

## BİL 301 İşletim Sistemleri 2025 Güz Dönemi Final Sınavı

- **Fotoğraflı üniversite kimliği** olmayan öğrenciler sınava giremez. Her türlü kağıt, kitap, not, hesap makinesi, telefon, saat vb. elektronik aygıt kullanımı **kopya** sayılır.
- TOPLAM SÜRE 90 DAKİKADIR. Toplam 105 puan üzerinden değerlendirme yapılacaktır. 5 puan bonus olarak düşünülmüştür.
- Cevaplarınızı okunaklı bir şekilde ayrılan kısımlara yazınız. İngilizce açıklamalar konu anlaşılabilirliğini artırmak içindir.

İsim ve Soyisim : \_\_\_\_\_

Öğrenci No : \_\_\_\_\_

İmza : \_\_\_\_\_

Question	Points	Score
1	3	
2	6	
3	3	
4	3	
5	3	
6	6	
7	3	
8	12	
9	6	
10	9	
11	9	
12	9	
13	3	
14	3	
15	3	
16	3	
17	3	
18	6	
19	3	
20	9	
Total:	105	

Doğru olanların işaretlendiği sorularda her yanlış seçim bir doğruyu götürmektedir.

## Bölüm 1: Synchronization (Senkronizasyon)

### 1. 3 pt Peterson's Algorithm

```
while (true) {  
    flag[i] = true;  
    turn = j;  
    while (flag[j] && turn == j)  
        ; // busy wait  
    /* critical section */  
    flag[i] = false;  
    /* remainder section */  
}
```

Peterson's Algorithm ile ilgili aşağıdaki ifadelerden hangileri doğrudur? (Doğru olanları işaretleyin: Her yanlış bir doğruyu götürmekte)

- ☒ Compiler ve CPU optimizasyonları sebebiyle mutual exclusion garanti edilemez.
- ☒ Sadece iki process/thread için çalışır.
- ☒ Memory barrier kullanılmazsa mutual exclusion garantisi bozulabilir.
- ☒ Bounded waiting koşulunu sağlar.

### 2. Lock

```
int lock = 0;  
  
void acquire_lock() {  
    while (test_and_set(&lock) == 1)  
        ; // spin  
}  
  
void release_lock() {  
    lock = 0;  
}
```

(a) 3 pt Bu lock türü nedir?

- A. Mutex
- B. Semaphore
- C. Spinlock**
- D. Monitor
- E. Futex

(b) 3 pt Bu lock'ın dezavantajları nelerdir? (Doğru olanları işaretleyin: Her yanlış bir doğruyu götürmekte)

- ☒ CPU cycle'larını boşa harcar
- ☐ Priority inversion'a sebep olur
- ☒ Single CPU sistemlerinde verimsizdir
- ☐ Deadlock'a yol açar
- ☒ Cache coherence problemlerine sebep olabilir

3. 3 pt Semaphore vs Mutex - Aşağıdaki ifadelerden hangileri doğrudur? (Doğru olanları işaretleyin: Her yanlış bir doğruyu götürmekte)

- ✓ Mutex sadece iki durumlu (0/1) iken semaphore sayma değişkenidir.
  - ✓ Mutex lock'ı sadece lock'ı alan thread bırakabilir.
  - ✓ Semaphore'da wait ve signal işlemleri farklı thread'ler tarafından yapılabilir.
  - Binary semaphore ile mutex tamamen aynı şeydir.
4. 3 pt **Deadlock Conditions** - Aşağıdakilerden hangisi deadlock'un koşullarındandır? (Doğru olanları işaretleyin: Her yanlış bir doğruyu götürmekte)
- ✓ Mutual Exclusion
  - ✓ Hold and Wait
  - ✓ No Preemption
  - ✓ Circular Wait
  - Priority Inversion
5. 3 pt **RCU (Read-Copy-Update)** tekniği için hangileri doğrudur? (Doğru olanları işaretleyin: Her yanlış bir doğruyu götürmekte)
- ✓ Reader'lar arasında senkronizasyon gerektirmez
  - ✓ Writer'lar arasında geleneksel lock mekanizmaları kullanılır
  - Reader'lar yazma işlemi sırasında bloke olur
  - ✓ Eski versiyonlar garbage collection ile temizlenir
  - ✓ Read-mostly senaryolarda verimlidir
6. **Memory Ordering** Aşağıdaki kod parçasında memory ordering ile ilgili soruları cevaplayınız:

```
struct data {
    int value;
    int timestamp;
};

_Atomic(struct data*) shared_data;
_Atomic(int) ready;

void writer(struct data* new_data) {
    new_data->timestamp = get_current_time();
    /* Region A */
    atomic_thread_fence(memory_order_release);
    /* Region B */
    atomic_store_explicit(&shared_data, new_data, memory_order_relaxed);
    /* Region C */
    atomic_store_explicit(&ready, 1, memory_order_relaxed);
    /* Region D */
}

struct data* reader() {
    int is_ready = atomic_load_explicit(&ready, memory_order_relaxed);
    /* Region X */
    if (!is_ready) return NULL;
    /* Region Y */
    atomic_thread_fence(memory_order_acquire);
    /* Region Z */
    return atomic_load_explicit(&shared_data, memory_order_relaxed);
}
```

