NAME: K SAI KRISHNA

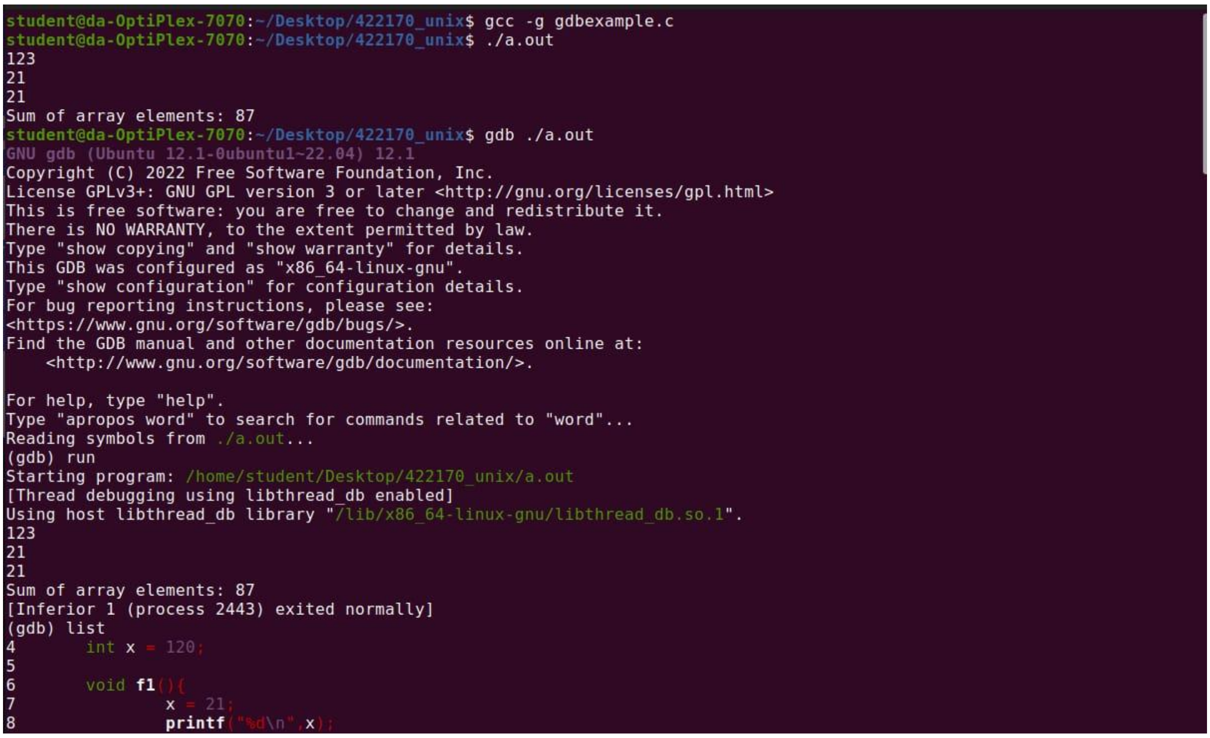
ROLL NO: 422170

SECTION : A

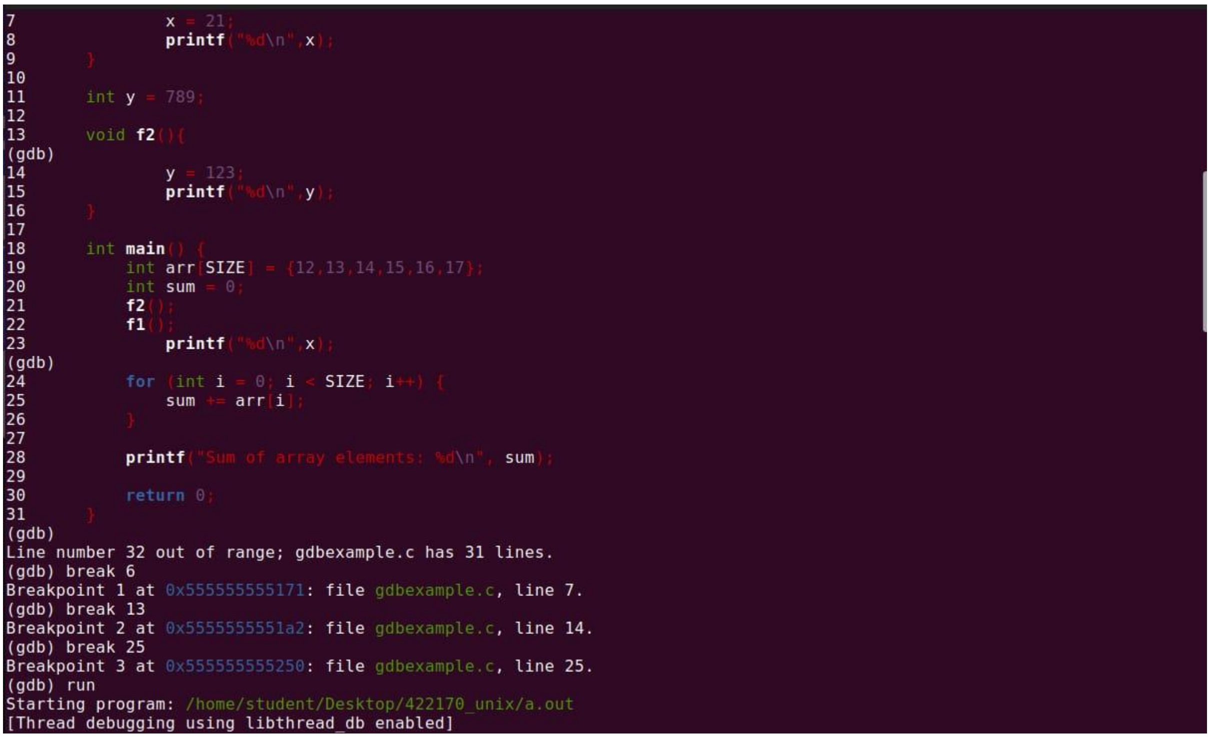
GDBEXAMPLE.C

```
include <stdio.h>

#define SIZE 6
int x = 120;

void f1(){
        x = 21;
        printf("%d\n",x);
}

int y = 789;

void f2(){
        y = 123;
        printf("%d\n",y);
}

int main() {
   int arr[SIZE] = {12,13,14,15,16,17};
   int sum = 0;
```

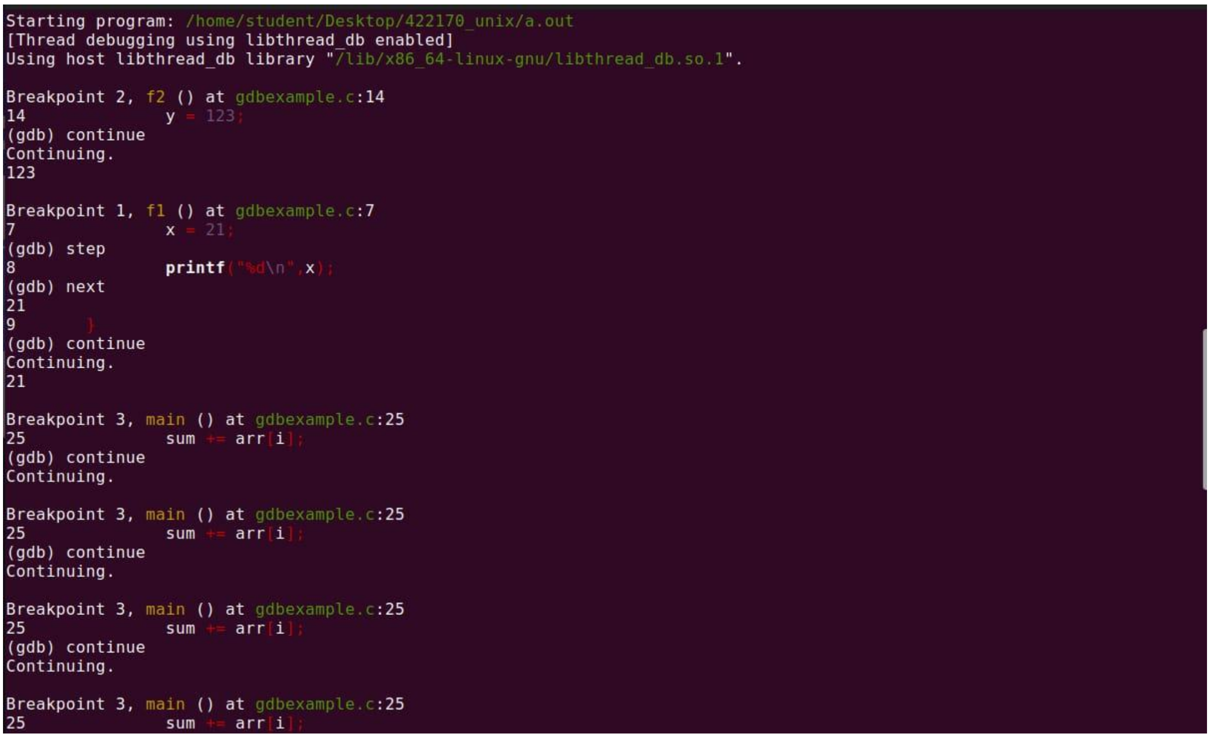
```
    f2();

    f1();

          printf("%d\n",x);

    for (int i = 0; i < SIZE; i++) {

        sum += arr[i];

    }



    printf("Sum of array elements: %d\n", sum);



    return 0;

}
```



