# Software Dev Challenge Problems

Please read through and complete the following challenges. You are free to use any tools/languages unless specifically instructed otherwise. Please create a 30-45 minute presentation to highlight your solutions. Please be sure to go into detail on your approach, any issues you encountered and any assumptions you made. Also include the process you took to validate that your solution was correct.

# Cuboid

The attached source file is supposed to print out a cuboid given the x, y, and z dimensions via command line arguments but it appears there are a lot of bugs preventing the compiled program from working correctly. We have a version of the binary working correctly attached as well but we want to eventually make extensions to the source. Can you fix the source and get the program working as intended?

To be clear, the compiled cuboid program is working as intended, we would like you to fix the cuboid.c source code to have identical behavior.

Requirements:

\* Program should take 3 numbers (such that 0 < dimension <= 10) from the command line as program arguments. If the user passes invalid input, the program should exit and print out nothing

\* No trailing whitespace after characters that make up the cuboid

\* Cuboid must be displayed using the same orientation/viewpoint

# Library Finder

Create a program in C that can locate and print out the base address of each loaded library in memory as hex on a single line each (printf format: 0x%016lx\n).

Example: Say that running `ldd` on a program produces the following output --

```

$ ldd my\_prog

linux-vdso.so.1 (0x00007ffe0eb03000)

libdl.so.2 => /lib/x86\_64-linux-gnu/libdl.so.2 (0x00007f4f1b70b000)

libc.so.6 => /lib/x86\_64-linux-gnu/libc.so.6 (0x00007f4f1b31a000)

/lib64/ld-linux-x86-64.so.2 (0x00007f4f1b90f000)

```

The written code here should then write to stdout:

```

0x00007ffe0eb03000

0x00007f4f1b70b000

0x00007f4f1b31a000

0x00007f4f1b90f000

```

Note: Output must be in load dependency order like `ldd`.

Hint: When symbols are lazily relocated during runtime in ELF binaries, the GOT (Global offset table) contains a link map entry. This entry contains linkage information to help the dynamic linker known what libraries and symbols are available.

# XML Parser

Using the given allitems-cvrf-year-2022.xml file, please parse and display all Vulnerabilities that include a reference to "gitlab". You can use any language to write the program. The output should contain only text (no XML) and follow this format:

Vulnerability Number:

Title:

Notes:

References:

<blank line>

# GDB

Using the provided coredump files, please find the value of the local variable “flag” within the “battelle” function when the core file was dumped