IC221 Lab: Memory Leaks Worksheet Name Patrick Catren

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? 34,19, 50\_\_\_\_\_

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

|  |
| --- |
| It reassigns pointer a to the array output by doubleup, however a was not freed from it’s initial value. |

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

(10) Describe how you fixed the memory leak:

|  |
| --- |
| I instead assigned the output of doubleup(a, TEST\_SIZE) to int \* b that I added. I then added free(a) and free(b) to the end of main. This resulted in no memory leaks as all heap blocks were freed. |

**Task 2 (50 points)**

(5) \_\_\_\_\_ Compile and execute the memviolation.c program.

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

|  |
| --- |
| The output matches the described function of the program, it outputs Hello World as desired. |

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

|  |
| --- |
| Line 16, the printf with str as the input. |

(15) Describe the programming bug:

|  |
| --- |
| By using strlen to create the new array, there is not space for it to copy the null byte ending of hello that would be one index higher than the strlen value. |

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

**Submission**

- Fixed memleak.c

- Fixed memviolation.c

- This completed worksheet