CS-2263

LAB - 2

Saheb Singh Arora

3742233

Exercise 1

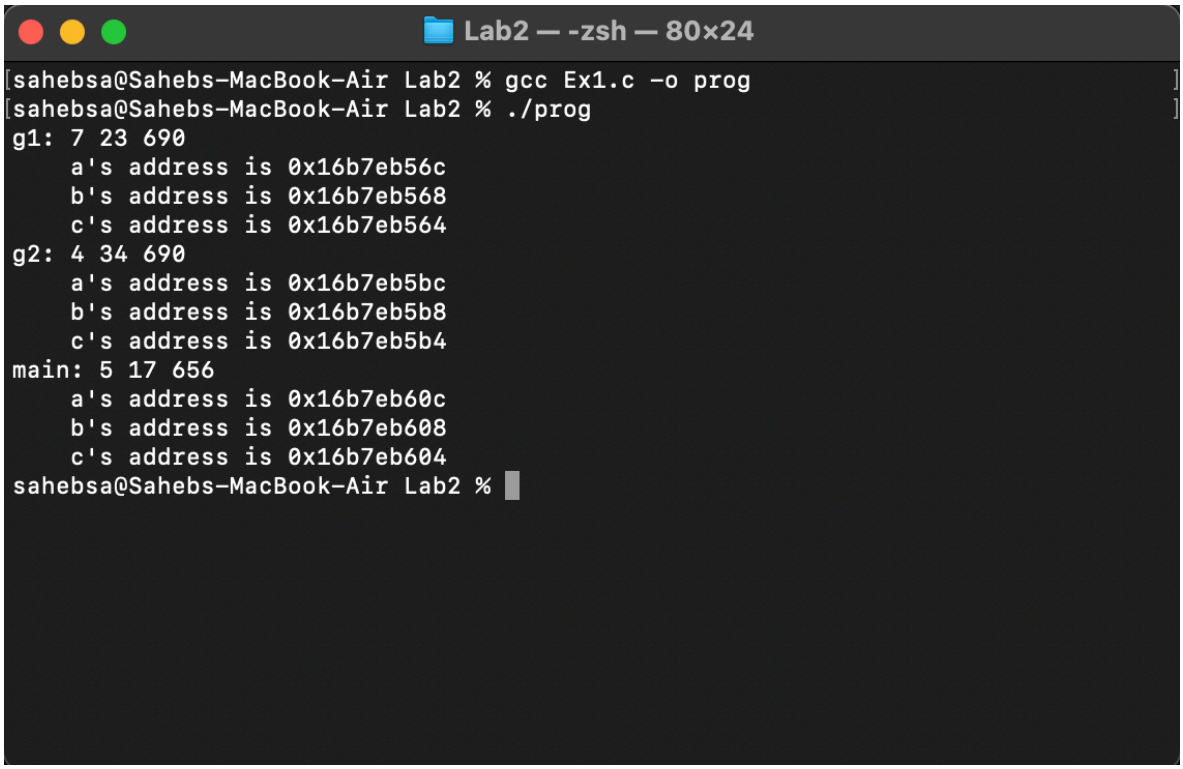
```
/**
@author Saheb Singh Arora
Student id: 3742233
*/

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

int g1(int a, int b)
{
   int c = (a + b) * b;
   printf("g1: %d %d %d\n", a, b, c);
   printf("    a's address is %p\n", (void*)&a);
   printf("    b's address is %p\n", (void*)&b);
   printf("    c's address is %p\n", (void*)&c);
   return c;
}

int g2(int a, int b)
{
   int c = g1(a + 3, b - 11);
   printf("g2: %d %d %d\n", a, b, c);
   printf("    a's address is %p\n", (void*)&a);
   printf("    b's address is %p\n", (void*)&b);
   printf("    c's address is %p\n", (void*)&c);
   return c - b;
}

int main(int argc, char* argv[])
{
   int a = 5;
   int b = 17;
   int c = g2(a - 1, b * 2);
   printf("main: %d %d %d\n", a, b, c);
   printf("    a's address is %p\n", (void*)&a);
   printf("    b's address is %p\n", (void*)&b);
   printf("    c's address is %p\n", (void*)&c);
   return EXIT_SUCCESS;
}
```



1) Are the values of the variables printed from your program the same as obtained by your colleagues? Why?

