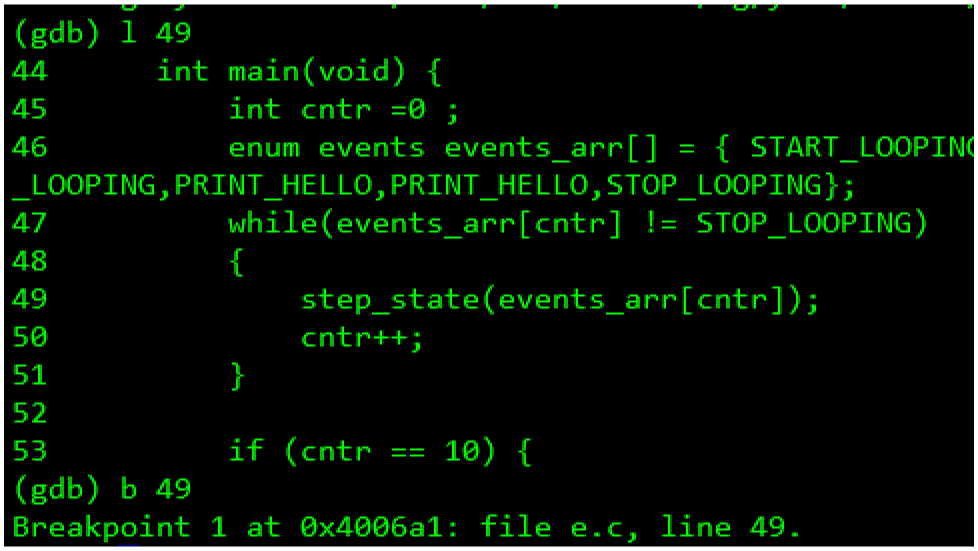
ASSIGNMENT 3

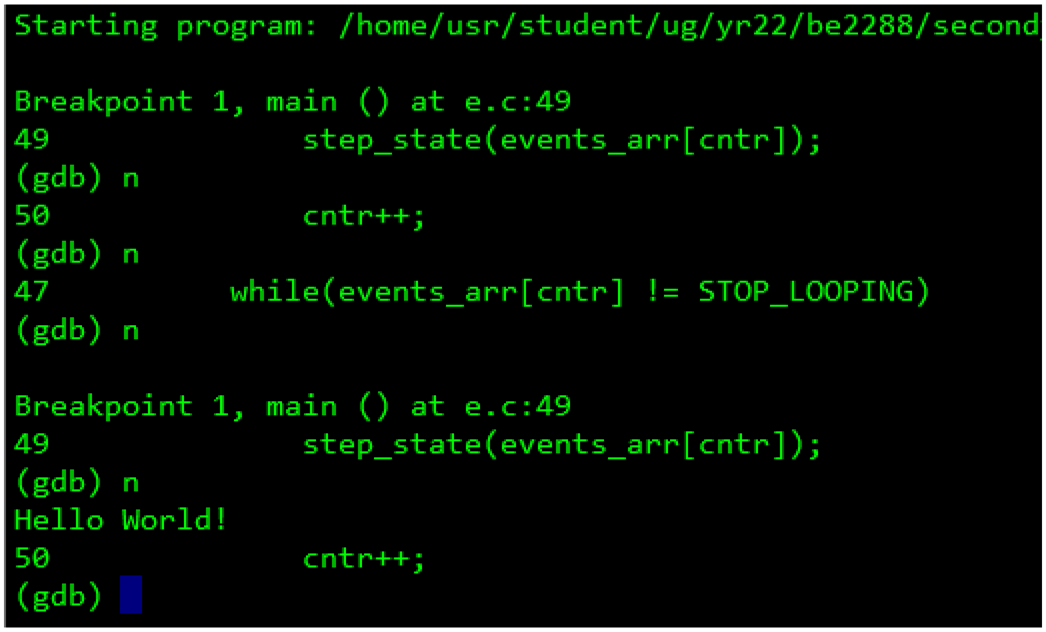
Name: Hrithvik kondalkar
Roll : 002211001088
section: A2

