Birla Institute of Technology & Science, Pilani, Hyderabad Campus

Programming Assignment-2

Operating System (CS F372)

Summer-Term 2019-2020

(Total Marks 3*5= 15 Marks)

Students are required to do this assignment in groups of 2 students. The same group of programming assignment 1 should be continued for this assignment as well. **The deadline for this assignment is July, 3, 2020.** There are 3 questions totally in this assignment and each carries 5 marks. The goal of this programming assignment is to get every student to understand and implement the multithreaded programming, semaphores and resolving deadlocks. Each of the questions of this assignment are provided below:

1. **Multithread Program:** Write a multithreaded C 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

   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.

   Also, create additional threads that determine other statistical values, such as median and standard deviation and also the total number of integers passed.

2. **Semaphore Problem**: Assume that a finite number of resources of a single resource type must be managed. Processes may ask for a number of these resources and will return them once finished. As an example, many commercial software packages provide a given number of ***licenses***, indicating the number of applications that may run concurrently. When the application is started, the license count is decremented. When the application is terminated, the license count is incremented. If all licenses are in use, requests to start the application are denied. Such requests will only be granted when an existing license holder terminates the application and a license is returned.
   The following program segment is used to manage a finite number of instances of an available resource. The maximum number of resources and the number of available resources are declared as follows:

```
#define MAX RESOURCES 5
int available resources = MAX RESOURCES;
```

When a process wishes to obtain a number of resources, it invokes the decrease count() function:

```
/* decrease available resources by count resources */
/* return 0 if sufficient resources available, */
/* otherwise return -1 */
int decrease count(int count) {
if (available resources <count)
return -1;
else {
available resources -= count;
return 0;
}
}
```

When a process wants to return a number of resources, it calls the increase count() function:

```
/* increase available resources by count */
int increase count(int count) {
available resources += count;
return 0;
}
```

The preceding program segment produces a race condition. Do the following:

a) Identify the data involved in the race condition.
b) Identify the location (or locations) in the code where the race condition occurs.
c) Using a semaphore or mutex lock, fix the race condition. It is permissible to modify the decrease count() function so that the calling process is blocked until sufficient resources are available**.**

1. **Bankers Algorithm:** Write a C program to determine the sequence of execution of processes such that the system will not enter the deadlock using Bankers algorithm. The system consists of five processes P_0 through P_4 and three resources types such as A(10 instances), B (5 instances), C(7 instances). The details of already allocated resources at time T_0 and the maximum resources required for completing the execution of these processes are provided in the below table.

	Allocation			**Max**			**Available**		
Process	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**
P_0	0	1	0	7	5	3	3	3	2
P_1	2	0	0	3	2	2			
P_2	3	0	2	9	0	2			
P_3	2	1	1	2	2	2			
P_4	0	0	2	4	3	3			

Submission Instructions:

- Create one main folder that contains three sub folders, which represent the source code of above three questions respectively. Each of the sub folders should contain the source code of the questions, one MS word document capturing the screenshots of the output and the observation if any. Each subfolder should also contain a read file specifying instructions to compile and run and shall also specify which program to be run first in order to see the expected output from your programs.
- The main folder must contain **one main word file** that should include the group information, i.e., **student name, roll no and email id of each student.**
- Zip the main folder and upload the zip file either on CMS or Piazza or mail me by **July 3, 2020 23.59 pm**

Demonstrations:

You will be required to give the demonstration of this assignment online. Each member of the group is required to be presented at the time of demonstration and will be evaluated individually.

Note:

- Plagiarism software would be used to check the similarity scores. If a higher match is found between the submissions of two or more groups, they will be awarded a grade penalty accordingly.
- The submission link for uploading zip file on CMS would be enabled soon for Hyderabad campus students and the students of both Pilani and Goa campus students can mail the zip file or upload on Piazza.

You can drop me a mail for any other queries in this regard or you can also post your query on Piazza as well.