Google Search". The GDB interface includes a menu bar with options like "Run", "Debug", "Stop", "Share", "Save", "Locality", and "User". Below the menu bar, there are tabs for "main.c", "authfile.txt", "gameserverfile.txt", and "ilegame.txt". The "main.c" tab is active, showing the following code:

```
1 1223 : ali
2 1234 : ali1
3 1235 : ali2
4 1236 : ali3
```

A variable window is open, displaying a list of variables and their values:

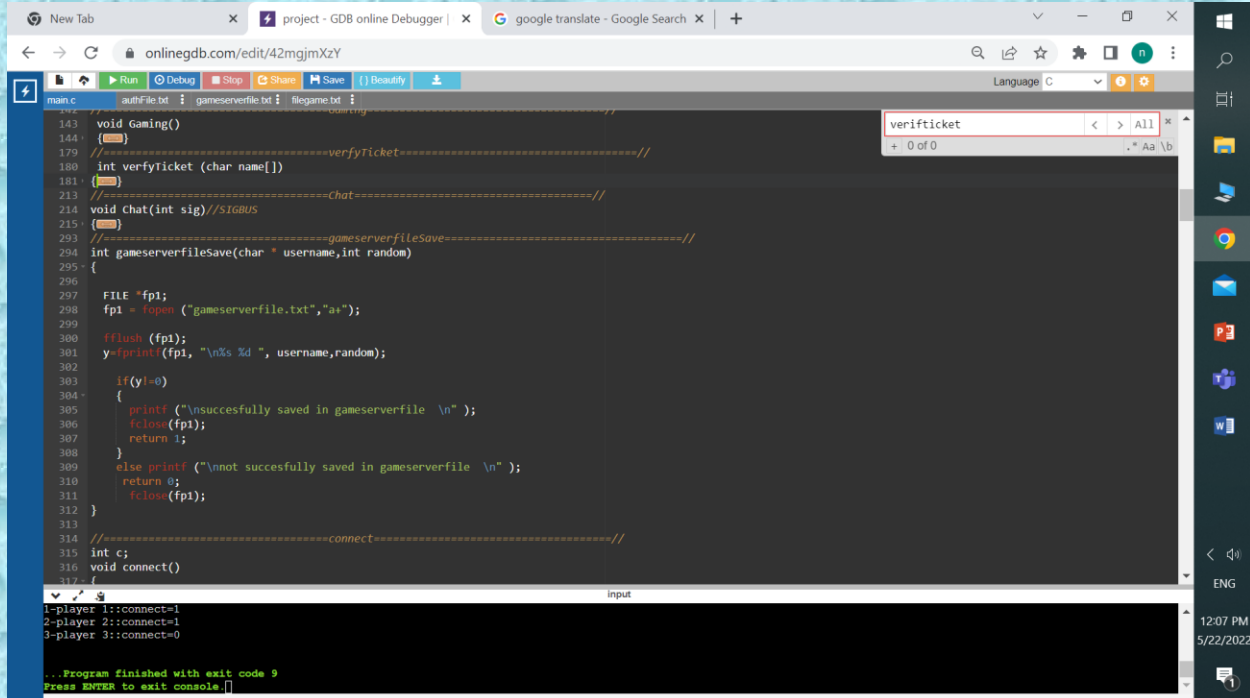
name	local
number	local
Not	local
nb	local
NULL	local
nNot	local
nSuccessfully	local
no	local

The "Input" window at the bottom shows the following output:

```
Not all players are connected !!! (-_-)
####choose the player####
1-player 1::connect=1
2-player 2::connect=1
3-player 3::connect=0
```

Functionality of saving in gameserverfile (imperative programming second year)





```
143 void Gaming()
144 {
145 //=====verifyTicket=====//
179
180 int verifyTicket (char name[])
181 {
213 //=====Chat=====//
214 void Chat(int sig)//SIGBUS
215 {
293 //=====gameserverfilesave=====//
294 int gameserverfilesave(char * username,int random)
295 {
296
297 FILE *fp1;
298 fp1 = fopen ("gameserverfile.txt","a+");
299
300 fflush (fp1);
301 y-fprintf(fp1, "\n%s %d ", username,random);
302
303 if(y!=0)
304 {
305 printf ("\nsuccesfully saved in gameserverfile \n" );
306 fclose(fp1);
307 return 1;
308 }
309 else printf ("\nnot succesfully saved in gameserverfile \n" );
310 return 0;
311 fclose(fp1);
312 }
313
314 //=====connect=====//
315 int c;
316 void connect()
317 {
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
1-player 1::connect=1
2-player 2::connect=1
3-player 3::connect=0

...Program finished with exit code 9
Press ENTER to exit console.
```

Connect function is called according to the value entered by the user we will talk about it in chat functionality in the case of playing , user wnters 0 so case:0 will be entered (login essential).