1. Consider the program in Assign3.It is a simple state machine.
   a. Put a breakpoint in line 49



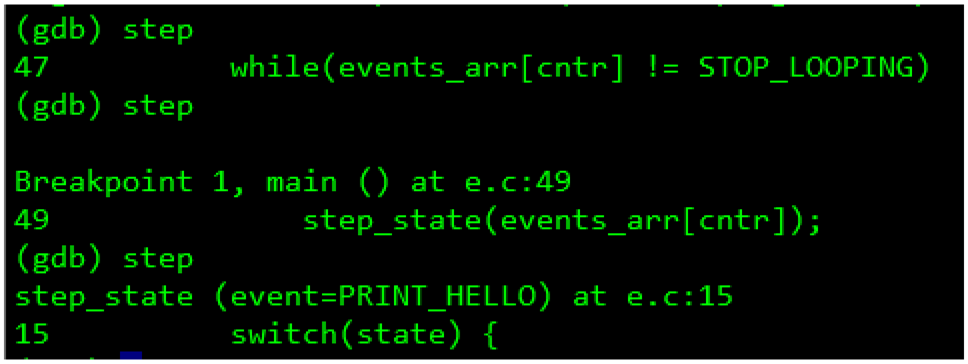
```
(gdb) l 49
44      int main(void) {
45          int cntr =0 ;
46          enum events events_arr[] = { START_LOOPIN
_LOOPING,PRINT_HELLO,PRINT_HELLO,STOP_LOOPING};
47          while(events_arr[cntr] != STOP_LOOPING)
48          {
49              step_state(events_arr[cntr]);
50              cntr++;
51          }
52
53          if (cntr == 10) {
(gdb) b 49
Breakpoint 1 at 0x4006a1: file e.c, line 49.
```

   b. Try next command



```
Starting program: /home/usr/student/ug/yr22/be2288/second

Breakpoint 1, main () at e.c:49
49              step_state(events_arr[cntr]);
(gdb) n
50              cntr++;
(gdb) n
47          while(events_arr[cntr] != STOP_LOOPING)
(gdb) n

Breakpoint 1, main () at e.c:49
49              step_state(events_arr[cntr]);
(gdb) n
Hello World!
50              cntr++;
(gdb)
```

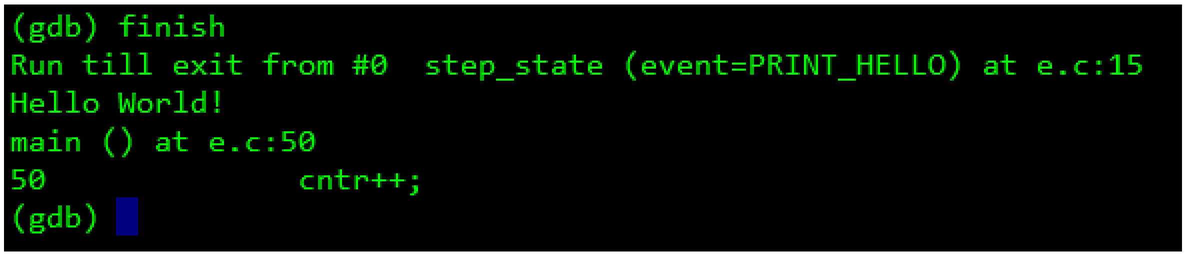
c. How will you get inside the function without using breakpoint?



```
(gdb) step
47          while(events_arr[cntr] != STOP_LOOPING)
(gdb) step

Breakpoint 1, main () at e.c:49
49              step_state(events_arr[cntr]);
(gdb) step
step_state (event=PRINT_HELLO) at e.c:15
15          switch(state) {
```

by using step function after using a breakpoint at function call we can step into a function without using a breakpoint inside the function itself.

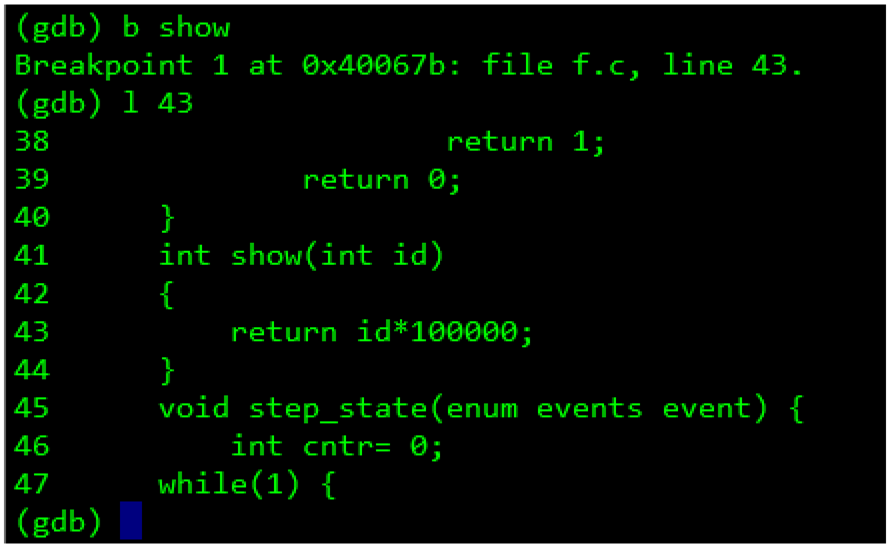
d. How will you come out the of the function without using next and continue?



```
(gdb) finish
Run till exit from #0  step_state (event=PRINT_HELLO) at e.c:15
Hello World!
main () at e.c:50
50                cntr++;
(gdb)
```

