

Assignment 5: Implementing TAS, CAS and Bounded Waiting CAS Mutual Exclusion Algorithms

Goal:- The goal of this assignment is to implement TAS, CAS and Bounded Waiting with CAS mutual exclusion (ME) algorithms studied in the class. Implement these algorithms in C++.

Details:- As shown in the book, you have to implement the three mutual exclusion algorithms in C++. Each mutual exclusion algorithm has an entry and exit sections. To test the performance of mutual exclusion algorithms, develop an application, me-test

(mutual exclusion test) is as follows. Once, the program starts, it creates n threads. Each of these threads, will enter critical section (CS) k times. The pseudocode of the test function is as follows:

Listing 1: main thread

```
1  void main ()
2  {
3      ...
4      ...
5      create  $n$  testCS threads;
6      ...
7      ...
8  }
```

Listing 2: testCS thread

```

1
2 void testCS ()
3 {
4     id = thread.getID ();
5     for (i=0; i < k; i++)
6     {
7         reqEnterTime = getSysTime ();
8         cout << i << "th CS Request at " << reqEnterTime << " by thread " << id;
9         entry-sec ();    // Entry Section
10        actEnterTime = getSysTime ();
11        cout << i << "th CS Entry at " << actEnterTime << " by thread " << id;
12        sleep (t1 );    // Simulation of critical-section
13        exit-sec ();    // Exit Section
14        exitTime = getSysTime ();

15        cout << i << "th CS Exit at " << exitTime << " by thread " << id;
16        sleep (t2 );    // Simulation of Reminder Section
17    }
18 }

```

Here t_1 and t_2 are delay values that are exponentially distributed with an average of λ_1, λ_2 milliseconds. The objective of having these time delays is to simulate that these threads are performing some complicated time-consuming tasks.

Input: The input to the program will be a file, named inp-params.txt, consisting of all the parameters described above: $n, k, \lambda_1, \lambda_2$. A sample input file is: 100 100 5 20.

Output: Your program should output to a file in the format given in the pseudocode for each algorithm. A sample output is as follows:

TAS ME Output:

1st CS Requested at 10:00 by thread 1
 1st CS Entered at 10:05 by thread 1
 1st CS Exited at 10:06 by thread 1
 1st CS Requested at 10:01 by thread 2

.
 .

.

CAS ME Output:

1st CS Requested at 11:00 by thread 1

1st CS Entered at 11:05 by thread 1

1st CS Exited at 11:06 by thread 1

1st CS Requested at 11:01 by thread 2

.

.

.

Bounded CAS ME Output:

1st CS Requested at 11:30 by thread 1

1st CS Entered at 11:35 by thread 1

1st CS Exited at 11:36 by thread 1

1st CS Requested at 11:32 by thread 2

.

.

.

The output should demonstrate that mutual exclusion is satisfied. The output could be in a single file or multiple files.

Report: You have to submit a report for this assignment. This report should contain a comparison of the performance of TAS, CAS and Bounded CAS ME algorithms. You must run these algorithms multiple times to compare the performances and display the result in form of a graph.

For performance, you have to specifically measure two metrics: (1) the average time taken by a process to enter the CS (2) the worst case time taken by a process to enter the CS in a simulation. This shows if processes are starving.

You run these algorithms varying the number of threads from 10 to 50 while keeping other

parameters same. Please have k , the number of CS requests by each thread, fixed to 10 in all these experiments. The graph in the report will be as follows: the x-axis will vary the number of threads. The y-axis will show the metrics defined above: (1) average time taken to enter the CS by each thread (2) the worst case time taken by a process to enter the CS in a simulation.

Also ensure that each point of the graph is obtained **by averaging over five runs**. Finally, you must also give an analysis of the results while explaining any anomalies observed.

Submission Format:- You have to upload: (1) The source code in the following format: Assgn5Src-<RollNo>.c (2) Readme: Assgn5Readme-<RollNo>.txt, which contains the instructions for executing the program. (3) Report: Assgn5Report-<RollNo>.pdf. **Don't zip any of your documents.** Upload all the documents on the blackboard.

Note: Please follow this naming convention mentioned above.

Grading Policy:- The policy for grading this assignment will be - (1) Design as described in the report and analysis of the results: 50% (2) Execution of the tasks based on the description in the readme: 40% (3) Code documentation and indentation: 10%.

Please note:

- All assignments for this course have a late submission policy of a penalty of 10% each day after the deadline of six days. After that, it will not be evaluated.
- **All submissions are subject to plagiarism checks.** Any case of plagiarism will be dealt with severely.