Now when the number of connected clients is 3,the game begin.

Here the game server enters the equation 2+2 for ex or any question/mysterword.

Followed by the solution.so every client send an answer ,after that all this answers is stored in a file called filegame.txt (each answer on a line) thus verified using a function called fileGame(char\*s, int x,int n,char\* intent) takes the file name and the mode (x,n is for comparing either char or int)note that comparison and saving is implimented in the same function



```
project - GDB online D... WhatsApp GDB online Debugger localhost/project/getP... google translate - Goo...
onlinegdb.com/edit/42mgjmXzY

Ticket Generated: 82
====GO TO====
L->login
H->chat or gmaing
h
$$$$$$$$$---What do you want to do?---$$$$$$$$$4
0- you want to play ? press 0
1- you want to chat client1 ? press 1
2- you want to chat client2 ? press 2
3- you want to chat client3 ? press 3
enter=>0

-----
Client number 3 :: Message saved:: ali:82
!!!!!!Ticket Verified ^_^!!!!!!

*****Gaming*****0
!!!!!!!!!!!!!!THE GAME STARTED ^ ^ !!!!!!!!!!!!!!!
PLEASE ENTER THE MATH EQUATION: 2+2

Iam the gameserver the solution is: 4
*****0
client -1::Question: 2+2

Answer=5
message=5
message=2+2
client -1::Question: 2+2

Answer=2
message=2
message=2+2
client -1::Question: 2+2

Answer=4
message=4
client4!!!!!!win!!!!!!
*****Game Over*****
```

```
project - GDB online D... WhatsApp GDB online Debugger localhost/project/getP... google translate - Goo...
onlinegdb.com/edit/42mgjmXzY

main.c authFile.txt gameserverfile.txt filegame.txt
Language: C

98
99 else if(n=1 && !strcmp(intent,"r"))
100 {
101     char line[10];
102     char line1[256];
103     int count=0;
104     int k=0,ind=0;
105     // fgets(line1,50*sizeof(char),fp1);
106     // k++;
107     while(fgets(line,20*sizeof(char),fp1))
108     {
109         if(k!=0)
110         {
111             strcpy(line,trimstring2(line));
112             // printf(" %d %d,%d\n",strcmp(line,solution),strlen(line),strlen(solution));fflush (stdout);
113             // printf("count: %d solution: %s line: %s\n",count,solution,line);fflush (stdout);
114             if(!strcmp(line,solution))
115             {
116                 printf("\nclient%d!!!!!!win!!!!!!\n",count);fflush (stdout);
117                 ind=1;
118             }
119         }
120         k++;
121         count++;
122     }
123     if(ind==0 && strcmp(line,solution) )
124     {
125         printf("no one win ^_^\n ");fflush (stdout);
126     }
127 }
128
129 if(y==0)
130
```

```
client4!!!!!!win!!!!!!
*****Game Over*****Killed

...Program finished with exit code 9
Press ENTER to exit console.
```

Snprintf is to concatenate ,written in pipe is done inorder to be read by parent and put them in pread to be read by server(gemeserver, auth..).note: in the case the user wants to connect to him self (can't connect .. message is displayed)

SIGURG is for read handler, here kill msg is sent to the parent(getppid()) to read from pipe in parallel the parent is ready for this signal to execute the handler function

