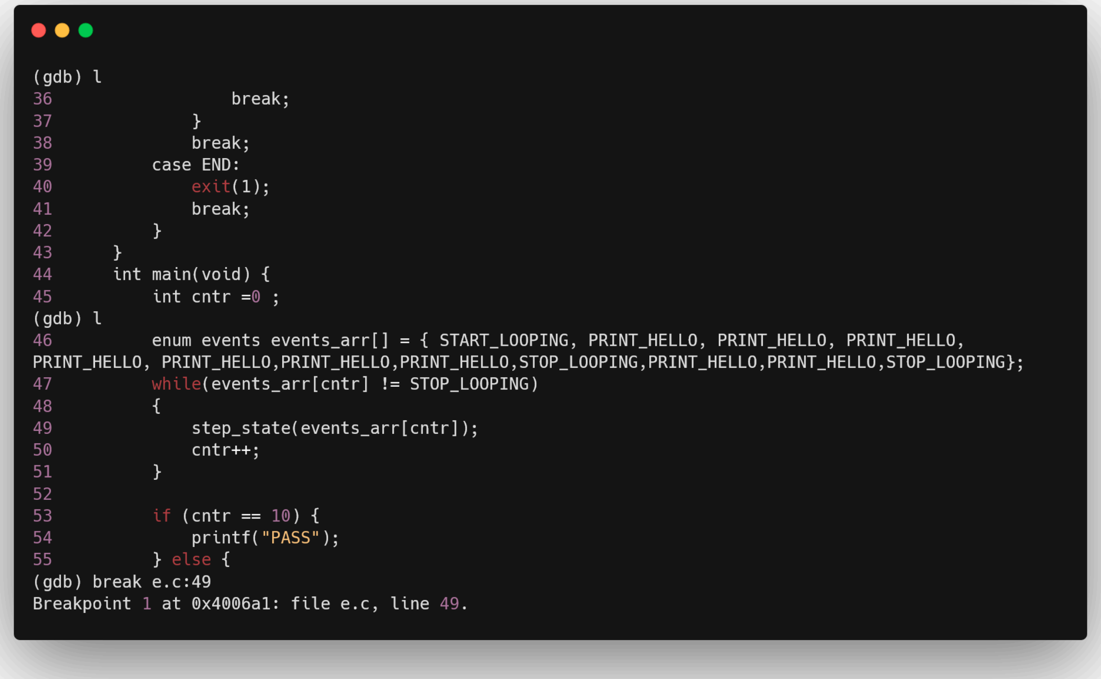
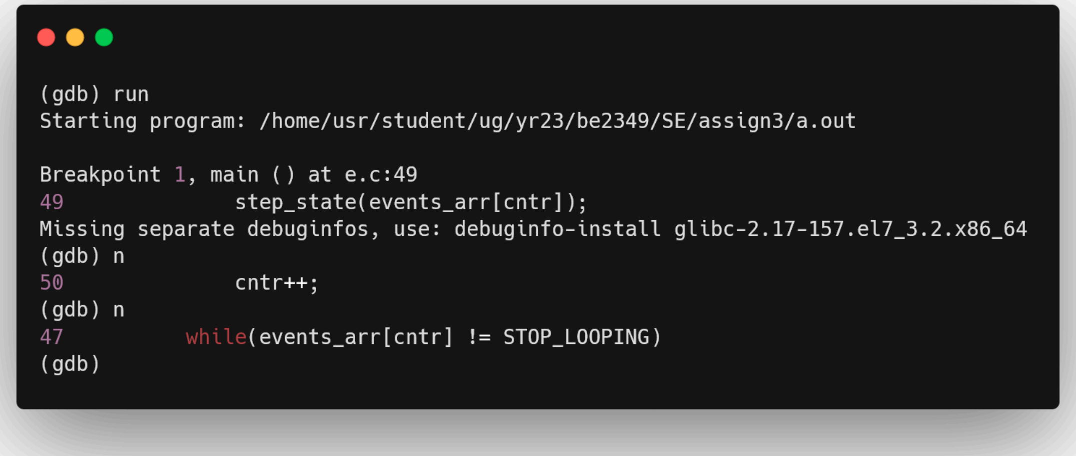
1.

