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

Introduction to Operating Systems
Spring 2025

Assignment 7
(10 points)

Write a program that uses a multi-threaded (for speedup) Monte-Carlo simulation to estimate the area of a circle with unit radius (thus area = the value of π). This can be achieved by enclosing a circle inside a square of length 2 units.

Your program's main routine shall create a number of worker threads NUM_THREADS=4 (defined as a macro) to speedup the computation (you may name the common routine "`WorkerThread`"). Each of the threads shall use **a shared variable**, `count` (an integer), as well as **synchronization primitives** (e.g. a semaphore or mutex).

Each of the worker threads shall generate a number of randomly located points, where the number of points is defined as a macro (#define NUM_POINTS 1,000,000). Each of the randomly located points shall have *(x,y)* coordinates, with *x* and *y* ranging between -1 and 1. Each thread shall then compute whether each of the NUM_POINTS points is inside the circle (it may do so by computing the radius $r = \sqrt{x^2 + y^2}$ and evaluating if it's <=1). Each thread shall **immediately** increment the shared variable if the point is inside the circle, i.e. the **updates shall not wait** for the entire NUM_POINTS points to be computed but rather update the shared variable after each point's evaluation.

The main thread shall wait for all four worker threads to exist and print the area of the circle as:

$$4 \times \frac{points\ inside}{total\ points}$$

Some useful notes:

- If you use semaphores, **do NOT use system V semaphores**, i.e. use the calls shown in the lectures (sem_wait and sem_signal).
- Use the man pages for more info on how to use a semaphore or a mutex.
- Each thread **must** seed the random number and use its own state so it won't match the other thread's state, and thus each thread **shall** have a different sequence of random numbers, for example:
  ```
  unsigned int rand_state = (unsigned int) time(NULL) + pthread_self();
  ```
- You need to use `rand_r` to generate the random numbers and not rand(), for example:
  ```
  rand_r(&rand_state)
  ```
- Note that you will need to use the -pthread option with gcc in order to link the pthread library.

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.