PAMUKKALE ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

CENG 305 – İşletim Sistemleri

2018-2019 EĞİTİM ÖĞRETİM YILI GÜZ DÖNEMİ DÖNEM SONU SINAVI

Süre:75dk

Öğrenci No: **Ad:** **Soyad:**

İmza: **09.01.2019**

PAÜ CENG 305 İşletim Sistemleri Dönem Sonu Sınavı

Soru	1 (20p)	2 (20p)	3 (20p)	4 (20p)	5(25p)	Toplam
Puan						

1.

a) Aşağıdaki ifadeleri doğru yanlış olarak **nitelendiriniz** ((D) ya da (Y) olarak.).

(D)İşletim sistemleri genel olarak deadlock ile ilgili önlem almayıp, deadlock ile ilgili önlemleri programcılara bırakır.

(D)Page size büyüklüğündeki artma daha az TLB miss oluşumuna neden olur.

(D)Critical section (Kritik bölge) programın daha serileşmesine neden olur

(Y?)Page fault aranılan değişkenin cache bellekte bulunamamasıdır.

(Y) Belady anomaly disk okuması sırasında yaşanılan bir sıkıntıdır.

(Y)Semaphore kullanımı işletim sisteminin hızını doğrudan yükseltir.

(D)Virtual memory sistemin sahip olduğu bellekten daha fazla belleği varmış gibi kullanılmasını sağlar.

(D)Temporal locality durumundan istifade edilerek bellek erişiminden kaynaklanan gecikmeler azaltılmaktadır.

(Y)Processler arası iletişim her zaman tek yönlüdür.

(D)Hyperthreading ile işlemcilerin daha yüksek performansta çalışmaları sağlanmaktadır.

2. a. 32 sayfadan oluşan ve her sayfada 1024 word (hücre) bulunan bir sanal belleğin 16 frame'den oluşan bir fiziksel belleğe map edildiğini varsayalım. Mantıksal adres kaç bit olmalıdır? Fiziksel adreste kaç bit olmalıdır?

2^5=32
2^10=1024
15 bit

frame-page ayni olmalidir
2^4=16
2^10=1024
14 bit

b. Bir processin ulaşmak istediği bir adresin main memoryde olmadığını varsayalım (Belki de **demand paging** yüzünden). İşletim sisteminin **page faultu** gidermek için gerçekleştirmesi gerektiği adımları listeleyiniz.

1. Operating system looks at another table to decide:
   - Invalid reference -> abort
   - Just not in memory
2. Get empty frame
3. Swap page into frame
4. Reset tables
5. Set validation bit = v
6. Restart the instruction that caused the page fault

3. ***(Virtual Memory)*** Bir processin 4 tane frame'den oluşan bir main memory'de çalıştığını düşünelim. Verilen referans string'ine göre aşağıdaki algoritmalar çalıştırıldığında her sayfa referansında main memorynin durumunu ve page fault (sayfalama hatalarını) sayılarını belirtiniz. (Demand paging kullanıldığı varsayılmaktadır.)

0, 1, 7, 0, 1, 2, 0, 1, 2, 3, 2, 7, 1

a. FIFO

0	0	0	0	3
-	1	1	1	1
-	-	7	7	7
-	-	-	2	2

FIFO'da 5 adet page fault

b. LRU

0	0	0	0	0	7
-	1	1	1	1	1
-	-	7	7	3	3
-	-	-	2	2	2

LRU'da 6 adet

c. Optimal Algoritma

0	0	0	0	3
-	1	1	1	1
-	-	7	7	7
-	-	-	2	2

Optimal'de 5 adet

4. Bir sistemin t anındaki durumu aşağıda verilmiştir.

	Allocation	*Max*	*Available*
	A B C D	A B C D	A B C D
P_0	0 0 1 2	0 0 1 2	1 5 2 0
P_1	1 0 0 0	1 7 5 0	
P_2	1 3 5 4	2 3 5 6	
P_3	0 6 3 2	0 6 5 2	
P_4	0 0 1 4	0 6 5 6	

a. Sistem bu durumda safe durumda mıdır?

		A	B	C	D
0012-0012	P0	0	0	0	0
1750-1000	P1	0	7	5	0
2356-1354	P2	1	0	0	2
0652-0632	P3	0	0	2	0
0656-0014	P4	0	6	4	2

P0 1520
0012 safe
+---
1532
1354
+---
P0->P2 2886

P3-P4-P1
hata

b. P1'in (0,4,2,0) isteği yerine getirilmeli midir?

1520-0420=1100

P0 1100
0012
+---
1112
1354
+---
P0->P2 2466

P0->P2->(P1->P3->P4)

		A	B	C	D
0012-0012	P0	0	0	0	0
1750-0420	P1	0	3	3	0
2356-1354	P2	1	0	0	2
0652-0632	P3	0	0	2	0
0656-0014	P4	0	6	4	2

5. ***(Process Syncronization)*** Doğadaki su oluşumunu simüle eden bir sistem oluşturmanız istenmektedir. Su oluşturma tepkimesi için **iki adet H** atomu **bir adet de O** atomu gerekmektedir. Sistemimizde **atomlar threadler** ile temsil edilecektir. Bir **alanda** iki adet H ve bir adet O olduğu anda alana **son giren thread suOlustur()** metodunu çağıracak ve alana yeni gelen atomlar su tepkimesini tekrardan oluşturabilecektirler. Bu sistemi gerçekleştiren yapıyı **semaforlar kullanarak** gerçekleştiriniz (sözde kodunu aşağıda veriniz). Oluşturduğunuz sistemde **deadlock oluşabilir mi** tartışınız. Başarılar

```
//"main" Class'i
        Semaphore alan = new Semaphore(3,3);
        tepkime myTepkime = new tepkime();
        loop
                Thread thread = new thread(myTepkime("O"or"H"))
                olusturulan threadler 2H ve 1O olacak sekilde alan semaphor'u yerlestirilir
                alan (3,3) oldugunda suOlustur(Semaphore alan) mehodu cagirilir

//"tepkime" Runnable class'i
        element alan construct olustur
        run()
                tepkime olusumu

//"suOlustur(Semaphore alan)" moethod'u
        alan semaphore icersindeki thread'ler calistirilir
        ".start()"
        ".join()"
        alan.relase(3)


//"H" Class'i


//"O" Class'i
```