a. Put a breakpoint in line 49



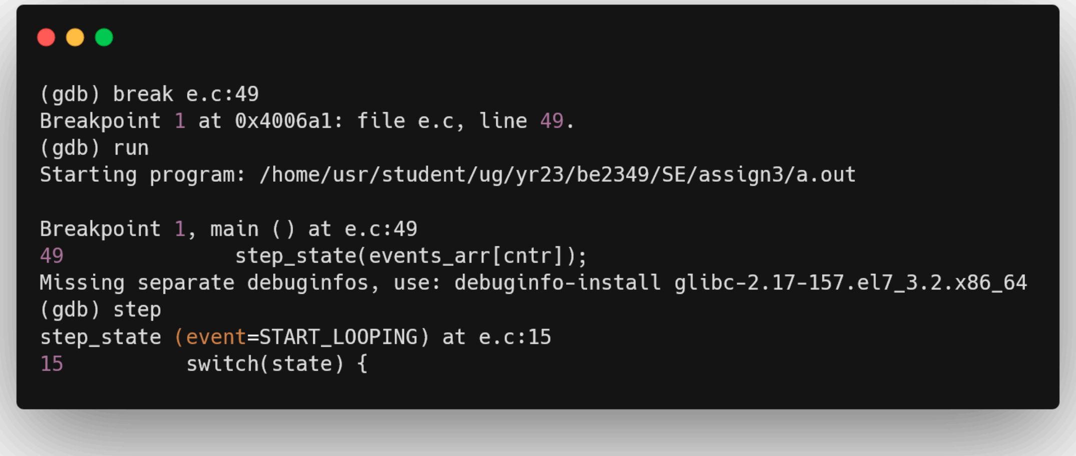
```
(gdb) l
36                      break;
37                  }
38                  break;
39              case END:
40                  exit(1);
41                  break;
42              }
43          }
44          int main(void) {
45              int cntr =0 ;
(gdb) l
46              enum events events_arr[] = { START_LOOPING, PRINT_HELLO, PRINT_HELLO, PRINT_HELLO,
PRINT_HELLO, PRINT_HELLO,PRINT_HELLO,PRINT_HELLO,STOP_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) {
54                  printf("PASS");
55              } else {
(gdb) break e.c:49
Breakpoint 1 at 0x4006a1: file e.c, line 49.
```

b. Try next command



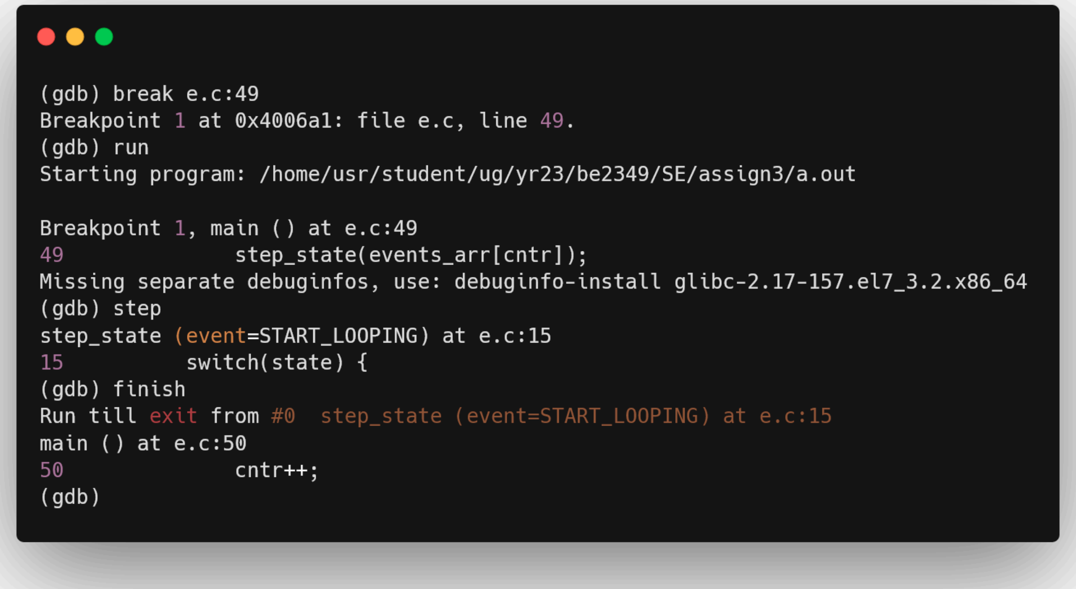
```
(gdb) run
Starting program: /home/usr/student/ug/yr23/be2349/SE/assign3/a.out

Breakpoint 1, main () at e.c:49
49                  step_state(events_arr[cntr]);
Missing separate debuginfos, use: debuginfo-install glibc-2.17-157.el7_3.2.x86_64
(gdb) n
50                  cntr++;
(gdb) n
47              while(events_arr[cntr] != STOP_LOOPING)
(gdb)
```

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



```
(gdb) break e.c:49
Breakpoint 1 at 0x4006a1: file e.c, line 49.
(gdb) run
Starting program: /home/usr/student/ug/yr23/be2349/SE/assign3/a.out

Breakpoint 1, main () at e.c:49
49              step_state(events_arr[cntr]);
Missing separate debuginfos, use: debuginfo-install glibc-2.17-157.el7_3.2.x86_64
(gdb) step
step_state (event=START_LOOPING) at e.c:15
15          switch(state) {
```