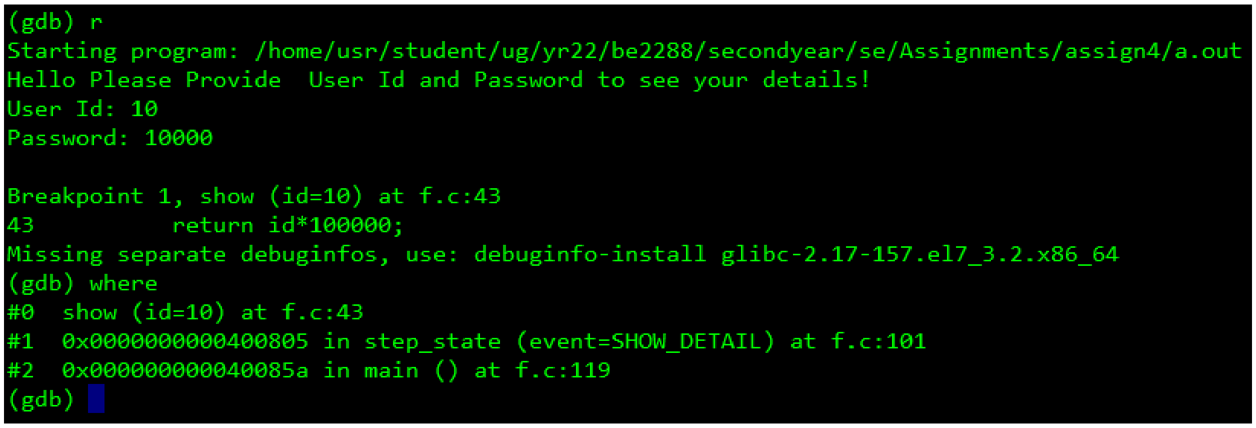
finish is used to step-out (exit from within function)
next is used to step-over(doesn't enter function body)
step is used to step-into (enters function body)

2. Consider the program in Assign4 .It is also a simple state machine. If you provide user id and password properly account details will be displayed. The basic rule is user id should be positive and less than 20 .password is userid *b1000 .The loop will terminate after 10 iteration. It works fine if you provide valid user id and password. It works fine for invalid userid. But it goes to infiniteloop for invalid password. Run the program .It goes into infinite loop. You need to kill the program by [ctrl^c]
   a. Set a suitable breakpoint in gdb in the routine show. give valid input and run



```
(gdb) b show
Breakpoint 1 at 0x40067b: file f.c, line 43.
(gdb) l 43
38                              return 1;
39                      return 0;
40              }
41              int show(int id)
42              {
43                  return id*100000;
44              }
45              void step_state(enum events event) {
46                  int cntr= 0;
47              while(1) {
(gdb)
```

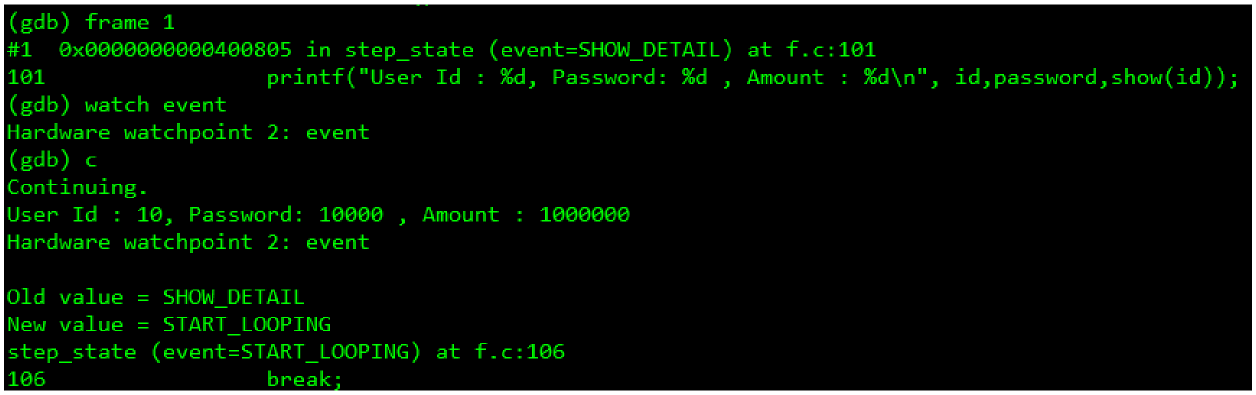
   b. How you can see the call stack of the routine.



