1467 2017-08-21 Page: 1

TDDB68 TEN2

Multiple choice form for answering question 1. Please put X:s in the appropriate cells:

	A	B	C	D	
1 a)				X	1
1 b)	X	X			1
1 c)		X			1
1 d)	X		X		0
1 e)	X			X	0
1 f)	X		X		0
1 g)	X		X		1
1 h)	X	X			1
1 i)			X	X	0
1 j)	X			X	1

/6p

LINKÖPINGS UNIVERSITET

AID-nummer: 1467	Datum: 2017-08-21	Blad nummer: 2
Kurskod: TDDB68	Provkod: TEN2	

2

a)

```
void fX (int A) {
    wait (condX)
        //Body of fX
        &A = return value
        signal (condX)
}
```

separate these

(dereference)

This ensures that the functions are mutualy exclusive. OK

b)

```
wait(condX){
    while ((test_and_set(condX, false) == false) {
        while (condX == false)
    }
    condX = false
}

int Signal (condX) {
    if(condX == false){
        condX = true }
        return 0
    else {
        return 1
```

semaphore implementation

2

b) The test and set function is atomic and in this case sets cond.X to false while returning its previous value. The while loop after it is just there for performance to limit I/O by limiting writes.

c) The current implementation has no race condition so no changes are needed. X

1,5p

LINKÖPINGS UNIVERSITET

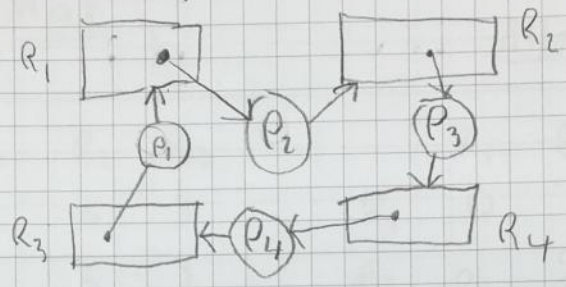
AID-nummer: *AID-number:* 1467	Datum: *Date:* 2017-08-21	Blad nummer: *Sheet number:* 4
Kurskod: *Course code:* TDDB68	Provkod: *Exam code:* TEN1	

3

Deadlocked

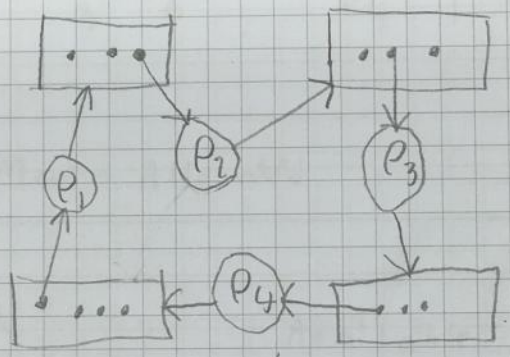
a)

(i)



ok

(ii)



No deadlock

ok, but not 4 instances..

2,5p

b)

3 b) Max

	R_1	R_2	R_3
P_1	0	6	3
P_2	0	5	3
P_3	1	0	6
P_4	1	3	6

Allocated

	R_1	R_2	R_3
P_1	0	1	2
P_2	0	5	3
P_3	0	0	0
P_4	1	2	0

Need = Max - Allocated

Need

	R_1	R_2	R_3	
P_1	0	5	1	$\leq [4,0,1]$ == false
P_2	0	0	0	$\leq [4,0,1]$ == true
P_3	1	0	6	$\leq [4,5,4]$ == false
P_4	0	1	6	$\leq [4,5,4]$ == false

Available = [4, 0, 1]

Available := [4, 5, 4]

We need a new iteration over the need matrix.

Need

	R_1	R_2	R_3	
P_1	0	5	1	$\leq [4,5,4]$ == true
P_2				No need to do for P_2 again
P_3	1	0	6	$\leq [4,6,6]$ == true
P_4	0	1	6	$\leq [4,6,6]$ == true

Available := [4, 6, 6]

Available := [4, 6, 6]

Available := [5, 8, 6]

3 b) Yes this request is save to grant and we get the save sequence:

$P_2 \rightarrow P_1 \rightarrow P_3 \rightarrow P_4$

Ok. You need to check request ∈ available.

Also would be good if you explain which step you are doing.

3.5p

6p

4 a) A process is an active entity executing its program. A user level thread does the same thing, IE executing code and following its instructions. The difference comes in how they are used, a process can have many user-level threads executing different parts of the programs code.

A process can exist without any user level threads a user level thread must always belong to a process.

