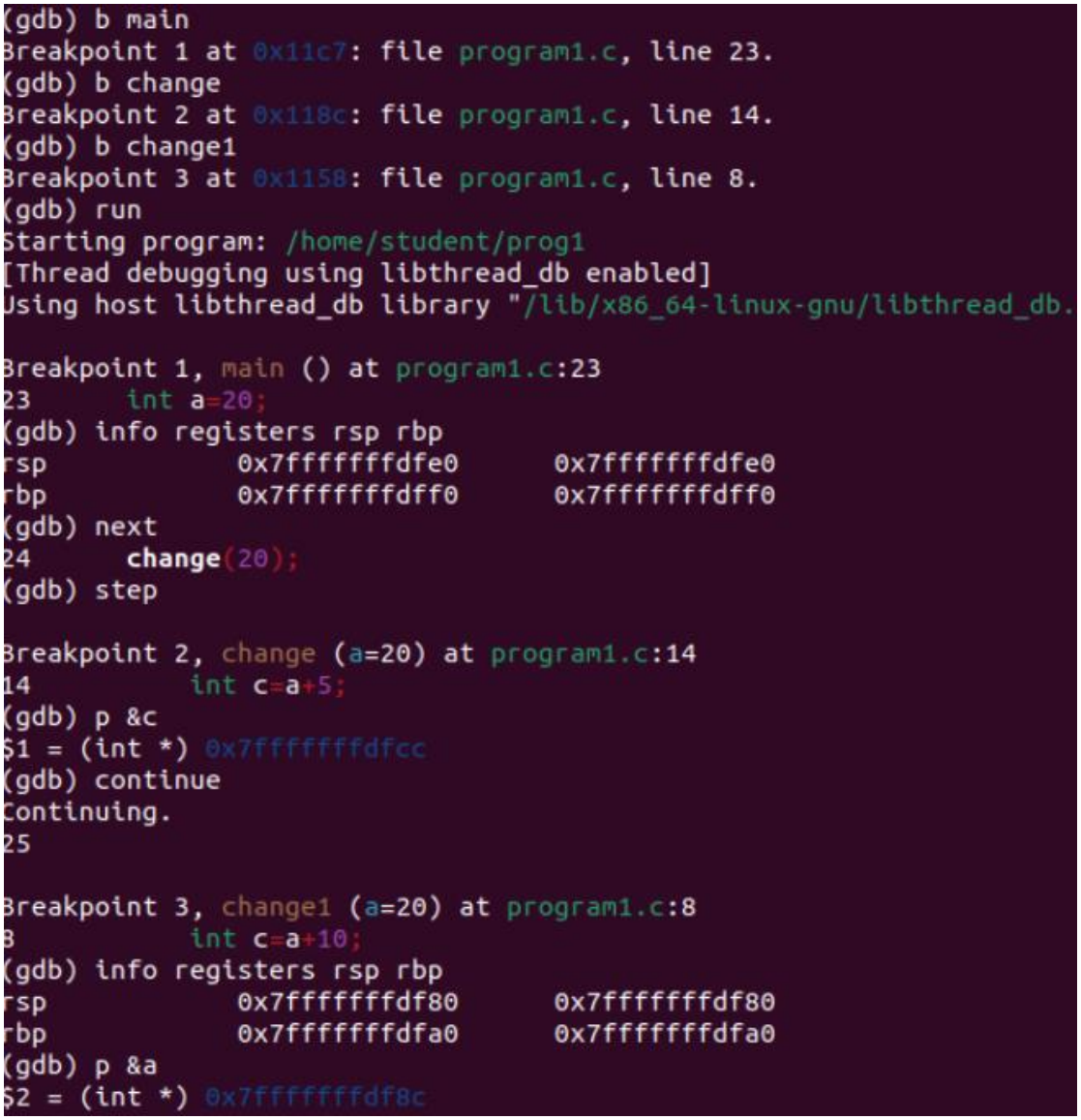
Task 1F-Make two level function calls, with passing of parameters. Observe the address of the stack frames, where the local variables are stored and how the SP and BP registers are manipulated and how the return address is stored on the stack.

```c
// observe layout of data in memory using gdb

#include <stdio.h>
#include <stdlib.h>

void change1(int a)
{
    int c=a+10;
    printf("%d\n",c);
}

void change(int a)
{
    int c=a+5;
    printf("%d\n",c);
    change1(a);
}



int main()
{
int a=20;
change(20);
return 0;
}
```


(gdb) b main
Breakpoint 1 at 0x11c7: file program1.c, line 23.
(gdb) b change
Breakpoint 2 at 0x118c: file program1.c, line 14.
(gdb) b change1
Breakpoint 3 at 0x1158: file program1.c, line 8.
(gdb) run
Starting program: /home/student/prog1
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.
Breakpoint 1, main () at program1.c:23
23 int a=20;
(gdb) info registers rsp rbp
rsp 0x7fffffffdfe0 0x7fffffffdfe0
rbp 0x7fffffffdff0 0x7fffffffdff0
(gdb) next
24 change(20);
(gdb) step
Breakpoint 2, change (a=20) at program1.c:14
14 int c=a+5;
(gdb) p &c
$1 = (int *) 0x7fffffffdfcc
(gdb) continue
Continuing.
25
Breakpoint 3, change1 (a=20) at program1.c:8
8 int c=a+10;
(gdb) info registers rsp rbp
rsp 0x7fffffffdf80 0x7fffffffdf80
rbp 0x7fffffffdfa0 0x7fffffffdfa0
(gdb) p &a
$2 = (int *) 0x7fffffffdf8c