CIS 452 02 – Lab 8 – Gabrielle Munson and Gabe Baksa

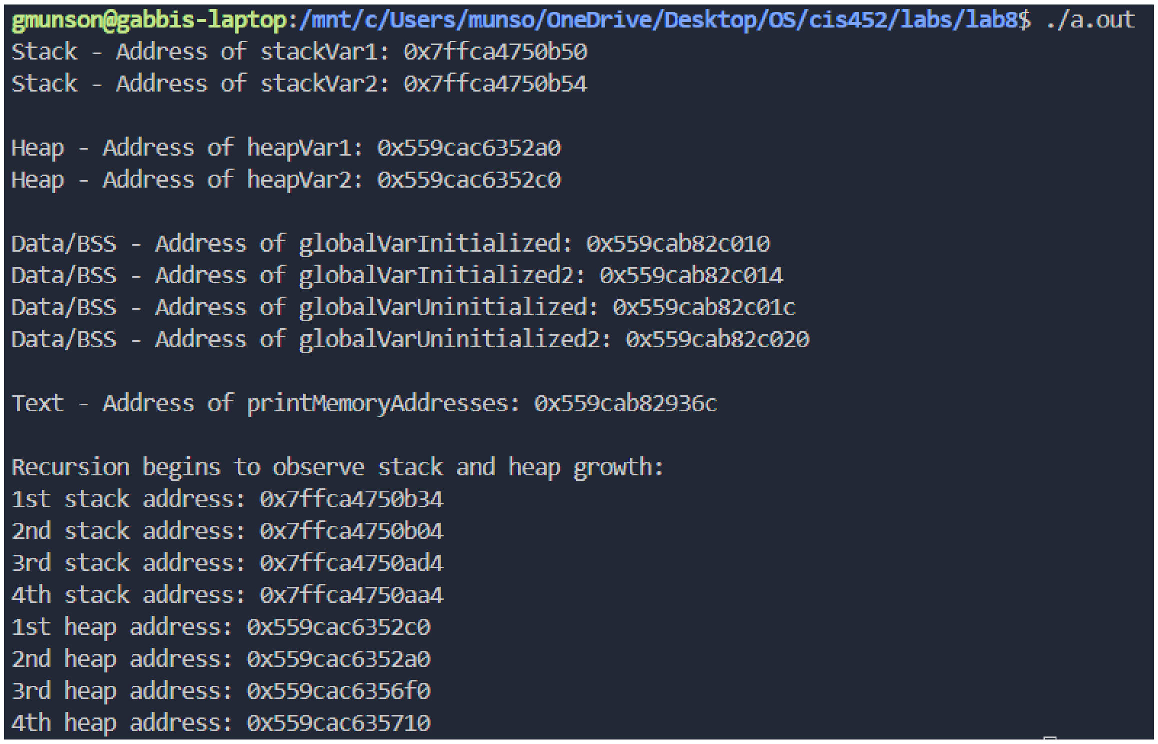
Sample Program 1

1. **Describe the error precisely (nature of problem, location). Fix the problem and submit your corrected source code along with a screenshot of the execution. Develop a robust solution that allows the user to enter any size username (within reason) without massively over allocating memory. (10 points)**
   The issue in this program is that no bounds checking is performed on the input. If the input exceeds the allocated memory of 16 bytes, it may overwrite memory that is not in the allocated space, leading to a buffer overflow. This problem specifically arises at the line where *scanf("%s", data1);* is called, which highlights the need for proper input validation to prevent memory corruption.

Memory Mapping

Section	Variable Type	Memory Address
Stack ↓	Local Stack Variable (stack_var2)	0x7ffeb9c8d484
	Local Stack Variable (stack_var1)	0x7ffeb9c8d480
	Dynamically Allocated (heap_var2)	0x55bd88eec2c0
Heap ↑	Dynamically Allocated (heap_var1)	0x55bd88eec2a0
	Uninitialized Global (global_var_uninitialized2)	0x55bd8818e020
	Uninitialized Global (global_var_uninitialized)	0x55bd8818e01c
	Initialized Global (global_var_initialized2)	0x55bd8818e014
Data/BSS ↑	Initialized Global (global_var_initialized)	0x55bd8818e010
Text ↑	Function (print_memory_addresses)	0x55bd8818b1a9

2. **What's going on here, and why?**
   The "hopping" of dynamic segments like the stack, heap, and libraries in memory each time the program is run is due to Address Space Layout Randomization (ASLR), a security feature that randomizes their starting addresses. ASLR, which is an example of address obfuscation, prevents attackers from predicting where data or code will reside, making it much harder to exploit vulnerabilities like buffer overflows.
3. **Submit the source code you used along with the memory map diagram mapping out all of the aforementioned regions. Submit a screenshot of your program execution.**



```
gmunson@gabbis-laptop:/mnt/c/Users/munso/OneDrive/Desktop/OS/cis452/labs/lab8$ ./a.out
Stack - Address of stackVar1: 0x7ffca4750b50
Stack - Address of stackVar2: 0x7ffca4750b54

Heap - Address of heapVar1: 0x559cac6352a0
Heap - Address of heapVar2: 0x559cac6352c0

Data/BSS - Address of globalVarInitialized: 0x559cab82c010
Data/BSS - Address of globalVarInitialized2: 0x559cab82c014
Data/BSS - Address of globalVarUninitialized: 0x559cab82c01c
Data/BSS - Address of globalVarUninitialized2: 0x559cab82c020

Text - Address of printMemoryAddresses: 0x559cab82936c

Recursion begins to observe stack and heap growth:
1st stack address: 0x7ffca4750b34
2nd stack address: 0x7ffca4750b04
3rd stack address: 0x7ffca4750ad4
4th stack address: 0x7ffca4750aa4
1st heap address: 0x559cac6352c0
2nd heap address: 0x559cac6352a0
3rd heap address: 0x559cac6356f0
4th heap address: 0x559cac635710
```