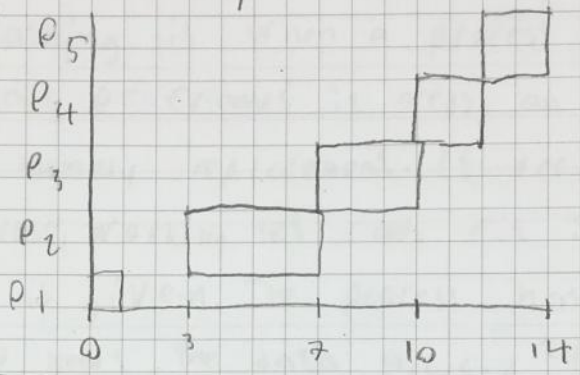
A kernel-level thread is used to map the user level threads and processes to a processor. The processor can only schedule kernel-level threads so mapping to them is essential. While user-level threads are thought to be parallel, if they map to the same kernel thread which is blocked, they both get blocked (both user-level threads).

user-level

All threads belonging to the same process share many data structures where as two different processes share no data by default.

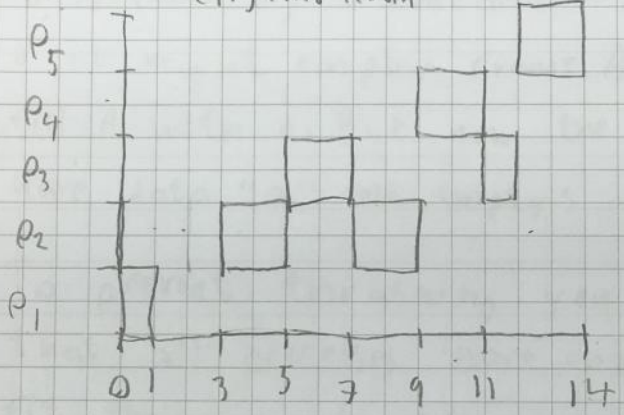
4 b)

(i) FIFO



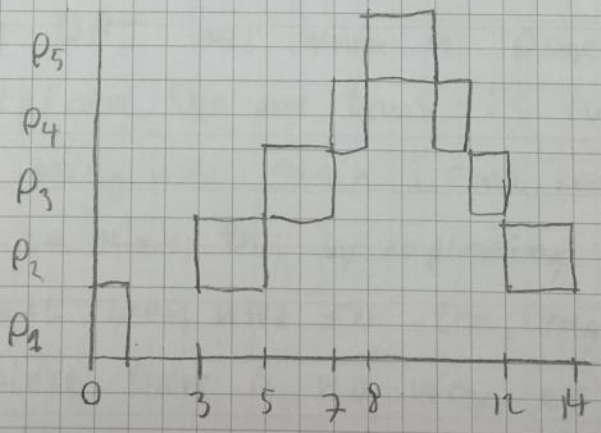
1

(ii) Round Robin



1

(iii) priority scheduling with preemption



1

(3)

5 a) Thrashing is when a process replaces the content of frames it needs, an endless loop of memory replacement. It occurs when the process' working set can't fit into the memory it is given. The process has 3 frames A, B, C and needs the data a, b, c, d. If A contains a, B contains, b, C contains c. To fit the data d it might empty frame A of data "a" and fill A with d. But now the processes doesn't have data "a" and empty's another frame. And so on...

To prevent thrashing you can make sure that all processes have enough memory to fit their working set. (2)

b) The optimal page replacement algorithm is called OPT and when a page is to be replaced it replaces the one that is furthest away from being used again. Least recently used (LRU) tries to mimic this by replacing the page that has not been used for the longest time. It does this since there is no way of knowing which page will be used next unless we can see the future. (2)

6 a) The reason for opening a file is to find it more easily. All open files are saved in a list so to find them we just go through this list. If this was not the case every operation called on a file would have to find it first, wasting time.

The open systemcalls adds the file to the kernels list of open files and returns a handle for the file.

(2)

b) Having a single copy of the file saves space and allows for instant updates if the file changes. If the file is shared through an acyclic graph system the system would need to continuously check for cycles with for example a garbage collector, which takes time and resources.

If the file is used by many users who often want write access to the file, having a single file would likely cause conflicts.

LINKÖPINGS UNIVERSITET

AID-nummer: *AID-number:* 1467	Datum: *Date:* 2017-08-21	Blad nummer: *Sheet number:* 11
Kurskod: *Course code:* TDDB68	Provkod: *Exam code:* TEN1	

6.

b) Letting users have their own version of the file is the default implementation in Linux and windows, so this implementation is likely easier than the other.

1.5