ANS:
The values of variables printed from my program should generally be the same as those obtained by colleagues if they are running the same code in the same environment with the same input. Any differences could be due to variations in environment, compiler optimization, undefined behavior, or input data.

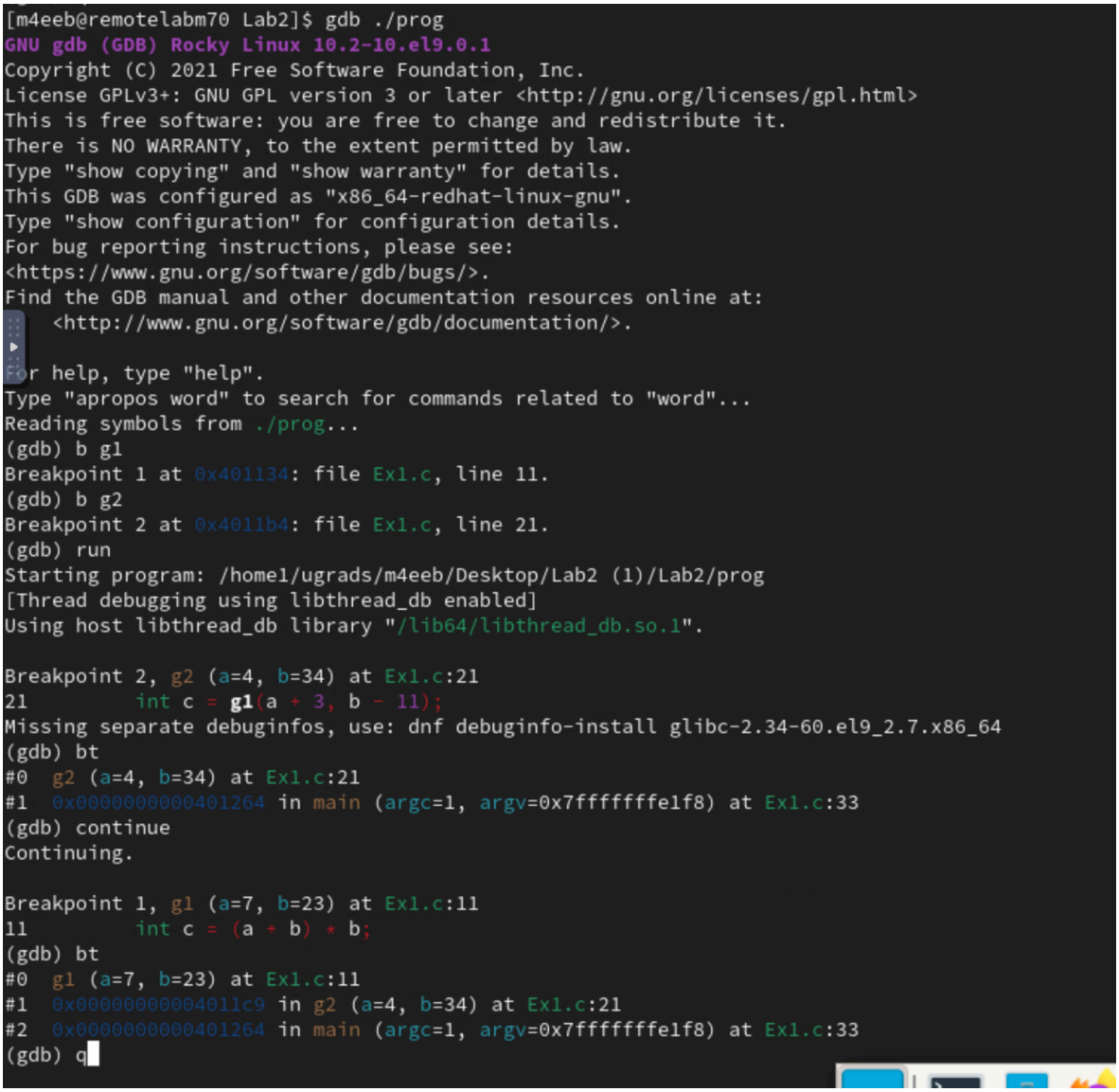
2) Are the addresses printed from your program the same as obtained by your colleagues? Why?