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

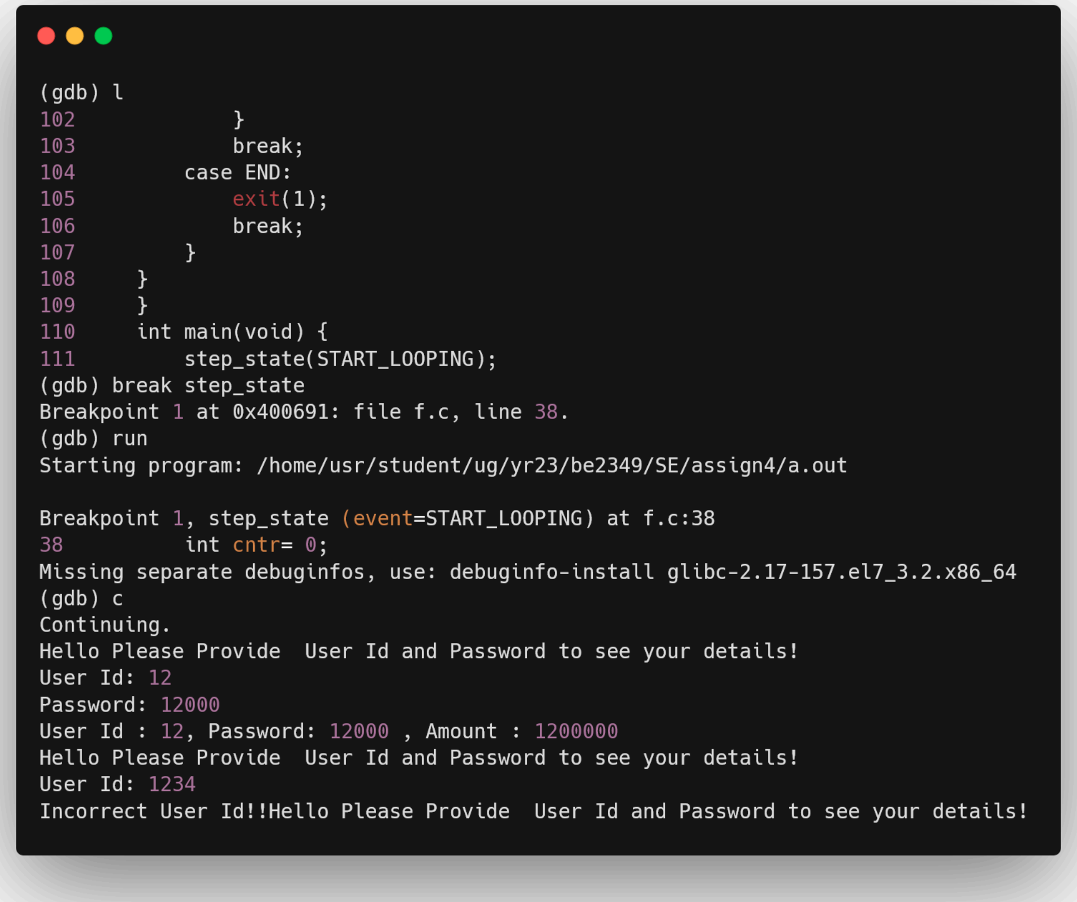


```
(gdb) break e.c:49
Breakpoint 1 at 0x4006a1: file e.c, line 49.
(gdb) run
Starting program: /home/usr/student/ug/yr23/be2349/SE/assign3/a.out

Breakpoint 1, main () at e.c:49
49              step_state(events_arr[cntr]);
Missing separate debuginfos, use: debuginfo-install glibc-2.17-157.el7_3.2.x86_64
(gdb) step
step_state (event=START_LOOPING) at e.c:15
15          switch(state) {
(gdb) finish
Run till exit from #0  step_state (event=START_LOOPING) at e.c:15
main () at e.c:50
50              cntr++;
(gdb)
```

2.

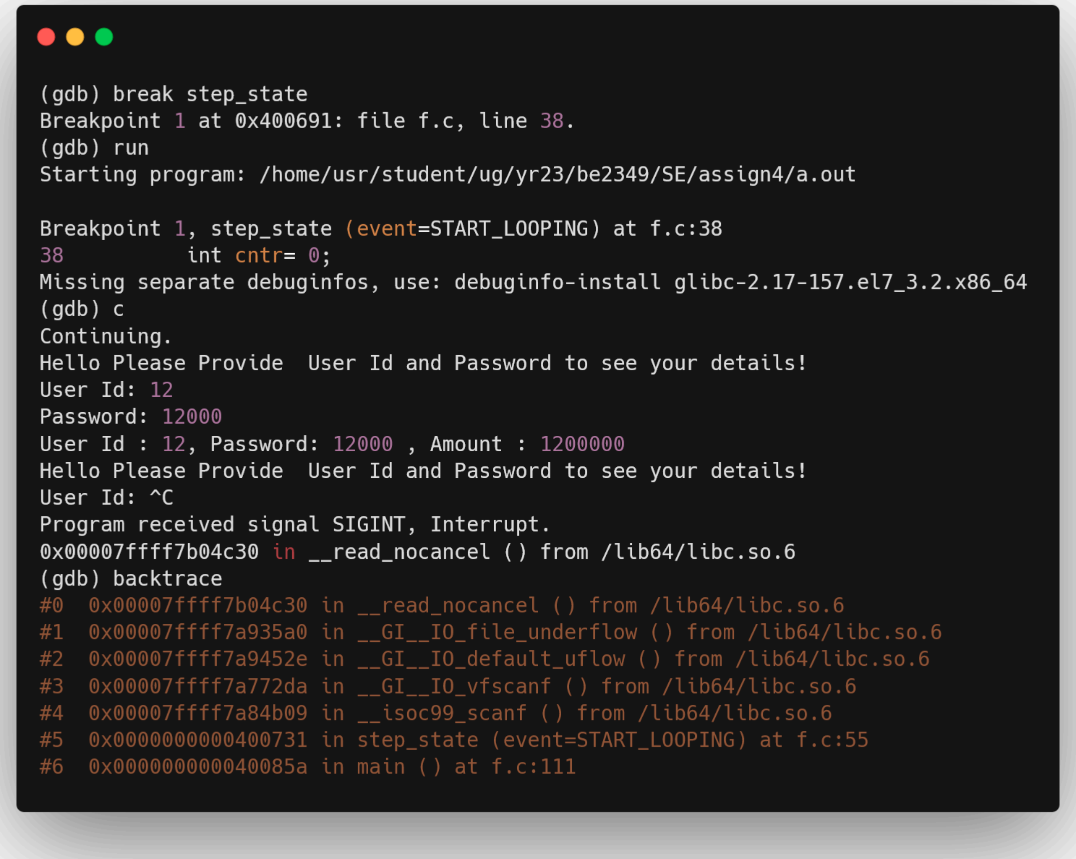
a. Set a suitable breakpoint in gdb in the routine

show.give valid input and run :

