Lösningsskisser, tenta OS 2017-03-22

1a) Virtuell maskin betyder att OS ger processen illusionen av en maskin som inte är den fysiska maskinen. Ursprungligen att processen fick illusionen av att vara ensam om hårdvaran fast hårdvaran i själva verket delades av flera processer.

1b) Utökad virtuell maskin betyder att OS ger processen illusionen av att datorn har mer avancerad funktionalitet än vad hårdvaran faktiskt har.

1c) OS:et styr och kontrollerar hur processer får använda datorns resurser. Exempel på resurser är processorkraft, minne, I/O-enheter, lagringsenheter.

1d) När ett systemanrop görs utförs en TRAP eller motsvarande instruktion som gör att processorn går över från user mode till supervisor mode. Det första som sker är att OS:et kontrollerar om den anropande processen får göra detta systemanrop med de parametrar som medföljer anropet. Om allt är OK så utförs systemanropet av OS-kärnan. Vid återgång från ett systemanrop ändras statusregistret så att processen återgår till user mode. Syftet med systemanrop är att skydda OS:et men att ändå låta processer på ett kontrollerat sätt kunna komma åt OS:ets funktioner. Det är inte fel att nämna att man oftast byter stack till en särskild kärn-stack när systemanropet genomförs, för att inte riskera problem med sidfel i användarminnet.

2a) Pseudoparallellism är när OS:et kör flera processer ”samtidigt”, en liten snutt i taget på varje. Det upplevs av användaren som att processerna kör parallellt, men det är en illusion.
b) Relokerbarhet är förmågan att flytta processer i minnet. Relokering kan vara statisk (processer blir kvar där man laddat den) eller dynamisk (processen kan flyttas under exekvering).
c) En räknande semafor kan låta mer än en process göra wait på den. Semaforen initieras till N, då kan N processer göra wait samtidigt utan att blockeras.
d) Intern fragmentering är när minnesresurser är allokerade till processer men processerna inte använder det allokerade minnet.
e) Osäkert tillstånd är ett tillstånd som riskerar att leda till ett baklås. Osäkra tillstånd bör undvikas.
f) Busy waiting innebär att processen ligger och testar oupphörligt om något visst villkor uppfyllts (exv. att en variabel skall få ett visst värde).
g) Thrashing är när alltför lite nyttigt arbete blir gjort i ett system på grund av att processerna spenderar alltför mycket av sin exekveringstid med att läsa in kod och data som inte finns kvar i minne. Problemet är för lite minne och/eller för många processer och/eller för kort tidskvantum.
h) Asynkront meddelandesystem innebär att en sändare lämnar meddelandet i en buffert där mottagaren sedan avhämtar det. Sändaren behöver alltså inte vänta på att mottagaren är redo att emot meddelandet.

3a) Trådar saknar egen adressrymd och saknar skydd mellan trådar som exekverar i samma adressrymd.
b) Det är mindre resurskrävande att starta en ny tråd i en existerande process än att starta en ny process. Kommunikationen är enklare mellan trådar i samma process.

4 En bitmapp eller en länkad lista av lediga block. Se bokens avsnitt 4.4.1.

5a) Preemptiv SJF innebär att en exekverande process kan avbrytas om det kommer in ett jobb med kortare exekveringstid. Exekveringen blir:
A B C A A D D D D D F E E E

b) Medelomloppstiden blir ((5-0) + (2-1) + (3-2) + (10-2) + (14-8) + (11-10))/5 =
= (5 + 1 + 1 + 8 + 6 + 1)/6 = 22/6

5c) Vid en tidpunkt då aktivering av ny process sker kan antingen en nyaktiverad process ställas sist i readykön *innan* den process som just varit Running ställs sist i readykön, eller tvärtom. Se tenta 2017-06-07 för en mer specifik probleminstruktion. Båda varianterna är OK, men lösningen måste vara konsekvent, d.v.s. alla aktiveringar måste följa samma princip.

Variant 1: nyaktiverad ställs sist *innan* running ställs sist.