ANS:
The addresses printed from my program may not be the same as those obtained by colleagues, even when running the same code in the same environment. This discrepancy can occur due to the nature of memory allocation and the specific memory layout of each execution. Each time a program runs, the operating system allocates memory dynamically, and the addresses assigned to variables can vary between different executions. Additionally, factors such as memory fragmentation and address space layout randomization (ASLR) can further contribute to differences in memory addresses. Therefore, while the general behavior of the program remains consistent, the specific memory addresses may differ between executions.

3) Are the addresses printed for the variables used in the function g1 higher or lower than the addresses printed from the function g2? Why?

ANS:
In most cases, the addresses printed for the variables used in the function `g1` will be lower than the addresses printed from the function `g2`. This is because the variables in `g1` are allocated on the stack when `g1` is called, and subsequent function calls, such as `g2`, typically push new stack frames onto the stack, causing the stack to grow downwards in memory. Consequently, variables allocated earlier in the program's execution (like those in `g1`) tend to have lower memory addresses compared to variables allocated later (like those in `g2`).

Exercise 2
2.1



2.2

```
[m4eeb@remotelabm70 Lab2]$ gdb ./prog
GNU gdb (GDB) Rocky Linux 10.2-10.el9.0.1
Copyright (C) 2021 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-redhat-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 ./prog...
(gdb) b g1
Breakpoint 1 at 0x401134: file Ex1.c, line 11.
(gdb) b g2
Breakpoint 2 at 0x4011b4: file Ex1.c, line 21.
(gdb) run
Starting program: /home1/ugrads/m4eeb/Desktop/Lab2 (1)/Lab2/prog
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".

Breakpoint 2, g2 (a=4, b=34) at Ex1.c:21
21          int c = g1(a + 3, b - 11);
Missing separate debuginfos, use: dnf debuginfo-install glibc-2.34-60.el9_2.7.x86_64
(gdb) info frame
Stack level 0, frame at 0x7fffffffe0c0:
 rip = 0x4011b4 in g2 (Ex1.c:21); saved rip = 0x401264
 called by frame at 0x7fffffffe0f0
 source language c.
 Arglist at 0x7fffffffe0b0, args: a=4, b=34
 Locals at 0x7fffffffe0b0, Previous frame's sp is 0x7fffffffe0c0
 Saved registers:
  rbp at 0x7fffffffe0b0, rip at 0x7fffffffe0b8
(gdb) continue
Continuing.

Breakpoint 1, g1 (a=7, b=23) at Ex1.c:11
11          int c = (a + b) * b;
(gdb) info frame
Stack level 0, frame at 0x7fffffffe090:
(gdb) info frame
Stack level 0, frame at 0x7fffffffe090:
 rip = 0x401134 in g1 (Ex1.c:11); saved rip = 0x4011c9
 called by frame at 0x7fffffffe0c0
 source language c.
 Arglist at 0x7fffffffe080, args: a=7, b=23
 Locals at 0x7fffffffe080, Previous frame's sp is 0x7fffffffe090
 Saved registers:
  rbp at 0x7fffffffe080, rip at 0x7fffffffe088
(gdb) x/4x &a
0x7fffffffe06c: 0x00000007      0x00000040      0x00000000      0x00140000
(gdb) x/4x &b
0x7fffffffe068: 0x00000017      0x00000007      0x00000040      0x00000000
(gdb) x/4x &c
0x7fffffffe07c: 0x00000000      0xffffe0b0      0x00007fff      0x004011c9
(gdb)
```