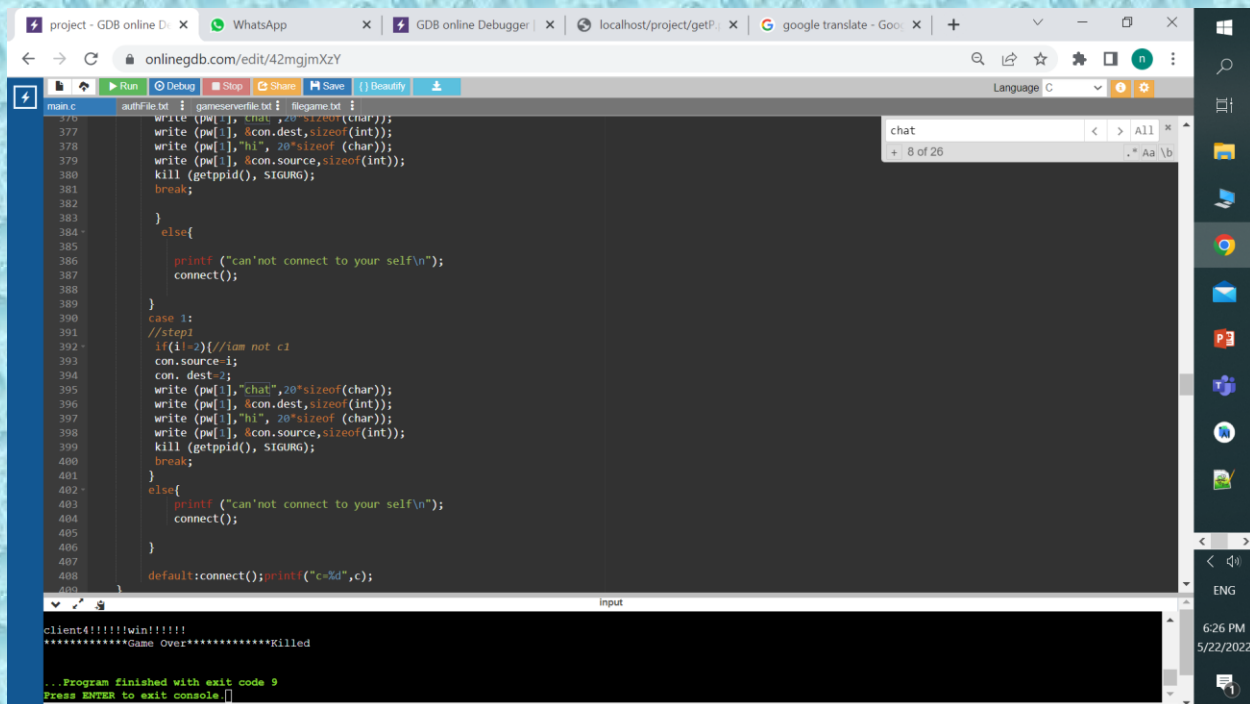
SIGHUP is for write handler..

SIGUSR1 resp. 2 is for read/write of the client

SIGBUS is for chat..

Chat process:

Chat is an important function, it is as indicated above if the type entered is chat or play, the user have free to choose, then if he choose a player the keyboard is for him while he choose with whom he wants to chat for ex:



The screenshot shows a web-based IDE with a C program for a chat server. The code is as follows:

```
376 write (pw[1], chat, 20*sizeof(char));
377 write (pw[1], &con.dest, sizeof(int));
378 write (pw[1], "hl", 20*sizeof(char));
379 write (pw[1], &con.source, sizeof(int));
380 kill (getppid(), SIGURG);
381 break;
382 }
383 }
384 else{
385     printf ("can't connect to your self\n");
386     connect();
387 }
388 }
389
390 case 1:
391     //step1
392     if(i!=2){//iam not c1
393         con.source=i;
394         con.dest=2;
395         write (pw[1], "chat", 20*sizeof(char));
396         write (pw[1], &con.dest, sizeof(int));
397         write (pw[1], "hl", 20*sizeof(char));
398         write (pw[1], &con.source, sizeof(int));
399         kill (getppid(), SIGURG);
400         break;
401     }
402     else{
403         printf ("can't connect to your self\n");
404         connect();
405     }
406 }
407
408 default:connect();printf ("c=%d",c);
409 }
```

The terminal output at the bottom shows:

