# New York University
# Tandon School of Engineering
Department of Computer Science and Engineering

Introduction to Operating Systems
Spring 2025

Assignment 5
(10 points)

A) (7 points) Write a program whose main routine obtains two parameter *n* and *d* from the user (i.e., passed to your program when it was invoked from the shell, `n>1`). Your program shall then create a shared memory and a child process. The shared memory shall have a size of `BUF_SZ*sizeof(int)`, where BUF_SZ is defined as 5 using a macro, e.g. "`#define BUF_SZ 5`".

The child process should obtain the values of *n* and *d* from the parent (you actually have multiple options for doing that) and create **(but not print)** an arithmetic sequence of length *n* and whose elements are of type `int`, such that each element has a value of *kd*, where *k* is the element number (`0` to *n*`-1`). For example, if *n*`=5` and *d*`=2`, the sequence shall be 0,2,4,6 and 8.

The child process shall create the elements, one at a time, and wait for a random interval of time ( 0 <= time < 3 seconds) between generating elements of the sequence. **As soon as an element is generated, the child places the element in the shared memory** by organizing it into a fixed-size shared buffer that implements a producer-cosumer queue as described in the lectures.

The parent process shall **print** elements it receives on the shared buffer **immediately as soon as they arrive**, without waiting for the child process to exit.

**Note:** For this assignment, you **may or may not** use a virtual file to create the shared memory (i.e. anonymous vs named shared memory). Both methods will work fine since the processes have a parent-child relationship. But if you use a virtual file to create a shared memory, you should delete your virtual file (using `unlink()` ) prior to both processes exiting. For our usage in this assignment, you do the `unlink` only once, such that one process creates the virtual file and the other deletes it.

The reason why deleting a file (virtual or real file) is referred to as `unlink` will be clarified towards the final week of this course, when we discuss filesystems.

**Hint:** Use fflush() to ensure printf's are printed immediately into the screen.

B) (3 points) Repeat part A, except that now you do not create a shared memory, but rather use an ordinary pipe to pass the sequence (multiple ways of passing *n* and *d*, as in part A).

## What to submit to gradescope:

Please submit the following files individually:

1) Source file(s) with appropriate comments.
   The naming should be similar to "**lab#_$.c**" (**#** is replaced with the assignment number and **$** with the question number within the assignment, e.g. lab4_b.c, for lab 4, question b OR lab5_1a for lab 5, question 1a).
2) A single pdf file (for images + report/answers to questions), named **"lab#.pdf"** (**#** is replaced by the assignment number), containing:
   - Screen shot(**s**) of your terminal window showing the current directory, the command used to compile your program, the command used to run your program and the output of your program.
3) Your Makefile, if any. This is applicable only to kernel modules.

## RULES:

- You shall **use kernel version 4.x.x or above**. You shall not use kernel version 3.x.x.
- You may consult with other students about GENERAL concepts or methods but copying code (or code fragments) or algorithms is NOT ALLOWED and is considered cheating (whether copied form other students, the internet or any other source).
- If you are having trouble, please ask your teaching assistant for help.
- You must submit your assignment prior to the deadline.