Operating Systems Concepts
CSci 474

Project #1: Fork and Pipe
Due Date: 11:55PM, October 30, 2023
Submission: Submit your source code and summary to the blackboard. Late submission will NOT be accepted. Email submission will NOT be accepted.

I. Project Organization

This project will study multiprogramming using fork and pipe on a Unix/Linux machine.

• Code	30 points
• Output	10 points
• Summary	10 points

Code

The code should be nicely formatted with plenty of comments. It should be easy to read, properly indented, employ good naming standards, good structure, etc.

Output

Output will be graded by running your program. Your program will be tested on a machine (lab00.cs.ndsu.nodak.edu -- lab20.cs.ndsu.nodak.edu) in lab 244.

Summary

The summary section should discuss the results of running a single process versus multiple processes on adding numbers in a file. You should complete Table 1 and discuss your result.

Tablet 1

	File 1	File 2	File 3
1 Process			
2 Processes			
4 Processes			

The summary section should include two parts:

(1) Record the **running time** in Table 1;
(2) Discuss your result in Table 1 (single column, sing space, 12 points, 0.5-1 page): Though we expect four processes are running faster than a single process, it may not be true in practice, which can be caused by the detailed implementation, run-time resource or other factors. So, the summary will be evaluated based on your discussion, regardless of faster or slower. **Your results must be reproducible.**

If your program is not running correctly, you can write "what you have learnt and what are the challenges" in the project" (single column, sing space, 12 points, at least 1 page) in the summary report to replace Table 1 and the discussion.

II. Project Description

Problem Overview

You must use fork and pipe to complete this project. This project creates processes to add all numbers in a file. The user will enter a number (1, 2, or 4) of parallel child processes to create for processing the numbers. The system will then create this many processes, evenly dividing the file contents between the processes. For example, a file has 1000 numbers and the user wants 4 child processes. The data in the file are divided into four blocks and each block has 250 numbers. The first child handles the first block, the second child handles the second block and so on.

III. System Description

The system is illustrated in the diagram shown below. The processes are as follows:

1. Parent process. This process allows user to input the number of child processes to create (1, 2, or 4) **by calling fork**. This process also allows a user to input the index of a file to process (1, 2, or 3). The parent then waits for each child to report its result. The parent prints the result from each child and the total final result.
2. Child process. Process the corresponding block in the file and send the result back to the parent process **through pipe**.

The data files are stored in the same folder as your source code. **Avoid using absolute path in your source code**.

IV. Project Guidelines

Data Files

Data files will be posted on the course website.

- Data file 1: 1000 lines (numbers)
- Data file 2: 10000 lines (numbers)
- Data file 3: 100000 lines (numbers)

Submitting

Submit your project on Blackboard. Include in your submission the following files:

1) A Word document for the written pieces of the project
2) Your source codes
3) Screenshot of output is NOT needed.

Cheating

All work must be your own. If cheating is detected, all parties involved will be given a zero for the project and the penalty will be documented on a form that you must sign. You may be referred to the Dean's office for further discussion. If you use any online resources, you must cite the source in both the comments and the summary document.

Grading

The written portions will be graded subjectively based on completeness and quality. The code will be graded based on points allocated for each key part of the processing as determined by the instructor. The output will be graded based on expected results for the runs.