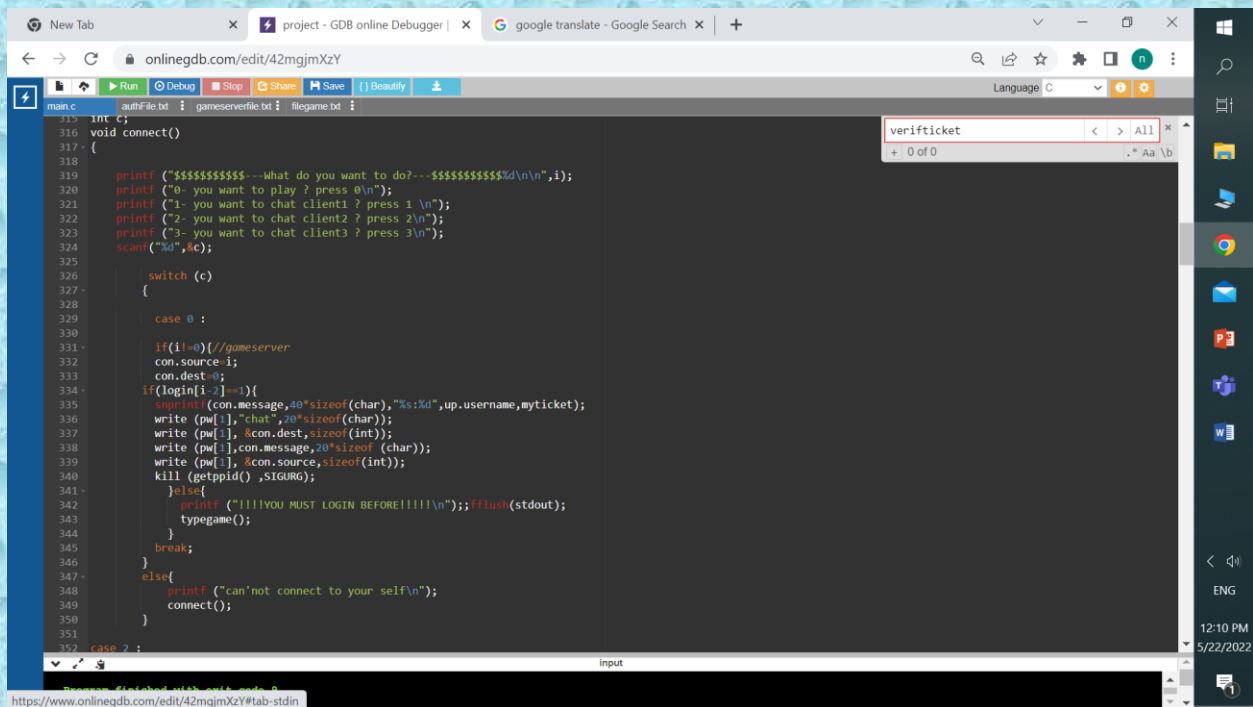
```
client4!!!!win!!!!
*****Game Over*****Killed
...Program finished with exit code 9
Press ENTER to exit console.
```

Here if client wants to chat with client1 so he enters case1 (the sender is not client1)

Above picture indicates that the destination is client1 with i=2.

This message is written in the pipe then readed by the parent→then written in pread by the parent .

To be readed by the destination this is an abstraction for chatting process



```
315 int c;  
316 void connect()  
317 {  
318  
319     printf ("$$$$$$$$$$$$---What do you want to do?---$$$$$$$$$$$$\n",i);  
320     printf ("0- you want to play ? press 0\n");  
321     printf ("1- you want to chat client1 ? press 1\n");  
322     printf ("2- you want to chat client2 ? press 2\n");  
323     printf ("3- you want to chat client3 ? press 3\n");  
324     scanf("%d",&c);  
325  
326     switch (c)  
327     {  
328  
329         case 0 :  
330  
331             if(i!=0){//gameserver  
332                 con.source=i;  
333                 con.dest=0;  
334                 if(login[i]==1){  
335                     sprintf(con.message,40*sizeof(char),"%s:%d",up.username,myticket);  
336                     write (pw[i], "chat",20*sizeof(char));  
337                     write (pw[i], &con.dest,sizeof(int));  
338                     write (pw[i],con.message,20*sizeof(char));  
339                     write (pw[i], &con.source,sizeof(int));  
340                     kill (getppid() ,SIGURG);  
341                 }else{  
342                     printf ("!!!!YOU MUST LOGIN BEFORE!!!!\n");fflush(stdout);  
343                     typegame();  
344                 }  
345                 break;  
346             }  
347             else{  
348                 printf ("can't connect to your self\n");  
349                 connect();  
350             }  
351         case 2 :  
352     }
```