Exercise 3

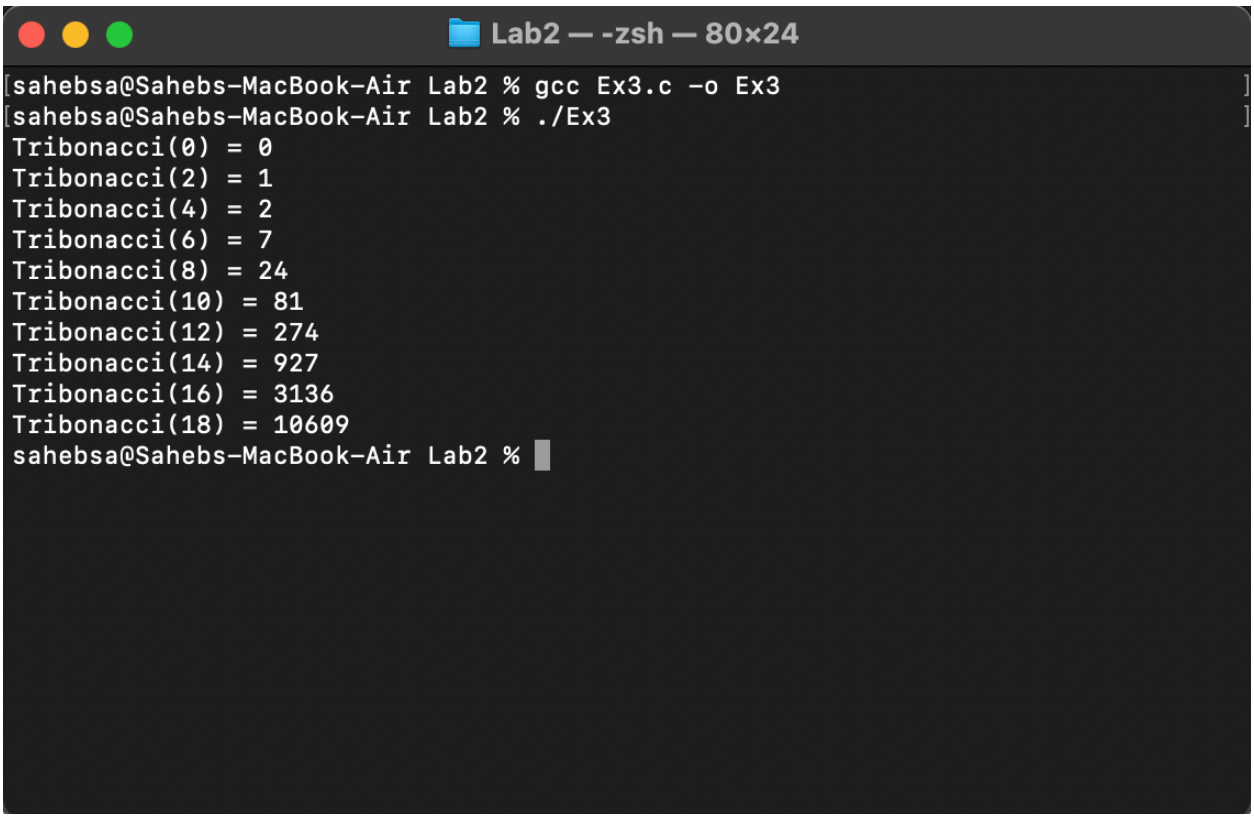
```
/**
@author Saheb Singh Arora
Student id 3742233
*/
#include <stdio.h>

unsigned long long calcTrib(int n) {
    if (n == 0)
        return 0;
    else if (n == 1 || n == 2)
        return 1;
    else {
        unsigned long long a = 0, b = 0, c = 1, next;
        for (int i = 3; i <= n; ++i) {
            next = a + b + c;
            a = b;
            b = c;
            c = next;
        }
        return c;
    }
}

int main() {
    int n = 0;
    unsigned long long trib;

    for (int i = 0; i < 10; ++i) {
        trib = calcTrib(n);
        printf("Tribonacci(%d) = %llu\n", n, trib);
        n += 2;
    }

    return 0;
}
```




```
[m4eeb@remotelabm32 Lab2]$ gdb ./Ex3
GNU gdb (GDB) Rocky Linux 10.2-10.el9.0.1
Copyright (C) 2021 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-redhat-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 ./Ex3...
(gdb) break calcTrib
Breakpoint 1 at 0x40112d: file Ex3.c, line 8.
(gdb) run
Starting program: /home1/ugrads/m4eeb/Downloads/Lab2 (2) (1)/Lab2/Ex3
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".

Breakpoint 1, calcTrib (n=0) at Ex3.c:8
8           if (n == 0)
Missing separate debuginfos, use: dnf debuginfo-install glibc-2.34-60.el9_2.7.x86_64
(gdb) continue
Continuing.
Tribonacci(0) = 0

Breakpoint 1, calcTrib (n=2) at Ex3.c:8
8           if (n == 0)
(gdb) bt
#0  calcTrib (n=2) at Ex3.c:8
#1  0x00000000004011d0 in main () at Ex3.c:29
(gdb)
```