Daniel Yowell
Dr. Song Fang
CS3113
2/23/2023

# Project 1 Report

This project covered the basics of forking processes and creating shared memory in C. Our goal was to create an integer variable “total” which could be shared between four different child processes, each of which would call a function to increment the variable by a certain amount. Each time a child process was created, the parent process would wait until its child finished executing before continuing its own code.

Each process printed to output to track the program’s progress. Below is an example output from a previous run:

```
From Process 1: counter = 100000.
Child with ID: 12898 has just exited.
From Process 2: counter = 200000.
Child with ID: 12899 has just exited.
From Process 3: counter = 300000.
Child with ID: 12900 has just exited.
From Process 4: counter = 500000.
Child with ID: 12901 has just exited.

End of simulation.
```

The first thing I noticed when running this program is that the process IDs of each child seem to increment after each fork. I deduced that when creating a new process, the child process receives the next available ID in increasing order. By extension, I speculated that the process ID of the earliest parent would be 1 less than its first child (such as 12897 in the run shown above). I later confirmed this theory by calling getpid() in the parent process during a test run.

I also noticed the process IDs continually growing larger after each execution, with the starting ID of a new run being a few dozen integers larger than the last child of the previous run. I concluded that this process of assigning IDs must be universal for every process in the computer system.