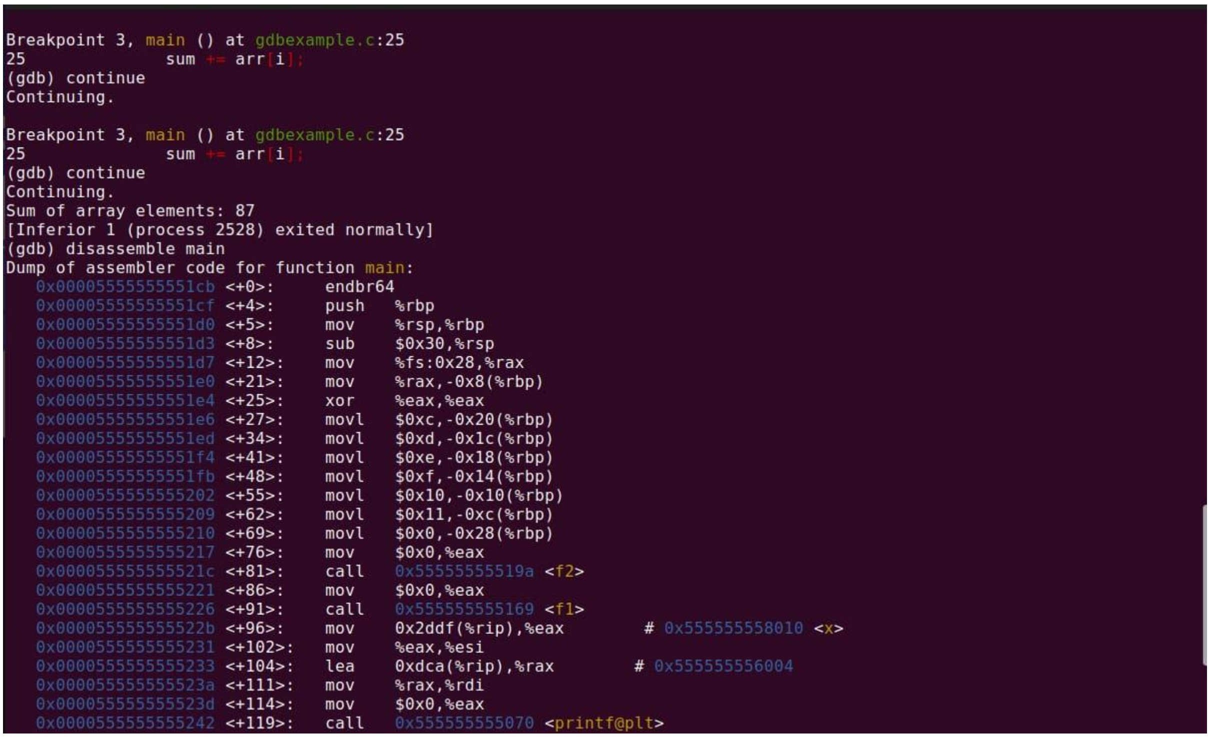
```
student@da-OptiPlex-7070:~/Desktop/422170_unix$ gcc -g gdbexample.c
student@da-OptiPlex-7070:~/Desktop/422170_unix$ ./a.out
123
21
21
Sum of array elements: 87
student@da-OptiPlex-7070:~/Desktop/422170_unix$ gdb ./a.out
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./a.out...
(gdb) run
Starting program: /home/student/Desktop/422170_unix/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
123
21
21
Sum of array elements: 87
[Inferior 1 (process 2443) exited normally]
(gdb) list
4       int x = 120;
5
6       void f1(){
7               x = 21;
8               printf("%d\n",x);
```



