IC221 Lab: Memory Leaks Worksheet Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Sping AY2022, 100 points total

**Task 1 (50 points)**

(5) Compile and execute memleak.c. Verify the output and review the program.

(10) Run valgrind on the memleak program. How many bytes does it say have been “definitely” lost?

|  |
| --- |
| 60 bytes |

(5) On what line(s) of code does valgrind indicate a memory leak has occurred? 34 and 50

(10) Identify and describe at least one memory leak in memleak.c.

|  |
| --- |
| When the program calls doubleup, it loses the original pointer a and the memory that was calloced at the address of a. |

(10) Fix the memory leak you identified and verify your fix with valgrind.

(10) Describe how you fixed the memory leak:

|  |
| --- |
| Before you return new\_a in doubleup, I freed a using free(a); |

**Task 2 (50 points)**

(5) Compile and execute the memviolation.c program. - Done

(10) Describe the output and exeuction of the program. Does it seem to be consistent?

|  |
| --- |
| The program outputs “Hello World!” but has an invalid read of size one on line 16. Yes this is consistent. |

(10) Run the program under valgrind. Identify the line of code that is causing the memory violation and its input:

|  |
| --- |
| The memory violation is being caused on line 16. It’s a printf statement trying to print a string, but we are passing a char array without a null byte. |

(15) Describe the programming bug:

|  |
| --- |
| The program callocs str as size 12. Char array hello is really 13 bytes long with a null byte at the end, so you need add one to the malloc creation of str. You also need to add 1 to the for loop condition. |

(10) \_\_\_\_\_ Fix the memory violation and verify your fix with valgrind. -Done

**Submission**

- Fixed memleak.c

- Fixed memviolation.c

- This completed worksheet