Försättsblad till skriftlig tentamen vid Linköpings universitet

Datum för tentamen	2019-03-08
Sal (1)	TER3(26)
Tid	8-12
Utb. kod	TDDD82
Modul	TEN2
Utb. kodnamn/benämning **Modulnamn/benämning**	Projekttermin inklusive kandidatprojekt: Säkra, mobila system Systemprogramvara: Skriftlig tentamen
Institution	IDA
Antal uppgifter som ingår i tentamen	6
Jour/Kursansvarig Ange vem som besöker salen	Mikael Asplund
Telefon under skrivtiden	0700 895 827
Besöker salen ca klockan	9-10
Kursadministratör/kontaktperson (namn + tfnr + mailaddress)	Veronica Kindeland Gunnarsson 013-28 56 34 veronica.kindeland.gunnarsson@liu.se
Tillåtna hjälpmedel	Inga
Övrigt	
Antal exemplar i påsen	

TEN2

Tentamen i TDDD82 Säkra mobila system (Systemprogramvara)

2019-03-08

- Inga hjälpmedel är tillåtna.
- Kom ihåg att svaren på samtliga uppgifter måste MOTIVERAS, och att motiveringarna skall vara uppställda på ett sådant sätt att det går att följa hur Du tänkt. OMOTIVERADE SVAR GER 0 POÄNG OM INGET ANNAT SÄGS.
- Ansvarig: Mikael Asplund (nåbar på tel. 0700-895827).
- Maxpoäng är 30 poäng. För betyg 3 krävs minst 15 poäng, för betyg 4 krävs 20 poäng och för betyg 5 krävs 25 poäng.

Lycka till!!!

1. A public transport company has started a project to enable on-demand bus tansports, guided by how passengers request certain travel needs. One part of this new system is a module that keeps track of the number of passengers currently located on each bus. Listing 1 shows the basic outline of the code running in this module.

Listing 1: BusFleet

```
1  public class BusFleet {
2      final int busCapacity = 60;
3      final int fleetSize = 100;
4      int[] passengers = new int[fleetSize]; //initialized to zeroes
5
6      //add a passenger to bus i if there is room
7      //returns true if a passenger could be added, and false otherwise
8      boolean addPassenger(int i){
9          if (passengers[i] <= busCapacity) {
10             passengers[i] = passengers[i]+1;
11             return true;
12         }
13         return false;
14     }
15
16     //remove a passenger from bus i if there is at least one
17     //returns true if a passenger could be removed, and false otherwise
18     boolean removePassenger(int i){
19         if (passengers[i] > 0) {
20             passengers[i] = passengers[i]-1;
21             return true;
22         }
23         return false;
24     }
25
26     int getTotalPassengerCount() {
27         int count = 0;
28         for (int p : passengers) {
29             count += p;
30         }
31         return count;
32     }
33 }
```

Assume that the methods can be invoked from multiple concurrent threads.

(a) Describe how a race condition can occur in the provided code. Be detailed in describing the trace of execution (e.g., by referring to line numbers of the listings).

(2 points)

(b) Modify the code by inserting synchronization primitives that will prevent race conditions in the code. Your code does not have to be syntactially correct (but you should be clear how variables are initialized). You do not have to repeat all the existing code in your solution, as long as it is clear what changes you want to make.

(2 points)

(c) Depending on your choice of choice of scope for the locking in (b), there might be a way to allow more concurrency (i.e., more simultaneous threads modifying the data at the same time). If you haven't already done so, change your code to provide maximum concurrency.

(2 points)

(d) Discuss how your solution in (c) affects the semantics of the *getTotalPassengerCount* function, and how this could potentially be changed.

(2 points)

2. (a) Name at least 4 important items that are an essential part of the state of a process (i.e., that must be saved and restored at a context switch).

(2 points)

(b) Given the following (Unix) C program:

```
#include <stdio.h>
#include <unistd.h>

int main()
{
   fork();
   fork();
   fork();
   printf("Hello\n");
   return 0;
}
```

How often is `Hello` printed to the standard output when executing this program? Explain your answer carefully (just guessing the right number gives no points).

(2 points)

3. Consider the following resource allocation problem in a system with 3 resources (R1-R3), and 4 processes (P1-P4). The table indicates the currently allocated resources and in parenthesis the maximum possible demand.

	R1	R2	R3
P1	0 (0)	2 (2)	0 (2)
P2	1 (1)	0 (0)	0 (0)
P3	1 (3)	0 (1)	0 (1)
P4	0 (0)	1 (4)	1 (1)

The currently available resources are: [1, 3, 1]. Use Banker's algorithm to determine if the request [1, 0, 1] from Process P3 should be granted.

(5 points)

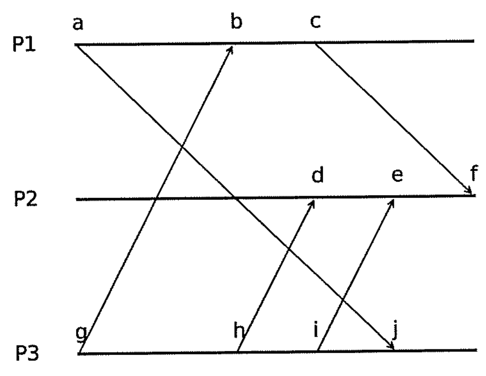
4. (a) Applications can have varying QoS requirements. Explain what it means if an application is elastic or inelastic, and give one example of each type of application.

(2 points)

(b) Describe the main parameters of the token bucket traffic shaping method.

(2 points)

5. Consider the three timelines for nodes P1, P2 and P3 in the figure below. The arrows between the lines indicate messages as they are sent and received by the respective nodes. Use Lamport's logical clock algorithm to set timestamps for the events a-j.



What can you conclude **from these timestamps alone** regarding the following statements:

- $h \rightarrow b$
- $g \rightarrow f$
- $b \rightarrow j$

where $\rightarrow$ indicates the *happened-before* relation?

(5 points)

6. Explain the concepts of *availability* and *reliability*. Also explain how each of these attributes can be measured over time.

(4 points)