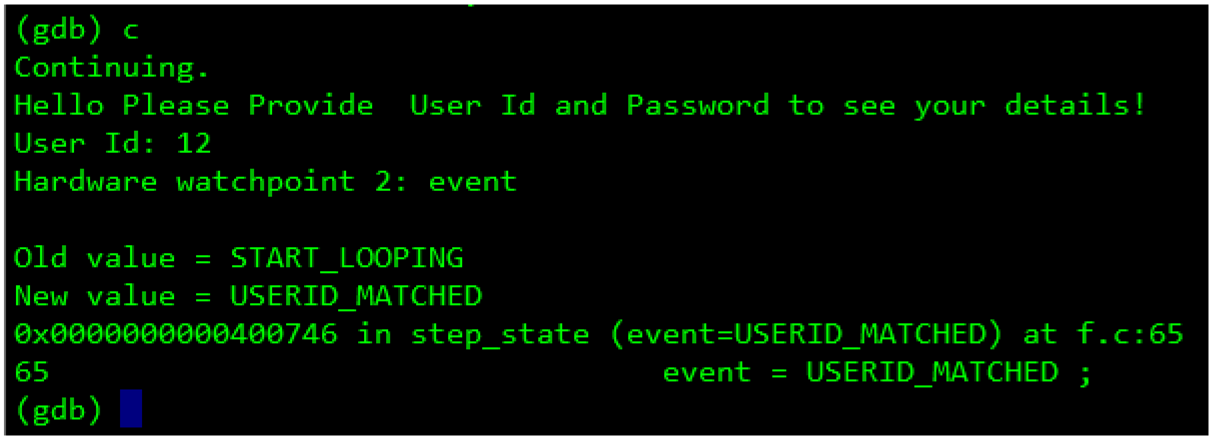
```
(gdb) r
Starting program: /home/usr/student/ug/yr22/be2288/secondyear/se/Assignments/assign4/a.out
Hello Please Provide  User Id and Password to see your details!
User Id: 10
Password: 10000

Breakpoint 1, show (id=10) at f.c:43
43          return id*100000;
Missing separate debuginfos, use: debuginfo-install glibc-2.17-157.el7_3.2.x86_64
(gdb) where
#0  show (id=10) at f.c:43
#1  0x0000000000400805 in step_state (event=SHOW_DETAIL) at f.c:101
#2  0x000000000040085a in main () at f.c:119
(gdb)
```

this means that main() called step_state function with parameter event=SHOW_DETAIL which made another function call for the show(id = 10) function.

c. Which commands will help you to see each value change of variable “event”?