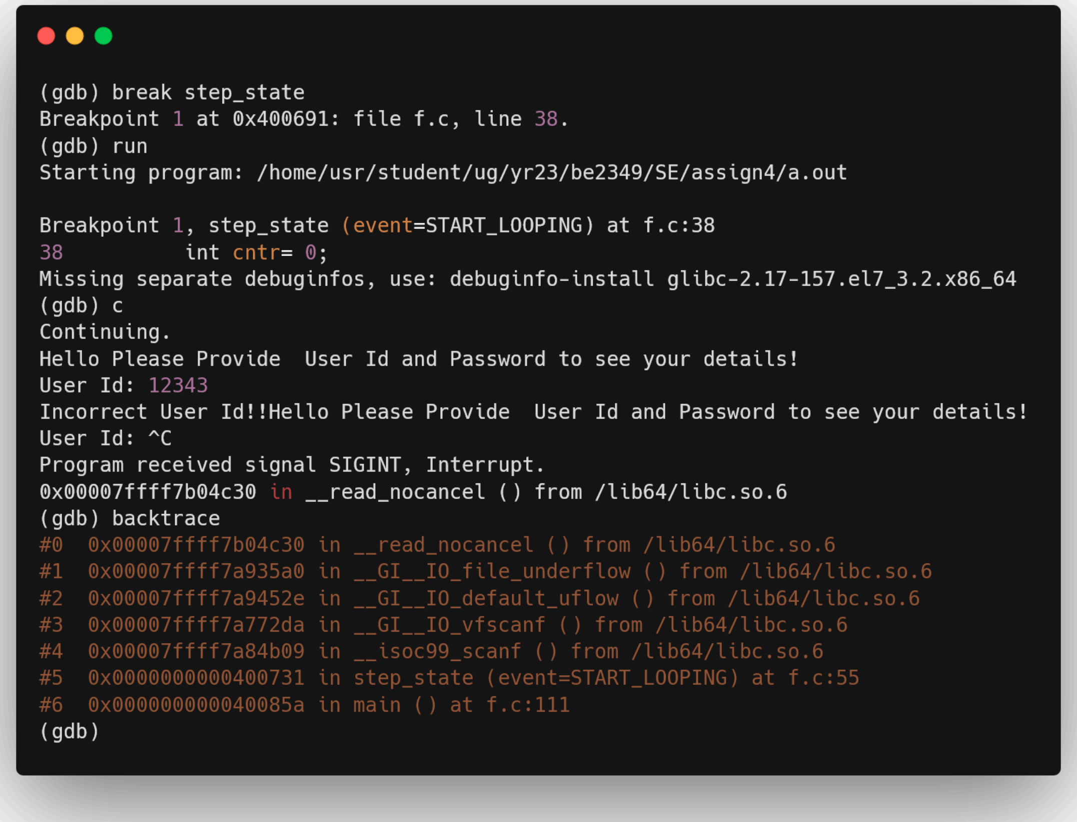


```
(gdb) l
102             }
103             break;
104         case END:
105             exit(1);
106             break;
107         }
108     }
109     }
110     int main(void) {
111         step_state(START_LOOPING);
(gdb) break step_state
Breakpoint 1 at 0x400691: file f.c, line 38.
(gdb) run
Starting program: /home/usr/student/ug/yr23/be2349/SE/assign4/a.out

Breakpoint 1, step_state (event=START_LOOPING) at f.c:38
38          int cntr= 0;
Missing separate debuginfos, use: debuginfo-install glibc-2.17-157.el7_3.2.x86_64
(gdb) c
Continuing.
Hello Please Provide  User Id and Password to see your details!
User Id: 12
Password: 12000
User Id : 12, Password: 12000 , Amount : 1200000
Hello Please Provide  User Id and Password to see your details!
User Id: 1234
Incorrect User Id!!Hello Please Provide  User Id and Password to see your details!
```

