CSCI 3000 (Operating Systems)
Department of Computer Science
East Carolina University
Total: 100 Marks, Due Apr 3, 2021

Instructor: Dr. Kamran Sartipi

Question 1 [10 Marks]
Consider the following code segment:

```
pid t pid;

pid = fork();
if (pid == 0) {

        fork();
        thread create( . . .);

}
fork();
```

A. How many unique processes are created?
B. How many unique threads are created?

Question 2 [10 Marks]

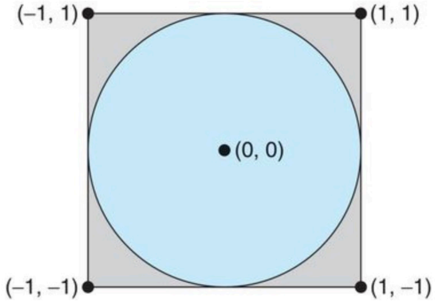
The Fibonacci sequence is the series of numbers 0, 1, 1, 2, 3, 5, 8, ... Formally, it can be expressed as:

$$\begin{aligned} fib_0 &= 0 \\ fib_1 &= 1 \\ fib_n &= fib_{n-1} + fib_{n-2} \end{aligned}$$

Write a multithreaded program that generates the Fibonacci sequence. This program should work as follows:
On the command line, the user will enter the number of Fibonacci numbers that the program is to generate. The program will then create a separate thread that will generate the Fibonacci numbers, placing the sequence in data that can be shared by the threads (an array is probably the most convenient data structure). When the thread finishes execution, the parent thread will output the sequence generated by the child thread. Because the parent thread cannot begin outputting the Fibonacci sequence until the child thread finishes, the parent thread will have to

wait for the child thread to finish. Use the techniques described in Section 4.4 to meet this requirement.

Question 3 [20 Marks]
An interesting way of calculating π is to use a technique known as Monte Carlo, which involves randomization. This technique works as follows: Suppose you have a circle inscribed within a square, as shown in figure below. (Assume that the radius of this circle is 1.)



First, generate a series of random points as simple (x, y) coordinates. These points must fall within the Cartesian coordinates that bound the square. Of the total number of random points that are generated, some will occur within the circle.

Next, estimate π by performing the following calculation:
π = 4 × (number of points in circle) / (total number of points)

Write a multithreaded version of this algorithm that creates a separate thread to generate a number of random points. The thread will count the number of points that occur within the circle and store that result in a global variable. When this thread has exited, the parent thread will calculate and output the estimated value of π. It is worth experimenting with the number of random points generated.

As a general rule, the greater the number of points, the closer the approximation to π.

Note: in the source-code download for this textbook (Module RESOURCES), you will find a sample program that generates random numbers for Java: RESOURCES >> All Source Code from Textbook >> ch 4 >> SumTask.java.

Question 4 [10 Marks]
Modify the socket-based **date server** in Chapter 3 (Figure 3.27) so that the server services each client request in a separate thread.

Question 5 [20 marks].
Write a multithreaded program that calculates various statistical values for a list of numbers. This program will be passed a series of numbers on the command line and will then create three separate worker threads. One thread will determine the average of the numbers, the second will determine the maximum value, and the third will determine the minimum value. For example, suppose your program is passed the integers

90 81 78 95 79 72 85 90 81 78 95 79 72 85

The program will report:

The average value is 82
The minimum value is 72
The maximum value is 95

The variables representing the average, minimum, and maximum values will be stored globally. The worker threads will set these values, and the parent thread will output the values once the workers have exited.

Question 6 [30 Mark]
In this program you are required to implement different process scheduling algorithms we studied in the lectures, including FCFS (first come first serve), RR (round-robin), SJF (shortest job first), PRI (priority), and PRI-RR (priority round-robin). The programs will be written in Java. The input file "sched.txt" for testing the schedulers is organized as follows:

Task name	Priority	CPU burst time
T1,	4,	20
T2,	1,	35
T3,	3,	20
T4,	2,	15

The object files (.class) for different scheduling algorithms are provided for you in a zipped file, so that you can run the programs in the Terminal window using the following command:
"java Driver ALG sched.txt"
Where "ALG" can be FCFS, RR, SJF, PRI, or PRI-RR.
The Java code for the driver program "Driver.java" and scheduling algorithm "FCFS.java" have been provided for you to learn from them and write the other scheduling algorithms accordingly and test them using the input data in file "sched.txt".

Put the Java code and the input file for Question 6 in a separate folder namely "Scheduling" and the Java code for the other programs in a folder namely "Others" and create a single zip file which contains both folders "Scheduling" and "Others", and submit to the Canvas.

END