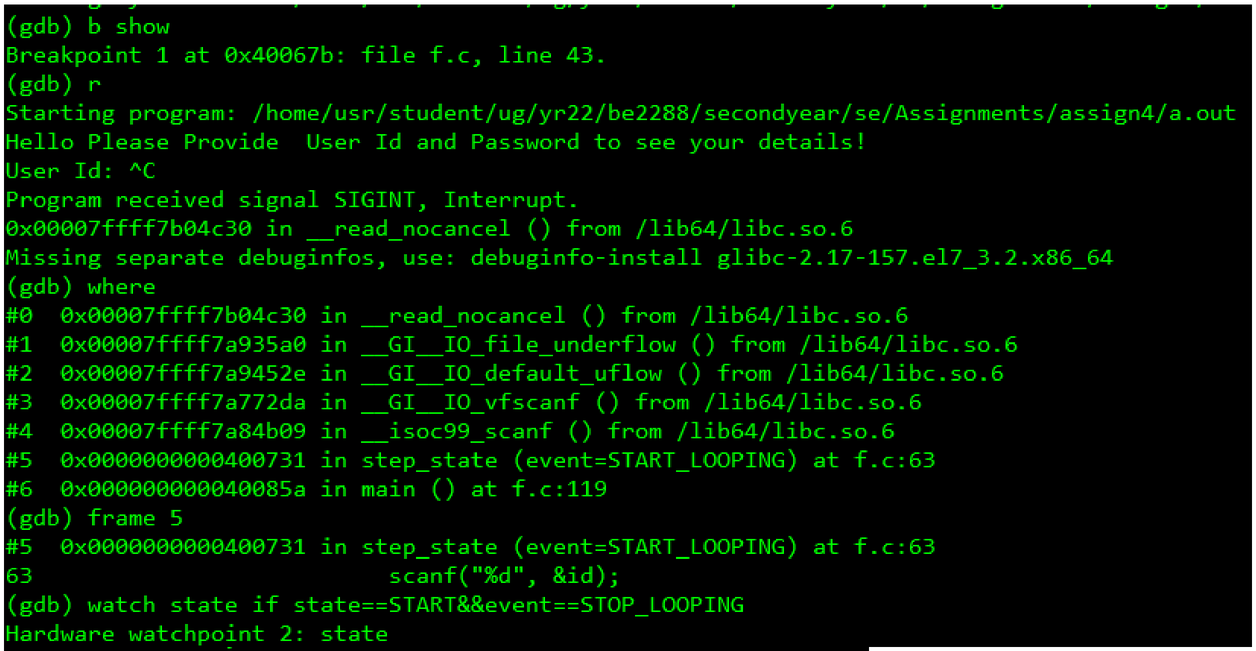


```
(gdb) frame 1
#1  0x0000000000400805 in step_state (event=SHOW_DETAIL) at f.c:101
101                     printf("User Id : %d, Password: %d , Amount : %d\n", id,password,show(id));
(gdb) watch event
Hardware watchpoint 2: event
(gdb) c
Continuing.
User Id : 10, Password: 10000 , Amount : 1000000
Hardware watchpoint 2: event

Old value = SHOW_DETAIL
New value = START_LOOPING
step_state (event=START_LOOPING) at f.c:106
106                     break;
```

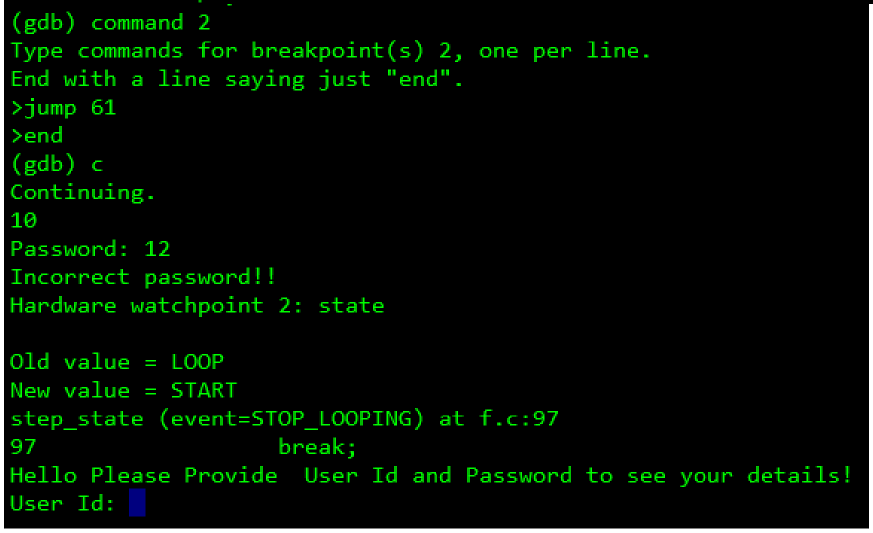


```
(gdb) c
Continuing.
Hello Please Provide  User Id and Password to see your details!
User Id: 12
Hardware watchpoint 2: event

Old value = START_LOOPING
New value = USERID_MATCHED
0x0000000000400746 in step_state (event=USERID_MATCHED) at f.c:65
65                                  event = USERID_MATCHED ;
(gdb)
```