- (a) 3 pt Hangi bölgelerdeki okuma/yazma işlemleri kendi aralarında sıralanabilir (reorder)?  
(Which regions can have their read/write operations reordered within the region?)

(a) **Bölgeler C, D, X, Y (Regions C, D, X, Y)**

- (b) 3 pt Aşağıdaki ifadelerden hangileri doğrudur? (Doğru olanları işaretleyin: Her yanlış bir doğruyu götürmekte)
- ☒ **Region A'daki işlemler Region B'den sonraya atılamaz.**
  - ☐ Region C'deki işlemler Region B'den önceye alınabilir.
  - ☒ **Region X'teki işlemler Region Y'den sonraya atılabilir.**
  - ☐ Region Z'deki işlemler Region Y'den önceye alınabilir.
  - ☒ **Reader fonksiyonu her zaman tutarlı (consistent) bir data yapısı döndürür.**

7. 3 pt **Lock Implementation** - Aşağıdaki ifadelerden hangileri doğrudur? (Doğru olanları işaretleyin: Her yanlış bir doğruyu götürmekte)
- ☒ **Kernel'da interrupt disable/enable ile lock implemente edilebilir.**
  - ☒ **User space'de futex (fast userspace mutex) system call kullanılabilir.**
  - ☒ **Spinlock'lar çok kısa kritik bölgeler için uygundur.**
  - ☒ **Ticket lock starvation'ı önler.**

## Bölüm 2: Virtual Memory (Sanal Bellek)

### 8. Page Replacement Algorithms

Aşağıdaki page reference string'i göz önüne alınız (her sayfa 0-9 arasında):

7, 3, 5, 1, 2, 5, 6, 4, 8, 3, 2, 5, 7, 1, 4

**3 frame** ile demand paging kullanıldığında:

- (a) 3 pt **FIFO** algoritması için:

Reference	Frames			Page Fault?
	Frame 1	Frame 2	Frame 3	(✓ veya ×)
7				
3				
5				
1				
2				
5				
6				
4				
8				
3				
2				
5				
7				
1				
4				

Toplam page fault sayısı: 14

- (b) 3 pt **Optimal** algoritması için:

Reference	Frames			Page Fault?
	Frame 1	Frame 2	Frame 3	(✓ veya ×)
7				
3				
5				
1				
2				
5				
6				
4				
8				
3				
2				
5				
7				
1				
4				

Toplam page fault sayısı: **12**

(c) **3 pt** LRU algoritması için:

Reference	Frames			Page Fault?
	Frame 1	Frame 2	Frame 3	(✓ veya ×)
7				
3				
5				
1				
2				
5				
6				
4				
8				
3				
2				
5				
7				
1				
4				

Toplam page fault sayısı: **14**

(d) **3 pt** Clock algoritması hangi algoritmanın yaklaşık (approximate) implementasyonudur?

(d) **LRU**

## 9. TLB and Multi-level Page Tables

(a) **3 pt** Page Table Structures - Aşağıdaki ifadelerden hangileri doğrudur? (Doğru olanları işaretleyin: Her yanlış bir doğruyu götürmekte)

- ✓ Multi-level page table sparse address space (seyrek adres uzayı) için verimlidir.
- ✓ Inverted page table (PID'lerle tüm processler için tek table) physical memory boyutu ile sınırlıdır.
- ✓ TLB hit (tlbde bulunma) oranı arttıkça efektif memory erişim süresi azalır.

