|  |  |  |
| --- | --- | --- |
|  | PAMUKKALE ÜNİVERSİTESİ**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ** **CENG 305 – İşletim Sistemleri 2016-2017 EĞİTİM ÖĞRETİM YILI GÜZ DÖNEMİ DÖNEM SONU SINAVI** |  |

**Öğrenci No: Ad: Soyad:**

**İmza: Süre:80dk 02.01.2017**

**PAÜ CENG 305 İşletim Sistemleri Dönem Sonu Sınavı**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Soru** | **1 (20p)** | **2 (20p)** | **3 (20p)** | **4 (20p)** | **5(20p)** | **Toplam** |
| **Puan** |  |  |  |  |  |  |

**Sonuçları kare içine alınız!**

1. Demand-paged memory (istenildiğinde sayfanın belleğe getirildiği) kullanıldığı ve sayfa tablosunun (*page table*) registerlerda tutulduğu varsayılsın. Eğer bellekte boş bir *frame* varsa veya üzerine yazılacak olan sayfa değiştirilmemişse (modifiye edilmemişse) *page fault* olayını çözmek 8 **milisaniye** tutuyor. Eğer değiştirilecek sayfanın üzerinde bir değişiklik yapılmışsa page fault’u çözmek 20 **milisaniye** sürüyor. Belleğe erişim 100 **nanosaniyedir**. *Page fault* için yerine başka bir sayfa getirilecek olan sayfa (üzerine yazılan sayfa), %70 oranla içeriği değiştirilmiş bulunuyor. Buna göre efektif erişim süresinin 200 **nanosaniyeden** daha yüksek çıkmaması için en yüksek *page-fault* oranı ne olmalıdır? (**Not : page fault çözüm süresi içerisinde belleğe erişim de bulunmaktadır, sayfa tablosuna erişimi 0 saniye kabul edebilirsiniz.)**
2. Manyetik diskinizin 5000 tane silindiri olduğunu düşünün. Bu silindirler 0’dan 4999’a kadar numaralandırılmış olsun. Okuma/yazma kafası şu anda 143’üncü silindirde okuma işlemi gerçekleştirmektedir. Bundan bir önceki okuma işlemi ise 125’inci silindirde gerçekleştirilmiştir. Bekleyen istekler FIFO sırasına göre 86, 1470, 913, 1774, 948, 1509,1022,1750 ve 130’dur. Disk okuma/yazma kafasının şu andaki pozisyonunda olduğunu düşünerek tüm bu isteklerin yerine getirilmesi için aşağıdaki *disk-scheduling* algoritmaları kullanıldığında, disk okuma/yazma kafasının toplamda ne kadar hareket etmesi gerekir (silindir sayısı cinsinden), silindirler hangi sıra ile okunur?
3. FCFS
4. SSTF
5. SCAN
6. LOOK
7. C-SCAN
8. Bir dosyanın, bu dosyaya erişen *linklerin* (bağlantıların) kaldırılmadan silinebildiği ve bu dosyanın depolama alanının işletim sistemine geri verilebildiği bir dosya sistemi düşünün. Eğer silinen bir dosyanın depolama alanı üzerinde yeni bir dosya oluşturulursa hangi problemler oluşabilir? Bu problemler nasıl önlenebilirdi? Açıklayınız.
9. Aşağıdaki sayfa referans dizisi (**page reference string**) verilmiştir.

1,2,3,4,2,1,5,6,2,1,2,3,7,6,3,2,1,2,3,6

Sırasıyla bellekte toplamda 4,5, 6 ve 7 frame olduğu durumlarda aşağıdaki yerleştirme algoritmaları (page replacement algorithms) kullanıldığında yukarıdaki sayfalara erişim toplamda kaç tane sayfa hatası (page fault ) oluşturur? Başta tüm framelerin boş olduğunu unutmayınız.

1. LRU
2. FIFO

1. Optimal Replacement
2. a. Yemek çubuklarının (*chopstick*) masanın ortasında bulunduğu bir dining philosopher (akşam yemeği yiyen filozoflar 5 yemek çubuğu ve 5 filozof için) örneği düşünününüz. Ancak burada yemek yeme işlemi için **üç** yemek çubuğuna ihtiyaç vardır. Kaynak ataması (yani bir anda sadece bir yemek çubuğunun isteyen bir filozofa atanması) bir anda bir tane olacak şekilde gerçekleştirilmelidir. Bu değiştirilmiş dining philosophers probleminde kaynakların *deadlock* oluşmayacak şekilde tahsis edilmesini sağlayacak basit bir kural bulup açıklayınız.
3. *Thrashing* neden oluşmaktadır? Açıklayınız. Sistem *thrashing* olduğunu nasıl farkeder? İşletim sistemi *thrashingi* farkettiğinde bu problemi gidermek için ne yapabilir?