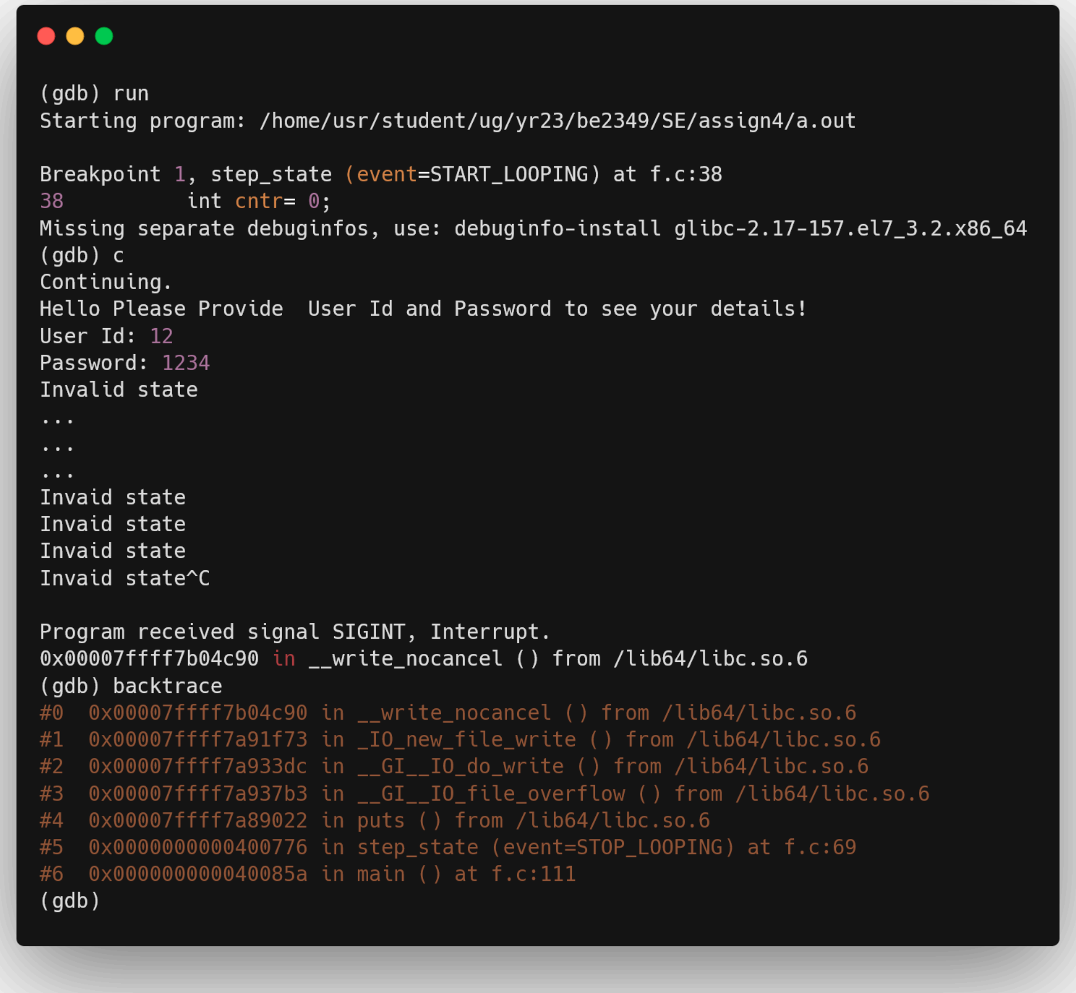
b. How you can see the call stack of the routine.



```
(gdb) break step_state
Breakpoint 1 at 0x400691: file f.c, line 38.
(gdb) run
Starting program: /home/usr/student/ug/yr23/be2349/SE/assign4/a.out

Breakpoint 1, step_state (event=START_LOOPING) at f.c:38
38          int cntr= 0;
Missing separate debuginfos, use: debuginfo-install glibc-2.17-157.el7_3.2.x86_64
(gdb) c
Continuing.
Hello Please Provide  User Id and Password to see your details!
User Id: 12
Password: 12000
User Id : 12, Password: 12000 , Amount : 1200000
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
(gdb) backtrace
#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:55
#6  0x000000000040085a in main () at f.c:111
```



