National University of Computer and Emerging Sciences, Lahore Campus

Assignment:2

Course: Operating Systems CS2006,

Weight: 3.3 **Total Marks:15**

Submission deadline: 02-11-2023

Instruction/Notes:

1. Understanding of the problems is part of the assignments. So, no query please.
2. You will get Zero marks if found any type of cheating.
3. 25 % deduction of over marks on the one-day late submission after due date.
4. 50 % deduction of over marks on the two-day late submission after due date. No submission after two days.
5. **MUST BE HANDWRITTEN, IN-CLASS SUBMISSION.**

Question: 1 **[1+1.5+2.5]**

Too much milk Problem definition:

Suppose that we have two invisible roommates sharing a refrigerator. Each roommate acts as a single thread of control, suppose that roommate A and B, buy milk using the following processes:

Roommate: A (Thread A)	Roommate: B (Thread B)

NoteA=TRUE;
while(NoteB==TRUE) ;
 	
if(NoteB==FALSE)
{
if(NoMilk)
{
 	
BuyMilk();
}
 	
}
noteA=FALSE;
	NoteB=TRUE;
	if(NoteA==FALSE)
	{
	
	if(NoMilk)
	{
	BuyMilk();
	
	}
	}
	
	noteB=FALSE;

A- Is there any chance that the two roommates buy too much milk for the house?

B- If yes, prove the above algorithm for all the three necessary conditions of critical section problem Solution.

C- If not, then suggest your solution for the above algorithm which satisfies all the necessary conditions.

Question: 2 **[1+1]**

Differentiate between the following code outputs. Justification is required:

```
using namespace std;
#define NTHREADS 8
void *helloWorld(void *threadid)
{
  long tid;
  tid = (long)threadid;
  cout << "Hello world! Function calling, 00" << tid
<< endl;
  pthread_exit(NULL);
}


int main ()
{
  pthread_t threads[NTHREADS];
  for( int i=0; i < NTHREADS; i++ )
{
    cout << "main: creating thread 00" << i << endl;
    pthread_create(&threads[i], NULL, helloWorld,
(void*)(intptr_t) i);
   void *status;
   pthread_join(threads[i], &status);
  }
pthread_exit(NULL);
}
```

```
using namespace std;
#define NTHREADS 8
void *helloWorld(void *threadid)
{
  long tid;
  tid = (long)threadid;
  cout << "Hello world! Function calling, 00" << tid
<< endl;
  pthread_exit(NULL);
}


int main ()
{
  pthread_t threads[NTHREADS];
  for( int i=0; i < NTHREADS; i++ ){
    cout << "main: creating thread 00" << i << endl;
    pthread_create(&threads[i], NULL, helloWorld,
(void*)(intptr_t) i);
  }
  for(int i=0; i < NTHREADS;
  i++ ){ void *status;
  pthread_join(threads[i], &status);
  }
  pthread_exit(NULL);
}
```

NOTE: Assuming pthread create() and pthread join() all work as expected (i.e., they don’t return an error).

Question: 3 **[4+2+2]**

Process Synchronization - Critical-Section Problem with TestAndSet
Suppose we have an atomic operation TestAndSet(), which works as if it were implemented by pseudocode such as:

```
Boolean test-and-set (boolean &lock)
{
        temp=lock;
        lock=TRUE;
        return temp;
}
```

Here is the function named: **Function1** which claims to satisfy the critical section problem:

```
1:      void Function1(int i, int j, int n)
2:          { boolean key;
3:          while (TRUE)
4:              { waiting[i] = TRUE;
5:              key = TRUE;
6:              while (waiting[i] && key) { key = test-and-set (&lock); }
7:              waiting[i] = FALSE;
8:              {
9:                  // CRITICAL SECTION
                }
                j = (i + 1) % n;
10:             while ( (j != i) && !waiting[j] ) { j = (j + 1) % n; }
11:             if (j == i) { lock = FALSE; }
12:             else {waiting[j] = FALSE; }
                {
13:                 // REMAINDER SECTION
                }
14:         }
15:     }
```

Here are two processes **PA & PB** which used to call function named **Function1**, and with some shared regions:

<table>
<tr><td>Process A</td><td>Process B</td></tr>
<tr><td colspan="2"><b>Memory region shared by both processes:</b>
#define N 2
boolean waiting[N]; // Assume initialized all FALSE
boolean lock = FALSE;</td></tr>
<tr><td>1. #define ME 0
2.int j = 0;
3.<b>Function1</b>(ME, j, N);</td><td>1. #define ME 1
2.int j = 1;
3.<b>Function1</b>(ME, j, N);</td></tr>
</table>

A- The above solution satisfies which necessary or optional requirement of critical section problem Conditions? Justify your answer.

B- What is the purpose of line6 (While Loop) in **Function1() ?**

C- What is the purpose of line10 (While Loop) in **Function1() ?**