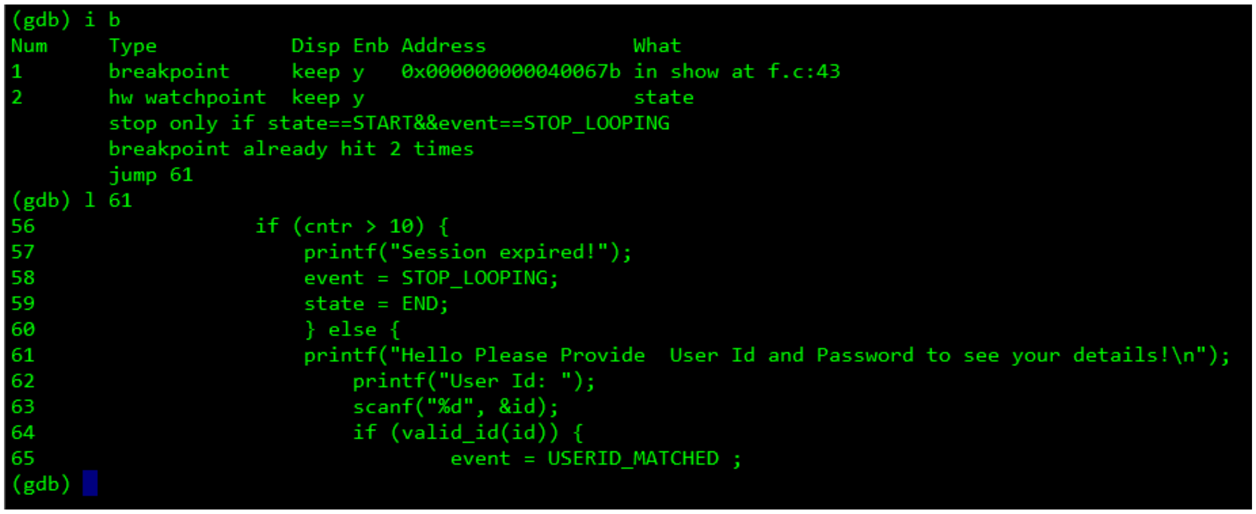
d. Correct the program so that it doesn't go to infinite loop for wrong password. Rather main iteration restarts . [follow the value change path of event for wrong password]



```
(gdb) b show
Breakpoint 1 at 0x40067b: file f.c, line 43.
(gdb) r
Starting program: /home/usr/student/ug/yr22/be2288/secondyear/se/Assignments/assign4/a.out
Hello Please Provide  User Id and Password to see your details!
User Id: ^C
Program received signal SIGINT, Interrupt.
0x00007ffff7b04c30 in __read_nocancel () from /lib64/libc.so.6
Missing separate debuginfos, use: debuginfo-install glibc-2.17-157.el7_3.2.x86_64
(gdb) where
#0  0x00007ffff7b04c30 in __read_nocancel () from /lib64/libc.so.6
#1  0x00007ffff7a935a0 in __GI__IO_file_underflow () from /lib64/libc.so.6
#2  0x00007ffff7a9452e in __GI__IO_default_uflow () from /lib64/libc.so.6
#3  0x00007ffff7a772da in __GI__IO_vfscanf () from /lib64/libc.so.6
#4  0x00007ffff7a84b09 in __isoc99_scanf () from /lib64/libc.so.6
#5  0x0000000000400731 in step_state (event=START_LOOPING) at f.c:63
#6  0x000000000040085a in main () at f.c:119
(gdb) frame 5
#5  0x0000000000400731 in step_state (event=START_LOOPING) at f.c:63
63                              scanf("%d", &id);
(gdb) watch state if state==START&&event==STOP_LOOPING
Hardware watchpoint 2: state
```



```
(gdb) command 2
Type commands for breakpoint(s) 2, one per line.
End with a line saying just "end".
>jump 61
>end
(gdb) c
Continuing.
10
Password: 12
Incorrect password!!
Hardware watchpoint 2: state

Old value = LOOP
New value = START
step_state (event=STOP_LOOPING) at f.c:97
97                      break;
Hello Please Provide  User Id and Password to see your details!
User Id: 
```



```
(gdb) i b
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x000000000040067b in show at f.c:43
2       hw watchpoint  keep y                      state
        stop only if state==START&&event==STOP_LOOPING
        breakpoint already hit 2 times
        jump 61
(gdb) l 61
56                  if (cntr > 10) {
57                      printf("Session expired!");
58                      event = STOP_LOOPING;
59                      state = END;
60                      } else {
61                      printf("Hello Please Provide  User Id and Password to see your details!\n");
62                          printf("User Id: ");
63                          scanf("%d", &id);
64                          if (valid_id(id)) {
65                                  event = USERID_MATCHED ;
(gdb) 
```