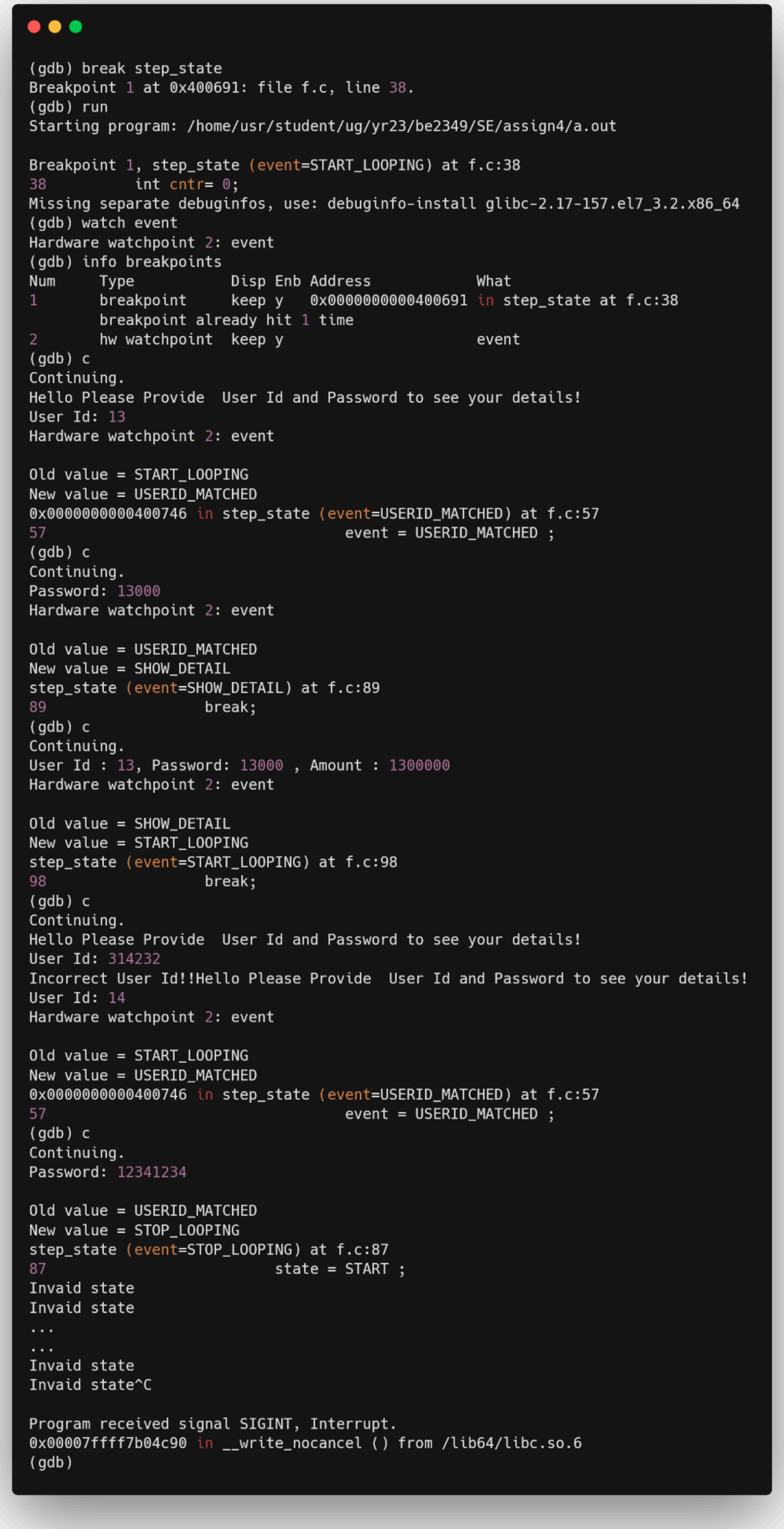
```
(gdb) break step_state
Breakpoint 1 at 0x400691: file f.c, line 38.
(gdb) run
Starting program: /home/usr/student/ug/yr23/be2349/SE/assign4/a.out

Breakpoint 1, step_state (event=START_LOOPING) at f.c:38
38          int cntr= 0;
Missing separate debuginfos, use: debuginfo-install glibc-2.17-157.el7_3.2.x86_64
(gdb) c
Continuing.
Hello Please Provide  User Id and Password to see your details!
User Id: 12343
Incorrect User Id!!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
(gdb) backtrace
#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:55
#6  0x000000000040085a in main () at f.c:111
(gdb)
```



```
(gdb) run
Starting program: /home/usr/student/ug/yr23/be2349/SE/assign4/a.out

Breakpoint 1, step_state (event=START_LOOPING) at f.c:38
38          int cntr= 0;
Missing separate debuginfos, use: debuginfo-install glibc-2.17-157.el7_3.2.x86_64
(gdb) c
Continuing.
Hello Please Provide  User Id and Password to see your details!
User Id: 12
Password: 1234
Invalid state
...
...
...
Invaid state
Invaid state
Invaid state
Invaid state^C

Program received signal SIGINT, Interrupt.
0x00007ffff7b04c90 in __write_nocancel () from /lib64/libc.so.6
(gdb) backtrace
#0  0x00007ffff7b04c90 in __write_nocancel () from /lib64/libc.so.6
#1  0x00007ffff7a91f73 in _IO_new_file_write () from /lib64/libc.so.6
#2  0x00007ffff7a933dc in __GI__IO_do_write () from /lib64/libc.so.6
#3  0x00007ffff7a937b3 in __GI__IO_file_overflow () from /lib64/libc.so.6
#4  0x00007ffff7a89022 in puts () from /lib64/libc.so.6
#5  0x0000000000400776 in step_state (event=STOP_LOOPING) at f.c:69
#6  0x000000000040085a in main () at f.c:111
(gdb)
```

c. Which commands will help you to see each value

change of variable “event”?

