ASSIGNMENT 1

Name : Hrithvik Kondalkar
Roll : 002211001088
Class :IT UG2 - A2

A) Compile it so that it compiles with debugging symbols [using proper option]

```
>gcc -g a.c b.c
>gdb a.out
```

B) Put breakpoint to function f1.

```
>(gdb) b f1
Breakpoint 1 at 0x400756: file b.c, line 4.
```

C) Put breakpoint to line 10 of b.c

```
>(gdb) b b.c:10
Breakpoint 2 at 0x40079e: file b.c, line 10.
```

D) Run the program until it finishes. Which commands are you using to take it to completion?

```
>(gdb) r
Starting program:
/home/usr/student/ug/yr22/be2288/secondyear/se/Assi
gnments/assign1/a.out
Enter a number between 2 and 6 (non-inclusive):
3
You have entered 3

Breakpoint 1, f1 (x=50, y=163) at b.c:4
4       printf("The numbers are : ");
Missing separate debuginfos, use: debuginfo-install glibc-
2.17-157.el7_3.2.x86_64
(gdb) c
Continuing.
The numbers are : < 50, 163>

Breakpoint 2, f2 (p=0x7fffffffe2b4, q=0x7fffffffe2b0) at
b.c:10
10      *q = (*p) -(*q);
(gdb) c
Continuing.

Breakpoint 1, f1 (x=163, y=50) at b.c:4
4       printf("The numbers are : ");
(gdb) c
Continuing.
After operation 1 The numbers are : < 163, 50>

Breakpoint 1, f1 (x=33, y=109) at b.c:4
4       printf("The numbers are : ");
(gdb) c
Continuing.
The numbers are : < 33, 109>
```

```
Breakpoint 2, f2 (p=0x7fffffffe2b4, q=0x7fffffffe2b0) at
b.c:10
10      *q = (*p) -(*q);
(gdb) c
Continuing.

Breakpoint 1, f1 (x=109, y=33) at b.c:4
4       printf("The numbers are : ");
(gdb) c
Continuing.
After operation 2 The numbers are : < 109, 33>

Breakpoint 1, f1 (x=25, y=81) at b.c:4
4       printf("The numbers are : ");
(gdb) c
Continuing.
The numbers are : < 25, 81>

Breakpoint 2, f2 (p=0x7fffffffe2b4, q=0x7fffffffe2b0) at
b.c:10
10      *q = (*p) -(*q);
(gdb) c
Continuing.

Breakpoint 1, f1 (x=81, y=25) at b.c:4
4       printf("The numbers are : ");
(gdb) c
Continuing.
After operation 3 The numbers are : < 81, 25>
[Inferior 1 (process 4401) exited with code 03]
```

ASSIGNMENT 1

E) How many times breakpoint “1” is hit in one run of the program?
F) How many times breakpoint “2” is hit in one run of the program?
G) How you can see details about a breakpoint ?
H) How you can see details about all breakpoints ?

```
(gdb) info b
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000000000400756 in f1 at b.c:4
        breakpoint already hit 6 times
2       breakpoint     keep y   0x000000000040079e in f2 at b.c:10
        breakpoint already hit 3 times

(gdb) info b 1
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000000000400756 in f1 at b.c:4
        breakpoint already hit 6 times
  (gdb) info b 2
Num     Type           Disp Enb Address            What
2       breakpoint     keep y   0x000000000040079e in f2 at b.c:10
        breakpoint already hit 3 times
```

I) What is value of variable x in f1 when breakpoint “1” is hit for 3 rd time ? How you can examine it ?

```
(gdb) ignore 1 2
Will ignore next 2 crossings of breakpoint 1.
 (gdb) info b 1
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000000000400756 in f1 at b.c:4
        breakpoint already hit 6 times
        ignore next 2 hits
(gdb) r
Starting program:
/home/usr/student/ug/yr22/be2288/secondyear/se/Assignments/assign1/a.out
Enter a number between 2 and 6 (non-inclusive):
4
You have entered 4
The numbers are : < 50, 163>
After operation 1 The numbers are : < 163, 50>

Breakpoint 1, f1 (x=33, y=109) at b.c:4
4           printf("The numbers are : ");
(gdb) print x
$1 = 33
```

ASSIGNMENT 1

J) Rerun the program. Put a breakpoint at function f0. List 5 lines where it has stopped with breakpoint 3 for first time.

```
gdb) b f0
Breakpoint 1 at 0x4005f9: file a.c, line 6.
  (gdb) set listsize 5
  (gdb) r
Starting program:
/home/usr/student/ug/yr22/be2288/secondyear/se/Assignments/assign1/a.out

Breakpoint 1, f0 (p=0x7fffffffe2b8) at a.c:6
6         int x, cntr = 1;
(gdb) l
4      int f0(int *p)
5      {
6         int x, cntr = 1;
7         printf("Enter a number between 2 and 6 (non-inclusive): \n");
8         scanf("%d", &x);
```