



Gebze Technical University
Department Of Computer Engineering

CSE 312 /CSE 504 Spring 2024

Operating Systems

Homework #01

Part A (50pts)

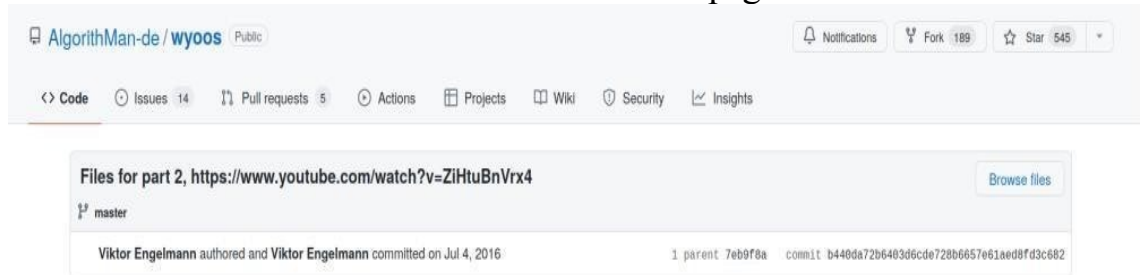
Due Date: 13.05.2024

1. Before you start the homework

Throughout the semester, we will try to write our operating system that runs on the [web site](#). This operating system is created by [Viktor Engelmann](#). He explains how to write his own operating system step by step in [the video series](#). Before you start HW#1, we strongly suggest you apply following steps:

- Firstly, watch and implement the first three videos in the playlist. [The first video](#) and [the second video](#) explain the basic idea and design of this project, so it helps you to understand his approach.
- [The third video](#) tells you how to install this operating system in a virtual box. To apply what he did so far in this video, you need to consider his source code of this video for your virtual machine. If you clone the source code on [the GitHub page](#), you will clone the latest version which includes everything about the operating system. To download the source code, which is related to the video you consider, you need to do the following steps:
 - In case of the third video, go to [the source page](#).
 - Select “Running the OS in a virtual machine” code.

- Click “Browse Files” button on the page shown below.



- In the next page, under the “Code” button, there will be “Download ZIP” option. When you select this option, you can reach the source code which is related to the video that you consider.
- If you apply everything in the third video with the source code which is explained in the previous step, and you obtain the same result as in the video, then you can start to do this homework. If you get some errors at the end of the video, you are responsible for fixing them.
- Before you start the homework, you should watch all the videos.

2. Your task for the homework

What we expect from your new OS is

- Implementing these POSIX system calls: fork, waitpid, execve, any other POSIX call that you need.
- **You must create all your processes by using fork system call.**
- Loading multiple programs into memory: Kernel will be able to load multiple programs into memory.
- Handling multi-programming: you need to develop a Process Table that will hold the necessary information about the processes in the memory. You should study what Process Tables holds. You can read carefully throughout chapter 2 of the course book or any other online resource).
- Handling Interrupts: Our given source code can handle interrupts, and your kernel will handle and respond between multiple processes.

Perform Round Robin scheduling: Every time a timer interrupt occurs, there is a chance to make a process switch. Whenever a context scheduling occurs, you will print all the information about the processes in the process table including but not limited to the entries in the list below. You can modify anything that you want if you comment on the report.

Lifecycle

This Strategy is loading each program 3 times, starting them and will enter an infinite loop until all the processes terminate.

Programs that you will test are

- Collatz

You are going to find collatz sequence for each number less than 100. You can either take input from the user or pass the number as a parameter.

You can find information about (Collatz conjecture on internet). For each number you will show the number being interested in, and its collatz sequence and go to the next number.

- Example for input 7

- Output 7: 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1

- `def long_running_program(n):`

- `result = 0`

- `for i in range(n):`

- `for j in range(n):`

- `result += i * j`

- `return result`

- `# Example usage`

- `# Use this without user input. There is no interaction with user for this.`

- `result = long_running_program(1000)`

3. General rules for homework

- a. Submit all your source files!!!

- b. Submit ReadMe.txt as execution instructions.

- c. Submissions without Makefile, Report, and ReadMe.txt will not be evaluated!

- d. Start early!

- e. It is not a group project. Do not share your answers with anyone in any circumstance. Any cheating means NA from the course directly.

f. Your homework report is very important, it should include your design decisions, your structures, your comments, codes, and results for each strategy with screen shots.

- g. For grading, syscall implementation and strategies implementation is very important.

- h. Write comments on the lines of your code where any critical actions happen.
- i. All the things that you have developed should be reported by using screen shots and comment on them!
 - a. After the deadline, you are responsible for explaining your project to the course assistant in a demonstration.
 - b. Each demo will be done in 10 minutes.
 - c. Many appointments will be open so you can select the most available time slot for yourself.
 - d. To answer all possible questions, you need to comment on your code. The solution is to write and to do everything on your own. Your homework grade will be evaluated by your answer