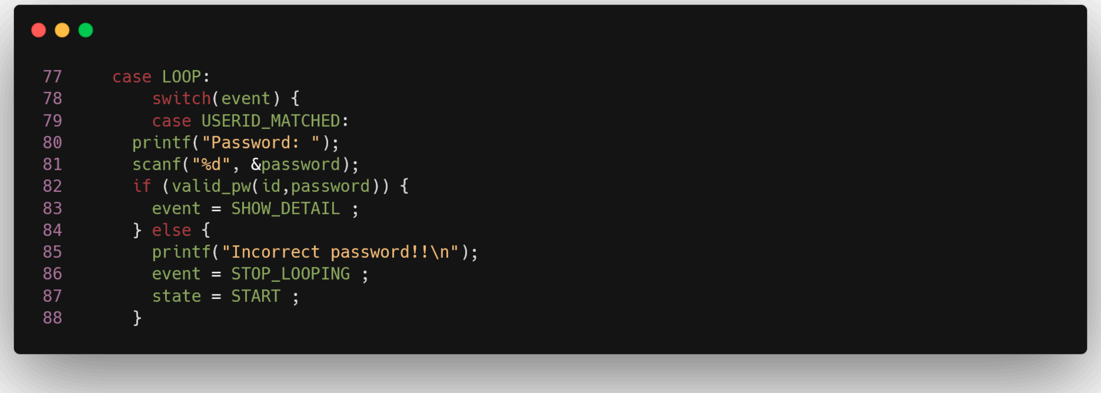


```
(gdb) break step_state
Breakpoint 1 at 0x400691: file f.c, line 38.
(gdb) run
Starting program: /home/usr/student/ug/yr23/be2349/SE/assign4/a.out

Breakpoint 1, step_state (event=START_LOOPING) at f.c:38
38          int cntr= 0;
Missing separate debuginfos, use: debuginfo-install glibc-2.17-157.el7_3.2.x86_64
(gdb) watch event
Hardware watchpoint 2: event
(gdb) info breakpoints
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000000000400691 in step_state at f.c:38
        breakpoint already hit 1 time
2       hw watchpoint  keep y                      event
(gdb) c
Continuing.
Hello Please Provide  User Id and Password to see your details!
User Id: 13
Hardware watchpoint 2: event

Old value = START_LOOPING
New value = USERID_MATCHED
0x0000000000400746 in step_state (event=USERID_MATCHED) at f.c:57
57                                  event = USERID_MATCHED ;
(gdb) c
Continuing.
Password: 13000
Hardware watchpoint 2: event

Old value = USERID_MATCHED
New value = SHOW_DETAIL
step_state (event=SHOW_DETAIL) at f.c:89
89                  break;
(gdb) c
Continuing.
User Id : 13, Password: 13000 , Amount : 1300000
Hardware watchpoint 2: event

Old value = SHOW_DETAIL
New value = START_LOOPING
step_state (event=START_LOOPING) at f.c:98
98                  break;
(gdb) c
Continuing.
Hello Please Provide  User Id and Password to see your details!
User Id: 314232
Incorrect User Id!!Hello Please Provide  User Id and Password to see your details!
User Id: 14
Hardware watchpoint 2: event

Old value = START_LOOPING
New value = USERID_MATCHED
0x0000000000400746 in step_state (event=USERID_MATCHED) at f.c:57
57                                  event = USERID_MATCHED ;
(gdb) c
Continuing.
Password: 12341234

Old value = USERID_MATCHED
New value = STOP_LOOPING
step_state (event=STOP_LOOPING) at f.c:87
87                          state = START ;
Invaid state
Invaid state
...
...
Invaid state
Invaid state^C

Program received signal SIGINT, Interrupt.
0x00007ffff7b04c90 in __write_nocancel () from /lib64/libc.so.6
(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

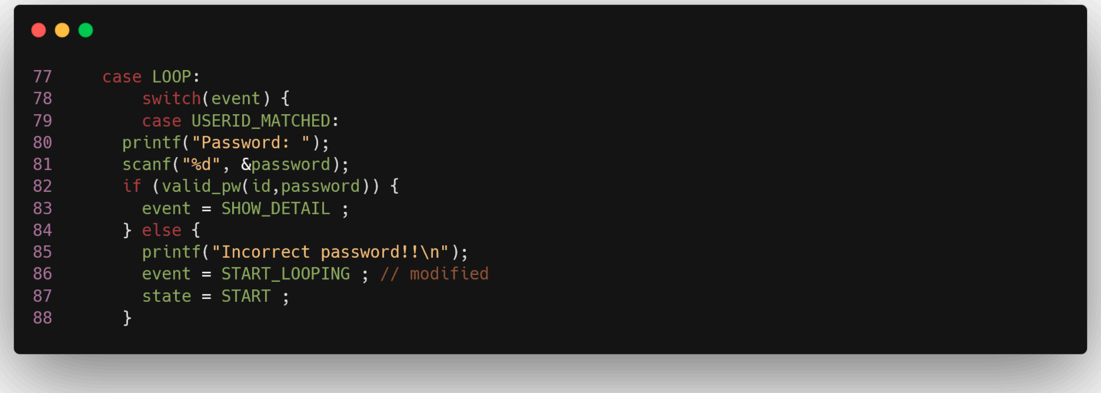
of event for wrong password]

Code snippet causing infinite loop on wrong password



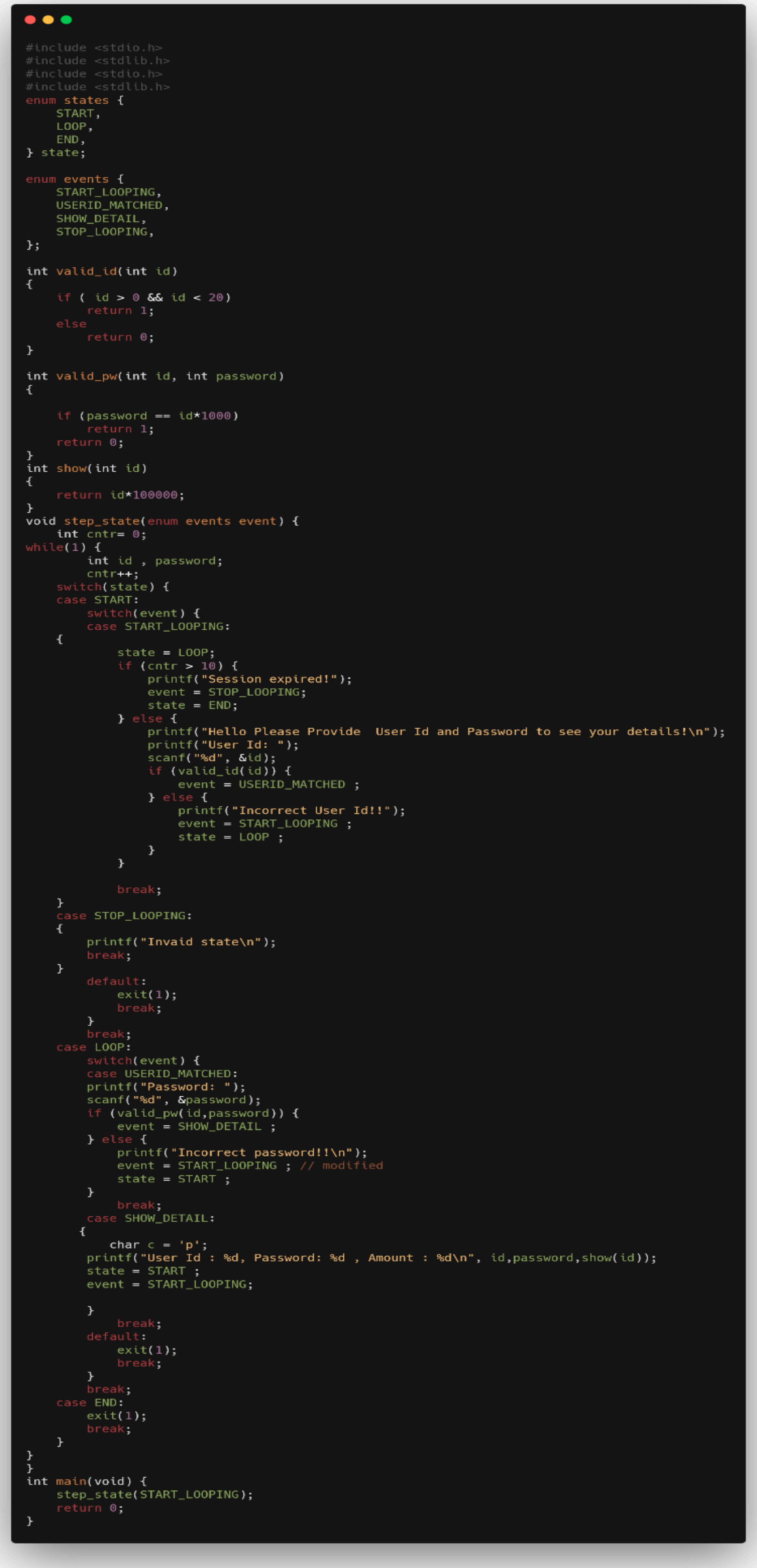
```
    case LOOP:
        switch(event) {
        case USERID_MATCHED:
      printf("Password: ");
      scanf("%d", &password);
      if (valid_pw(id,password)) {
        event = SHOW_DETAIL ;
      } else {
        printf("Incorrect password!!\n");
        event = STOP_LOOPING ;
        state = START ;
      }
```

modified



