Exercise 5 - Synchronization, Resource allocation, Deadlocks

Questions are taken from Stallings, Operating Systems Internals and Design Principles, fifth edition and Silberschatz et al., Operating System Concepts, seventh edition.

1 – Stallings 5.3

Consider the following program:

```
const int n = 50;
int tally;
void total()
{
        int count;
        for (count = 1; count <= n; count++)
        {
                tally++;
        }
}
void main()
{
        tally = 0;
        parbegin (total(), total());
        write (tally);
}
```

a) Determine the proper lower bound and upper bound on the final value of the shared variable `tally` output by this concurrent program. Assume processes can execute at any relative speed and that a value can only be incremented after it has been loaded into a register by a separate machine instruction.

b) Suppose that an arbitrary number of these processes are permitted to execute in parallel under the assumptions of part (a). What effect will this modification have on the range of final values of `tally`?

2 – Stallings 5.17

Show that message passing and semaphores have equivalent functionality by

a) Implementing message-passing using semaphores. *Hint:* Make use of a shared buffer area to hold mailboxes, each one consisting of an array of message slots.

b) Implementing a semaphore using message passing. *Hint:* Introduce a separate synchronization process.

3 – Silberschatz 6.19

A file is to be shared among different processes, each of which has a unique number. The file can be accessed simultaneously by several processes, subject to the following constraint: The sum of all unique numbers associated with all the processes currently accessing the file must be less than n. Write a monitor to coordinate access to the the file.

4 – Silberschatz 6.22

Write a monitor that implements an *alarm clock* that enables a calling program to delay itself for a specified number of time units (*ticks*) You may assume the existence of a real hardware clock that invokes a procedure *tick* in your monitor at regular intervals.

5

Give an example with processes in a deadlock. What is the reason for this condition? What conditions are necessary for a deadlock to occur?

6 – Silberschatz 7.5

In a real computer system, neither the resources available nor the demands of processes for resources are consistent over long periods (months). Resources break or are replaced, new processes come and go, new resources are bought and added to the system. If deadlock is controlled by the banker's algorithm, which of the following changes can be made safely (without introducing the possibility of deadlock), and under what circumstances?

a) Increase *Available* (new resources added).

b) Decrease *Available* (resource permanently removed from system).

c) Increase *Max* for one process (the process needs more resources than allowed; it may want more).

d) Decrease *Max* for one process (the process decides it does not need that many resources).

e) Increase the number of processes.

f) Decrease the number of processes.

7 – Silberschatz 7.7

Consider a system consisting of m resources of the same type being shared by n processes. Resources can be requested and released by processes only one at a time. Show that the system is deadlock free if the following two conditions hold:

a) The maximum need of each process is between 1 and m resources.

b) The sum of all maximum needs is less than $m + n$.

8

In the table below the processes P1, P2, ... and the resources R1, R2, ... are given. An (s) after a resource request specifies that the resource can be shared with other processes that also specified (s), while en (e) indicates exclusive right to the resource. An exclusive resource cannot be shared with another process independent of if the other process specified (e) or (s).

Show by using a resource allocation graph and graph reduction how the processes P1,P2,... can be allowed to execute to completion, that is determine a possible execution sequence for the processes. If a deadlock occurs, show how this condition can be solved by removing one or several processes.

a) Is a deadlock condition present in the following case:

Process	Have resources	Requests resources
P1	R2(e), R10(s), R7(s)	R4(e)
P2	R11(e), R5(e)	R12(s)
P3	R6(s), R7(s), R4(e), R9(e)	R5(s)
P4	R10(s), R1(e), R13(s)	R2(s)

If no deadlock is present, how can the processes execute to completion. If a deadlock is present, can it be solved by killing one of the processes?

b) Is a deadlock condition present in the following case:

Process	Have resources	Requests resources
P1	R3(s), R10(e), R5(s)	R7(s)
P2	R4(s), R8(s)	R7(s)
P3	R7(e), R9(s), R12(e)	R8(e)
P4	R5(s), R9(s), R1(s)	R7(s)
P5	R6(e), R13(s)	R2(e)

If no deadlock is present, how can the processes execute to completion. If a deadlock is present, can it be solved by killing one of the processes?