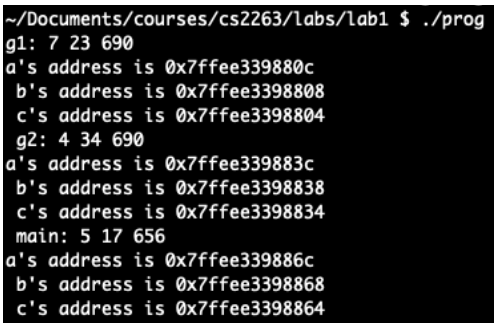
CS2263 Lab 1
Stephen Cole
3553803

Exercise 1

p1.c

```
/* p1.c */
#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 ", &a);
 printf("b's address is %p\n ", &b);
 printf("c's address is %p\n ", &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 ", &a);
 printf("b's address is %p\n ", &b);
 printf("c's address is %p\n ", &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 ", &a);
 printf("b's address is %p\n ", &b);
 printf("c's address is %p\n ", &c);
 return EXIT_SUCCESS;
}
```



Questions:

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

Yes, they are the same because the values in the program were explicitly set.

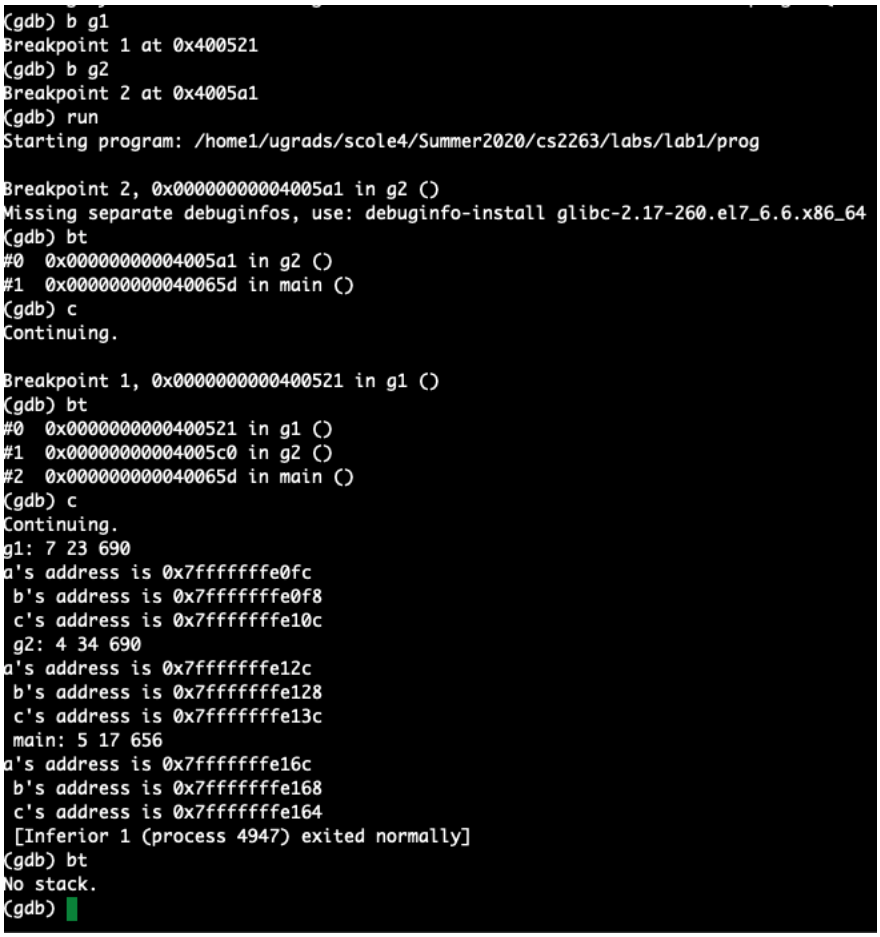
Are the addresses printed from your program the same as obtained by your colleagues? Why?

No, they are not the same because we are using different operating systems and different sections of memory were allocated.

Are the addresses printed for the variables in the function g1 bigger or smaller than the addresses printed from the function g2? Why?

The addresses in g2 are larger than the addresses of g1 because the stack frame for g2 was created before g1.

Question 2



Questions

The stack address is the position relative to the stack, where the variable addresses are their address in memory.

Question 3

```
(gdb) b isFib
Breakpoint 1 at 0x400591
(gdb) b isPrime
Breakpoint 2 at 0x4005ff
(gdb) run
Starting program: /home1/ugrads/scole4/Summer2020/cs2263/assignments/a1/primefibs

Enter the smallest number of your range: 1
Enter the largest number of your range: 10

Breakpoint 1, 0x0000000000400591 in isFib ()
Missing separate debuginfos, use: debuginfo-install glibc-2.17-260.el7_6.6.x86_64
(gdb) c
Continuing.

Breakpoint 2, 0x00000000004005ff in isPrime ()
(gdb) bt
#0  0x00000000004005ff in isPrime ()
#1  0x0000000000400686 in findPrimeFibs ()
#2  0x000000000040070e in main ()
(gdb) c
Continuing.

Breakpoint 1, 0x0000000000400591 in isFib ()
(gdb) bt
#0  0x0000000000400591 in isFib ()
#1  0x0000000000400677 in findPrimeFibs ()
#2  0x000000000040070e in main ()
(gdb)
```