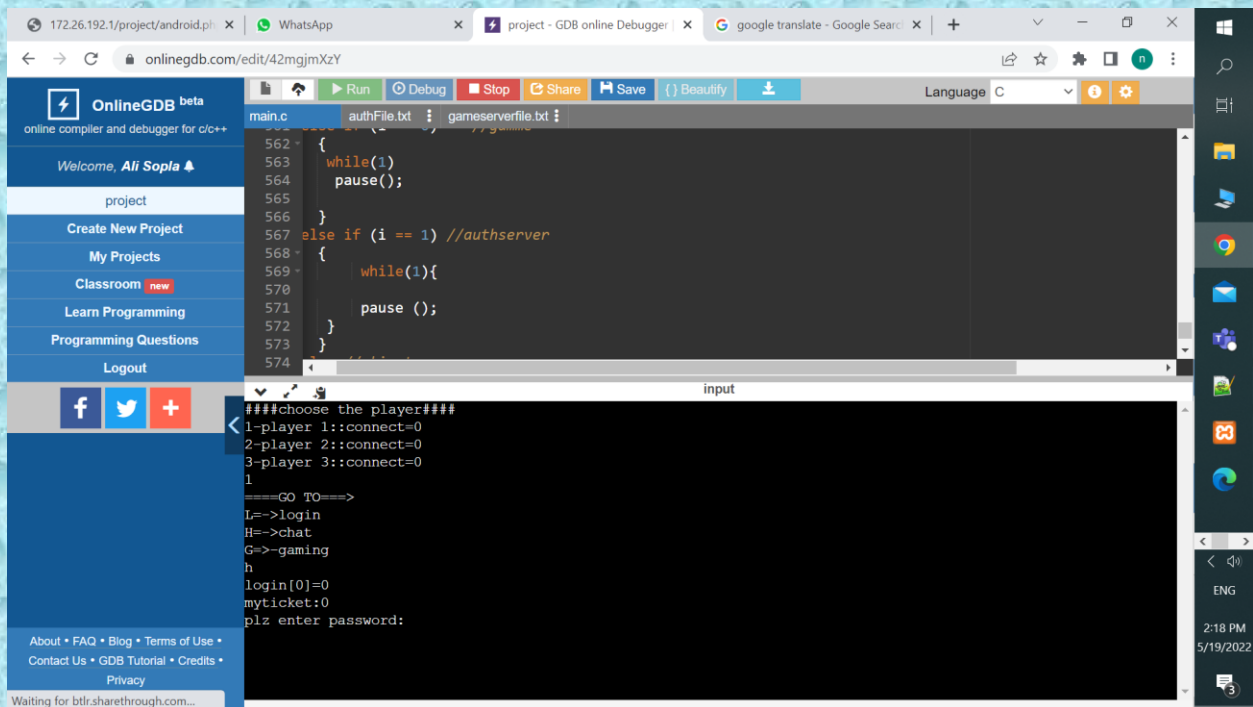
After that, in the case where the login is successfully done, A will generate a unique random variable between 1 and 100, available for only one round [ srand(time(0))

ra= rand()%100 +1] this process will be done by calling the function int createtickets() (return 1 if succ generated and 0 otherwise)

in the case the user enter h character thus he wants to make chat conversation with his friend

he will not be able to chat instead he make login the following screenshot illaustrate that if the user wants to chat he must login beforelogin[0]=0 it is an indicator that the player has not login yet:





The screenshot shows the OnlineGDB web interface. The top navigation bar includes links for 'Welcome, Ali Sopla', 'project', 'Create New Project', 'My Projects', 'Classroom', 'Learn Programming', 'Programming Questions', and 'Logout'. The main editor area displays a C program with the following code:

