Assignments-3 on gdb

Name:- ATRIJ ROY

Roll no:- 002311001086

Class:- IT UG2 A3

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

```
(gdb) break e.c:49
Breakpoint 1 at 0x140001525: file e.c, line 49.
```

   b. Try next command

```
(gdb) run
Starting program: C:\Users\acer\Documents\atr86\prog.exe
[New Thread 4708.0x638]

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

Thread 1 hit Breakpoint 1, main () at e.c:49
49              step_state(events_arr[cntr]);
(gdb)
```

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

      Using the **step** command, we can go to the next line and step into function calls(if any function is called), in contrast to **next**, which

goes to the next line by stepping over the function call (i.e. after the function call executes),without setting a breakpoint inside the function.

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

Thread 1 hit 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) {
(gdb) step
27              switch(event) {
(gdb)
```

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

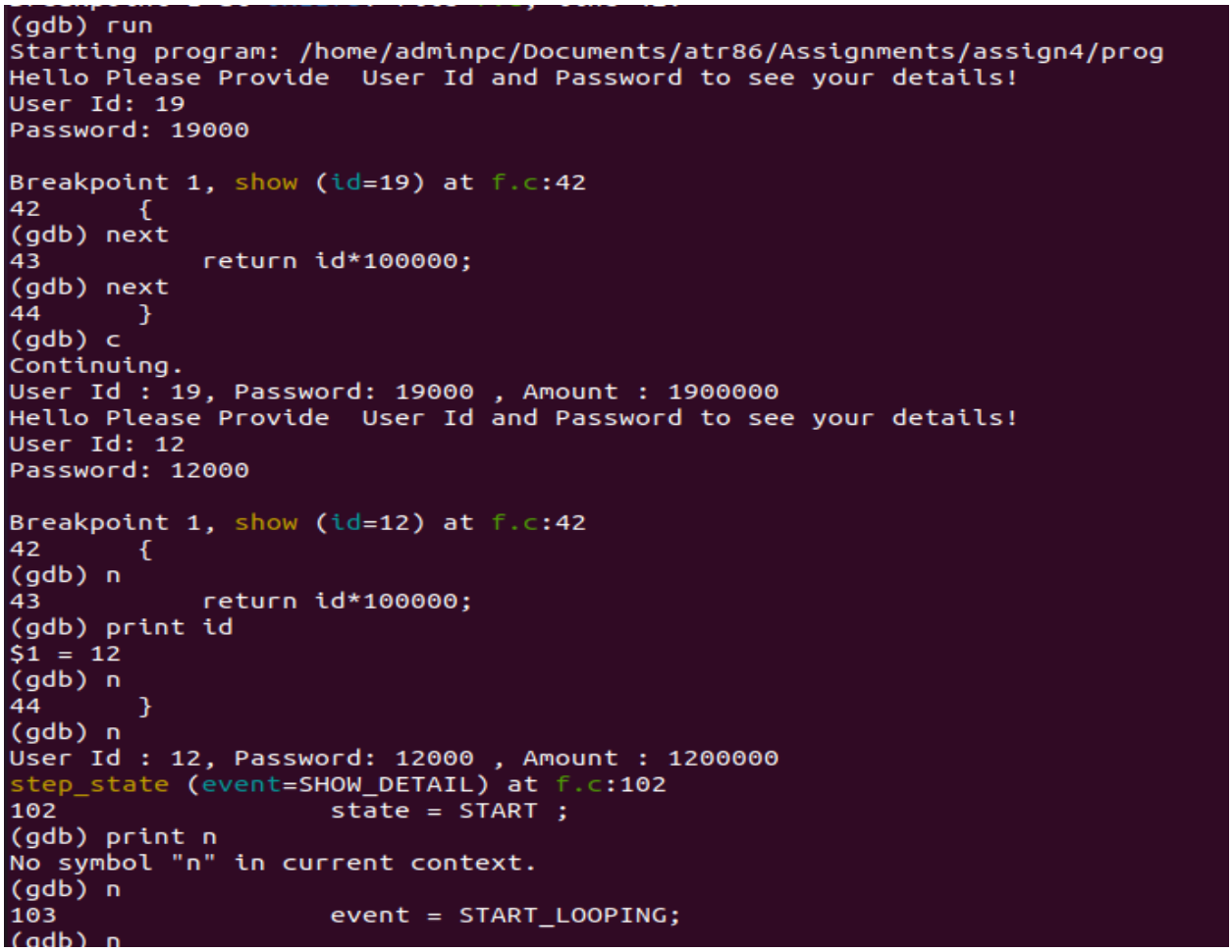
e.

```
Thread 1 hit 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) {
(gdb) step
27              switch(event) {
(gdb) finish
Run till exit from #0  step_state (event=PRINT_HELLO) at e.c:27
Hello World!
main () at e.c:50
50              cntr++;
(gdb) n
```

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 *1000 .The loop will terminate after 10 iterations. It works fine if you provide valid user id and password.It works fine for invalid userid. But it goes to an infinite loop 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 :



```
103                 event = START_LOOPING;
(gdb) n
106                 break;
(gdb) c
Continuing.
Hello Please Provide  User Id and Password to see your details!
User Id: 13
Password: 13000

Breakpoint 1, show (id=13) at f.c:42
42      {
(gdb) c
Continuing.
User Id : 13, Password: 13000 , Amount : 1300000
Hello Please Provide  User Id and Password to see your details!
User Id: 14
Password: 14000

Breakpoint 1, show (id=14) at f.c:42
42      {
```

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

```
Breakpoint 1, show (id=12) at f.c:42
42      {
(gdb) where
#0  show (id=12) at f.c:42
#1  0x00005555555553b9 in step_state (event=SHOW_DETAIL) at f.c:101
#2  0x0000555555555414 in main () at f.c:119
(gdb) backtrace
#0  show (id=12) at f.c:42
#1  0x00005555555553b9 in step_state (event=SHOW_DETAIL) at f.c:101
#2  0x0000555555555414 in main () at f.c:119
(gdb)
```

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

```
(gdb) watch event
Hardware watchpoint 2: event
(gdb) c
Continuing.

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: c

Hardware watchpoint 2: event

Old value = START_LOOPING
New value = USERID_MATCHED
step_state (event=USERID_MATCHED) at f.c:73
73                  break;
(gdb) c
Continuing.

Hardware watchpoint 2: event

Old value = USERID_MATCHED
New value = SHOW_DETAIL
step_state (event=SHOW_DETAIL) at f.c:97
97                  break;
(gdb) c
Continuing.

Breakpoint 1, show (id=12) at f.c:42
42      {
(gdb) c
Continuing.
Password: User Id : 12, Password: 12000 , Amount : 1200000

Hardware watchpoint 2: event

Old value = SHOW_DETAIL
New value = START_LOOPING
```

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

Explore the commands found for 5c to see/use content of a pointer

```
case STOP_LOOPING:
{
    printf("Invaid state\n");
    state = END;
    break;
}
```

```
(gdb) run
Starting program: /home/adminpc/Documents/atr86/Assignments/assign4/prog
Hello Please Provide  User Id and Password to see your details!
User Id: 12
Password: 120000
Incorrect password!!
Invaid state
[Inferior 1 (process 8430) exited with code 01]
(gdb) list f.c:75, f.c:80
75              case STOP_LOOPING:
76              {
77                  printf("Invaid state\n");
78                  state = END;
79                  break;
80              }
(gdb)
```