```
    case LOOP:
        switch(event) {
        case USERID_MATCHED:
      printf("Password: ");
      scanf("%d", &password);
      if (valid_pw(id,password)) {
        event = SHOW_DETAIL ;
      } else {
        printf("Incorrect password!!\n");
        event = START_LOOPING ; // modified
        state = START ;
      }
```

Modified program



```
#include <stdio.h>
#include <stdlib.h>
#include <stdio.h>
#include <stdlib.h>
enum states {
    START,
    LOOP,
    END,
} state;

enum events {
    START_LOOPING,
    USERID_MATCHED,
    SHOW_DETAIL,
    STOP_LOOPING,
};

int valid_id(int id)
{
    if ( id > 0 && id < 20)
        return 1;
    else
        return 0;
}

int valid_pw(int id, int password)
{

    if (password == id*1000)
        return 1;
    return 0;
}
int show(int id)
{
    return id*100000;
}
void step_state(enum events event) {
    int cntr= 0;
while(1) {
        int id , password;
        cntr++;
    switch(state) {
    case START:
        switch(event) {
        case START_LOOPING:
    {
            state = LOOP;
            if (cntr > 10) {
                printf("Session expired!");
                event = STOP_LOOPING;
                state = END;
            } else {
                printf("Hello Please Provide  User Id and Password to see your details!\n");
                printf("User Id: ");
                scanf("%d", &id);
                if (valid_id(id)) {
                    event = USERID_MATCHED ;
                } else {
                    printf("Incorrect User Id!!");
                    event = START_LOOPING ;
                    state = LOOP ;
                }
            }

            break;
    }
    case STOP_LOOPING:
    {
        printf("Invaid state\n");
        break;
    }
        default:
            exit(1);
            break;
        }
        break;
    case LOOP:
        switch(event) {
        case USERID_MATCHED:
        printf("Password: ");
        scanf("%d", &password);
        if (valid_pw(id,password)) {
            event = SHOW_DETAIL ;
        } else {
            printf("Incorrect password!!\n");
            event = START_LOOPING ; // modified
            state = START ;
        }
            break;
        case SHOW_DETAIL:
       {
           char c = 'p';
        printf("User Id : %d, Password: %d , Amount : %d\n", id,password,show(id));
        state = START ;
        event = START_LOOPING;

        }
            break;
        default:
            exit(1);
            break;
        }
        break;
    case END:
        exit(1);
        break;
    }
}
}
int main(void) {
    step_state(START_LOOPING);
    return 0;
}
```

Explore the commands found for 5c to see/use

content of a pointer