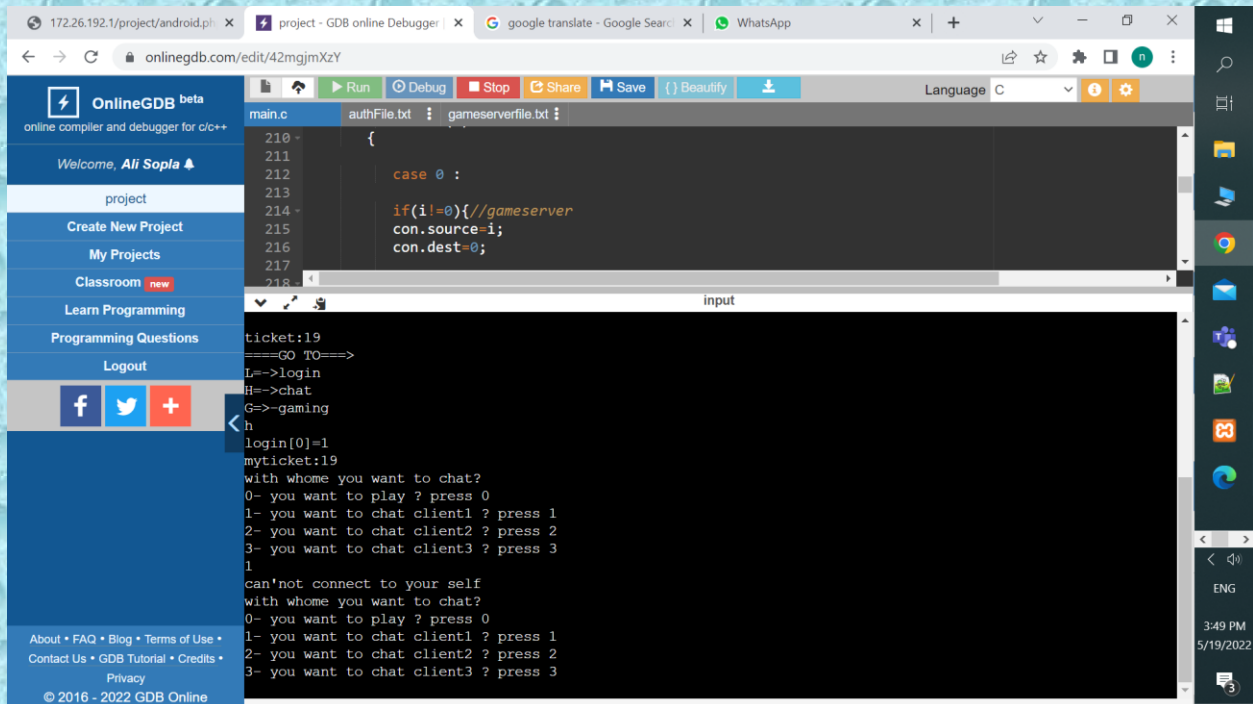
```
main.c
562 {
563     while(1)
564     pause();
565 }
566
567 else if (i == 1) //authserver
568 {
569     while(1){
570         pause ();
571     }
572 }
573
574
```

The terminal window shows the following output:

```
input
####choose the player####
1-player 1::connect=0
2-player 2::connect=0
3-player 3::connect=0
1
====GO TO====
L->login
H->chat
G->-gaming
h
login[0]=0
myticket:0
plz enter password:
```

However, after login if the user wants to make a chat with himself!!!

An alert displayed to rechoose destination



The screenshot shows the OnlineGDB web interface. The top navigation bar includes links for 'Welcome, Ali Sopla', 'project', 'Create New Project', 'My Projects', 'Classroom', 'Learn Programming', 'Programming Questions', and 'Logout'. The main editor area displays a C program with the following code:

```
main.c
210 {
211
212     case 0 :
213
214     if(i!=0){//gameserver
215         con.source=i;
216         con.dest=0;
217     }
218 }
```

The terminal window shows the following output:

```
input
ticket:19
====GO TO====
L->login
H->chat
G->-gaming
h
login[0]=1
myticket:19
with whom you want to chat?
0- you want to play ? press 0
1- you want to chat client1 ? press 1
2- you want to chat client2 ? press 2
3- you want to chat client3 ? press 3
1
can't connect to your self
with whom you want to chat?
0- you want to play ? press 0
1- you want to chat client1 ? press 1
2- you want to chat client2 ? press 2
3- you want to chat client3 ? press 3
```



***BEST REGARDS***