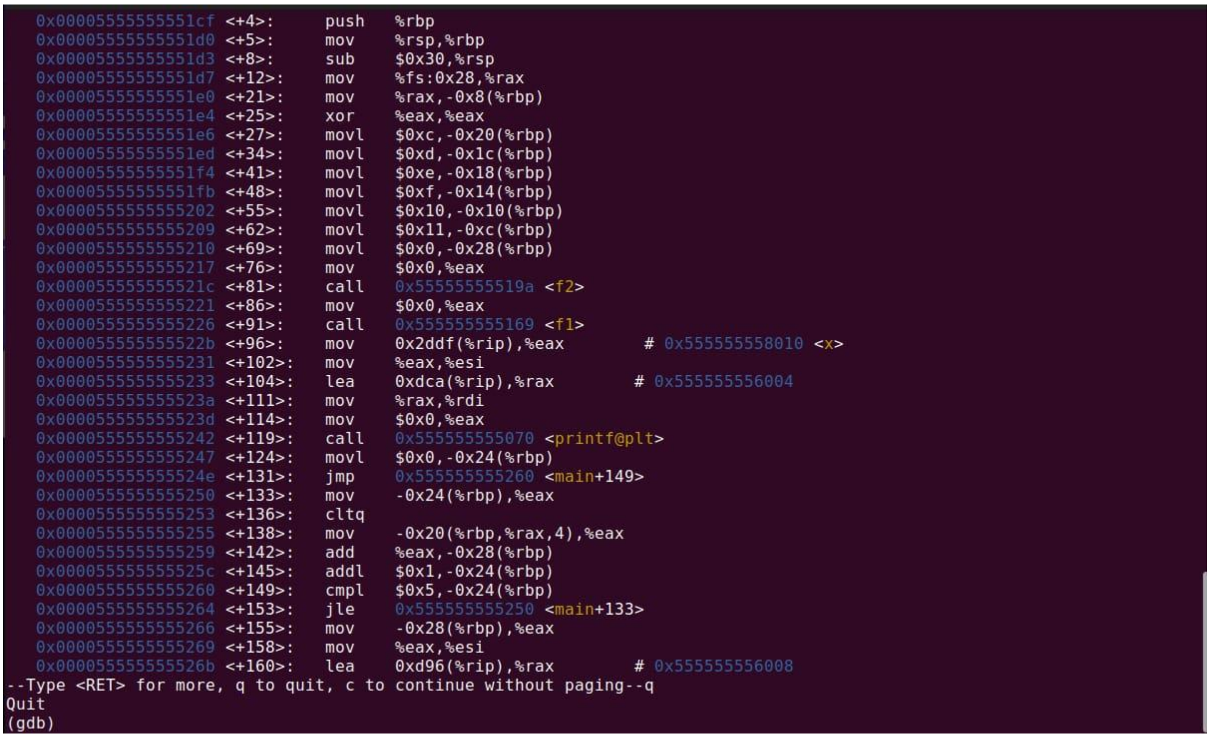
```
7               x = 21;
8               printf("%d\n",x);
9       }
10
11      int y = 789;
12
13      void f2(){
(gdb)
14              y = 123;
15              printf("%d\n",y);
16      }
17
18      int main() {
19          int arr[SIZE] = {12,13,14,15,16,17};
20          int sum = 0;
21          f2();
22          f1();
23              printf("%d\n",x);
(gdb)
24          for (int i = 0; i < SIZE; i++) {
25              sum += arr[i];
26          }
27
28          printf("Sum of array elements: %d\n", sum);
29
30          return 0;
31      }
(gdb)
Line number 32 out of range; gdbexample.c has 31 lines.
