School of Computer Engineering

Dr. Reza Entezari-Maleki

Fall 2021

Practical Assignment #6
Operating Systems

Sina Shabani - Hadi Sheikhi

Due Date: 23:59:59 2th Jan, 2021

OS Practical Assignment #6

This Assignment is not as hard as others you can do it simply by following the steps. You will learn about tools that will help you level up your c programming skills and a touch on memory as well.

Submission ** IMPORTANT **

- All your files must be in an archive with your student id as the name. So `student_id.zip`
- You are faced with some tasks. After finishing one there might be some questions marked with letter `Q`. The answer to all of them must be in a single file (pdf, markdown or text) and included in your archive.
- You also need to write some programs which you have to include in your archive as well.

How to Get Help

- Try to get comfy with `man` command of your teminal. You can find many usefull information about this assignment
- If you didn't find your answer with the `man` command which I highly doubt that, You can surf the web for solutions. I added titles for each question to help you to search better.
- You didn't find it yet?! How? Obviouslly that's a joke
- You can contact us
  - @itsloopp
  - @hadichhh

Requirements

We assume you are using ubuntu. If you are using operating systems or another distribution of linux these steps might be different for you and you need to look it up on the web.

- Remeber to update your repositories with `sudo apt update`
- gcc
  - You should be familiar with this one by now.
- gdb
  - It's a debuging tool for c and c++ programs that let's you see what is going on inside a program while it is being executed. gdb is short for GNU Debugger
  - To install it on ubuntu you can type `sudo apt install gdb` in your terminal.
  - After installation to get more information type `man gdb` in your terminal.
- valgrind

- Valgrind is a flexible program for debugging and profiling Linux executables. You can find memory leaks with this program. It even works for multi-threaded programs.
- To install it on ubuntu you can type `sudo apt install valgrind` in your terminal.
- After installation to get more information type `man valgrind` in your terminal.

Question 1 : Segmentation Fault

For this Question you will be writing a simple program that tries to derefrece a `NULL` pointer and then debug it with some tools.

Steps

- Write a program called null.c
- Create a pointer and initialize it with `NULL` (i.e. `int *ptr = NULL`).
- Then Derefrence this pointer (access the value that the pointer is pointing).
- Now compile this program with the name **`null`** and run it (`gcc -o null null.c`).
- Q: What is the output of the program? And why do you think this happened? I
- Now compile the program again this time add **`-g`** flag to gcc (`gcc -g -o null null.c`)
- Q: What do you think adding `-g` option does? (you can use `man gcc` or search on the web)
- Now Run the program with gdb (`gdb null`), and after gdb is running type **`run`** and press enter.
- Q: What does gdb shows you and How does running the program with gdb help us?
- To exit out of gdb type `quit` and if it says a debugging session is active just type `y` to quit anyway.
- Now run the program with `valgrind`. You need to use the `memcheck` tool, which is for analyzing the memory and what happened. Run `valgrind --tool=memcheck --leak-check=yes ./null`
- Q: What happens when you run this command and can you interpret the output of `valgrind`.

Question 2 : Memory Leak

Steps

- Write a program `leak.c`, that allocates some memory and exits without freeing the memory.
- Remeber to use `-g` flag.
- Run in these three scenarios.

  - Run it normally
  - Run it with `gdb`
  - Run it with `valgrind` (don't forget the --leak-check=yes flag)
- Q: For each scenario explain what was the output and can you find any problems with your program.

Question 3 : Index Out of Range

Steps

- Write another program called `index.c`.
- This program should create an array of integers called `data` and then sets `data[100]` to your student number.
- Q: What happens when you run this program? Does your program crash? explain why anyway.
- Now run your program with `valgrind` (again don't forget the --leak-check=yes flag, compile the program with `-g` flag)
- Q: What do you understand from the output of valgrind about your program, is your program correct?
- Now try to set `data[1000000000] = 0`.
- Q: What happens and why?

Question 4 : Access

Steps

- Write a program called `access.c`.
- This time your program should allocate `data` array form previous example and then `free` it.
- Q: Does this program run at all?
- Q: What happens when you use valgrind on it?
- Now modify your program and give a funny value to `free`. An example for a funny value can be the pointer to middle of you array (`free(data + 50)` if size of data is 100).
- Q: What happens and do you think you need special tools to find and solve this problem?