Tid	Running	Readykön	Omloppstid
0	A	-	
1	B	A	
2	A	C D	B klar: 2-1 = 1
3	C	D A	
4	D	A	C klar: 4-2 = 2
5	A	D	
6	D	-	A klar: 6-0 = 6
7	D	-	
8	E	D	
9	D	E	
10	E	F D	
11	F	D E	
12	D	E	F klar: 12-10 = 2
13	E	-	D klar: 13-2 = 11
14	-	-	E klar: 14-8 = 6

Medelomloppstid: 28/6

Variant 2: running ställs sist innan nyaktiverad ställs sist.

Tid	Running	Readykön	
0	A	-	
1	A	B	
2	B	A C D	
3	A	C D	B klar: 3-1 = 2
4	C	D	A klar: 4-0 = 4
5	D	-	C klar: 5-2 = 3
6	D	-	
7	D	-	
8	D	E	
9	E	D	

10	D	E F	
11	E	F	D klar: 11-2 = 9
12	F	E	
13	E	-	F klar: 13-10 = 3
14	-		E klar: 14-8 = 6

Medelomloppstid: 27/6

e) Multipla köer: algoritmen är preemptiv, men uppgiften säger inget om ifall en nyaktiverad process avbryter en process med lägre prioritet mitt i dess tidskvantum eller inte. Gör endera antagandet och var konsekvent.

För båda alternativen gäller:

Tid	Running	HÖG	MELLAN	LÅG	
0	A	-	-	-	
1	B	-	A	-	
2	C	D	A	-	B klar: 2-1 = 1
3	D	-	A D	-	C klar: 3-2 = 1
4	A	-	D	-	
5	A	-	D	-	
6	D	-	-	-	A klar: 6-0 = 6
7	D	-	-	-	
8	E	-	-	D	
9	E	-	-	D	

Alt 1: E i kö MELLAN får köra hela kvantat 2 enheter fast F hamnar i kö HÖG efter 1 enhet

10	E	F	-	D	
11	F	-	-	D	E klar: 11-8 = 3
12	D				F klar: 12-10 = 2
13	D				
14	-				D klar: 14-2 = 12

Medelomloppstid: 25/6

Alt 2: F avbryter E efter det att E exekverat 1 enhet i kö MELLAN

10	F	-	E	D	
11	E	-	-	D	F klar: 11-10 = 1
12	D				E klar: 12-8 = 4
13	D				
14	-				D klar: 14-2 = 12

Medelomloppstid: 25/6

Uppgift 6:

a)
E = (6 6 6 6)
A = (2 0 3 0)

C = 0 0 0 3
1 1 0 0
3 3 3 0
0 2 0 3

R = 2 3 2 0
2 0 2 0
0 2 0 4
3 1 0 0

b) Nej, systemet är inte i baklås. Motivation:
p2 kan köra, efter detta blir A' = (3 1 3 0)
p4 kan köra, efter detta blir A'' = (3 3 3 3)
p1 kan köra, efter detta blir A''' = (3 3 3 6)
p3 kan köra, efter detta blir A'''' = (6 6 6 6)
Alla kunde alltså köra klart.

Uppgift 7:
a) Bitar för index: 4 bitar. Bitar för offset: 3 bitar
b) Rita en tabell för varje process. Använd binära minnesadresser.
c) För A: Virtuella sidan är 0101 (5) och offset är 101 (5). På index 5 i sidtabellen för A finns ram nummer 3 (011). Den fysiska adressen blir då 011101 (ramnummer följt av offset).
För B: För A: Virtuella sidan är 0101 (5) och offset 101 (5). På index 5 i sidtabellen för B finns ram nummer 7 (111). Den fysiska adressen blir då 111101.

Uppgift 8:
a) Bestäm storleken på en inod först. Om den kan hålla minst 5 block blir det enklare att lösa uppgiften. Lämpligt att allokera från block 0 och framåt. Det innebär att (vi skippar delstegen här, se till att era tentasvar redovisar alla delsteg):
inoden för A pekar på block 0, 10, 11
inoden för B pekar på block 1, 2, 3, 6, 7
inoden för C pekar på block 4, 5, 8, 9
b)
(Någonstans finns tre directoryentryn som pekar ut första blocket: A: 0, B: 1, C:4.)
FAT-tabellen ser (efter alla steg, se till att era tentasvar redovisar alla delsteg) ut som (E betyder slut på filen):
0: 10
1: 2
2: 3
3: 6
4: 5
5: 8
6: 7
7: E
8: 9
9: E
10: 11
11: E