(gdb) break 6
Breakpoint 1 at 0x555555555171: file gdbexample.c, line 7.
(gdb) break 13
Breakpoint 2 at 0x5555555551a2: file gdbexample.c, line 14.
(gdb) break 25
Breakpoint 3 at 0x555555555250: file gdbexample.c, line 25.
(gdb) run
Starting program: /home/student/Desktop/422170_unix/a.out
[Thread debugging using libthread_db enabled]
```



```
Starting program: /home/student/Desktop/422170_unix/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 2, f2 () at gdbexample.c:14
14              y = 123;
(gdb) continue
Continuing.
123

Breakpoint 1, f1 () at gdbexample.c:7
7               x = 21;
(gdb) step
8               printf("%d\n",x);
(gdb) next
21
9       }
(gdb) continue
Continuing.
21

Breakpoint 3, main () at gdbexample.c:25
25              sum += arr[i];
(gdb) continue
Continuing.

Breakpoint 3, main () at gdbexample.c:25
25              sum += arr[i];
(gdb) continue
Continuing.

Breakpoint 3, main () at gdbexample.c:25
25              sum += arr[i];
(gdb) continue
Continuing.

Breakpoint 3, main () at gdbexample.c:25
25              sum += arr[i];
```



```
Breakpoint 3, main () at gdbexample.c:25
25                sum += arr[i];
(gdb) continue
Continuing.

Breakpoint 3, main () at gdbexample.c:25
25                sum += arr[i];
(gdb) continue
Continuing.
Sum of array elements: 87
[Inferior 1 (process 2528) exited normally]
(gdb) disassemble main
Dump of assembler code for function main:
   0x00005555555551cb <+0>:     endbr64
   0x00005555555551cf <+4>:     push   %rbp
   0x00005555555551d0 <+5>:     mov    %rsp,%rbp
   0x00005555555551d3 <+8>:     sub    $0x30,%rsp
   0x00005555555551d7 <+12>:    mov    %fs:0x28,%rax
   0x00005555555551e0 <+21>:    mov    %rax,-0x8(%rbp)
   0x00005555555551e4 <+25>:    xor    %eax,%eax
   0x00005555555551e6 <+27>:    movl   $0xc,-0x20(%rbp)
   0x00005555555551ed <+34>:    movl   $0xd,-0x1c(%rbp)
   0x00005555555551f4 <+41>:    movl   $0xe,-0x18(%rbp)
   0x00005555555551fb <+48>:    movl   $0xf,-0x14(%rbp)
   0x0000555555555202 <+55>:    movl   $0x10,-0x10(%rbp)
   0x0000555555555209 <+62>:    movl   $0x11,-0xc(%rbp)
   0x0000555555555210 <+69>:    movl   $0x0,-0x28(%rbp)
   0x0000555555555217 <+76>:    mov    $0x0,%eax
   0x000055555555521c <+81>:    call   0x55555555519a <f2>
   0x0000555555555221 <+86>:    mov    $0x0,%eax
   0x0000555555555226 <+91>:    call   0x555555555169 <f1>
   0x000055555555522b <+96>:    mov    0x2ddf(%rip),%eax        # 0x555555558010 <x>
   0x0000555555555231 <+102>:   mov    %eax,%esi
   0x0000555555555233 <+104>:   lea    0xdca(%rip),%rax        # 0x555555556004
   0x000055555555523a <+111>:   mov    %rax,%rdi
   0x000055555555523d <+114>:   mov    $0x0,%eax
   0x0000555555555242 <+119>:   call   0x555555555070 <printf@plt>
```



```
   0x00005555555551cf <+4>:     push   %rbp
   0x00005555555551d0 <+5>:     mov    %rsp,%rbp
   0x00005555555551d3 <+8>:     sub    $0x30,%rsp
   0x00005555555551d7 <+12>:    mov    %fs:0x28,%rax
   0x00005555555551e0 <+21>:    mov    %rax,-0x8(%rbp)
   0x00005555555551e4 <+25>:    xor    %eax,%eax
   0x00005555555551e6 <+27>:    movl   $0xc,-0x20(%rbp)
   0x00005555555551ed <+34>:    movl   $0xd,-0x1c(%rbp)
   0x00005555555551f4 <+41>:    movl   $0xe,-0x18(%rbp)
   0x00005555555551fb <+48>:    movl   $0xf,-0x14(%rbp)
   0x0000555555555202 <+55>:    movl   $0x10,-0x10(%rbp)
   0x0000555555555209 <+62>:    movl   $0x11,-0xc(%rbp)
   0x0000555555555210 <+69>:    movl   $0x0,-0x28(%rbp)
   0x0000555555555217 <+76>:    mov    $0x0,%eax
   0x000055555555521c <+81>:    call   0x55555555519a <f2>
   0x0000555555555221 <+86>:    mov    $0x0,%eax
   0x0000555555555226 <+91>:    call   0x555555555169 <f1>
   0x000055555555522b <+96>:    mov    0x2ddf(%rip),%eax        # 0x555555558010 <x>
   0x0000555555555231 <+102>:   mov    %eax,%esi
   0x0000555555555233 <+104>:   lea    0xdca(%rip),%rax        # 0x555555556004
   0x000055555555523a <+111>:   mov    %rax,%rdi
   0x000055555555523d <+114>:   mov    $0x0,%eax
   0x0000555555555242 <+119>:   call   0x555555555070 <printf@plt>
   0x0000555555555247 <+124>:   movl   $0x0,-0x24(%rbp)
   0x000055555555524e <+131>:   jmp    0x555555555260 <main+149>
   0x0000555555555250 <+133>:   mov    -0x24(%rbp),%eax
   0x0000555555555253 <+136>:   cltq
   0x0000555555555255 <+138>:   mov    -0x20(%rbp,%rax,4),%eax
   0x0000555555555259 <+142>:   add    %eax,-0x28(%rbp)
   0x000055555555525c <+145>:   addl   $0x1,-0x24(%rbp)
   0x0000555555555260 <+149>:   cmpl   $0x5,-0x24(%rbp)
   0x0000555555555264 <+153>:   jle    0x555555555250 <main+133>
   0x0000555555555266 <+155>:   mov    -0x28(%rbp),%eax
   0x0000555555555269 <+158>:   mov    %eax,%esi
   0x000055555555526b <+160>:   lea    0xd96(%rip),%rax        # 0x555555556008
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
(gdb)
```