- ☐ TLB hit (tlbde bulunma) oranı arttıkça efektif memory erişim süresi artar.  
☒ **Context switch sırasında TLB flush edilmelidir.**
- (b) 3 pt 32-bit sistemde 4KB page size, 2-level page table (10-10-12) kullanılıyorsa:  
 a) Page table entry boyutu 4 byte ise, bir process'in maksimum page table memory kullanımı nedir?  
 (b) 4KB (inner) + 4KB (outer) = 8KB

10. **Thrashing and Working Set Model** Bir sistemde aşağıdaki gözlemler yapılmıştır:

- CPU kullanımı %20'nin altında
- Disk I/O oranı çok yüksek
- Process'ler sürekli page fault veriyor

- (a) 3 pt Bu durumun adı nedir?  
 (a) Thrashing
- (b) 3 pt Bu sorunu çözmek için hangi önlemler alınabilir? **(Doğru olanları işaretleyin: Her yanlış bir doğruyu götürmekte)**  
☒ **Process sayısını azaltmak**  
☐ Process sayısını artırmak  
☒ **Fiziksel belleği artırmak**  
☐ Fiziksel belleği azaltmak  
☒ **Working set model kullanarak memory allocation optimize etmek**  
☐ Working set model kullanarak processler arası haberleşmeyi optimize etmek
- (c) 3 pt Aşağıdaki belirtilerden hangisi thrashing'i gösterir? **(Doğru olanları işaretleyin: Her yanlış bir doğruyu götürmekte)**  
☒ **CPU utilization düşük, disk I/O yüksek**  
☒ **Sürekli page fault oluşması**  
☒ **Process'lerin çoğunun waiting durumunda olması**  
☐ TLB hit (bulunma) oranı (rate)'in yüksek olması  
☒ **Response time (cevap verme zamanı)'nın artması**

## Bölüm 3: I/O Systems and File Systems

11. **I/O Methods and Performance**

- (a) 3 pt Aşağıdaki I/O yöntemlerini açıklayınız:  
 a) **Polling:**  
 (a) CPU sürekli device status kontrol eder
- b) **Interrupt-driven:**  
 (a) Device hazır olduğunda CPU'yu interrupt eder
- c) **DMA:**  
 (a) Device direkt memory'ye erişir, CPU'yu rahatsız etmez, interrupt sadece ba
- (b) 3 pt Aşağıdaki ifadelerden hangisi I/O (Giriş/Çıkış) yöntemleri ve DMA hakkında doğrudur?  
☐ DMA, Polling ve Interrupt-Driven ile birlikte kullanılamaz  
☐ Polling yönteminde, aygıt işlemi tamamlayınca CPU'ya kesme gönderir  
☐ Interrupt-Driven yönteminde, CPU sürekli olarak aygıtın durumunu kontrol eder  
☒ **Polling ve Interrupt-Driven aygıt senkronizasyonu için kullanılır; DMA ise veri transferi için kullanılır**

(c) 3 pt Aşağıdaki durumlar için hangi I/O yöntemi uygundur?

a) Yüksek hızda sürekli data transferi (video streaming)

(c) DMA

b) Nadiren tetiklenen kullanıcı girişi (klavye)

(c) Interrupt-driven

c) Real-time sistemde deterministik davranış gerektiren durum

(c) Polling veya hybrid

12. **Disk Scheduling** Disk request'leri: 98, 183, 37, 122, 14, 124, 65, 67 Başlangıç head pozisyonu: 53

Aşağıdaki algoritmalar için erişim sırasını ve toplam seek mesafesini hesaplayınız:

(a) 3 pt **FCFS:**

(a) Sıra: 98,183,37,122,14,124,65,67; Mesafe: ...

(b) 3 pt **SSTF:**

(b) Sıra: 65,67,37,14,98,122,124,183; Mesafe: ...

(c) 3 pt **SCAN** (yukarı veya aşağı doğru başladığını varsayabilirsiniz, her iki yönde gidiyor):

(c) Sıra: 65,67,98,122,124,183,37,14(her iki yonede tariyor); Mesafe:...

13. 3 pt **RAID Levels(Partial List)** - Aşağıdaki ifadelerden hangileri doğrudur?

✓ RAID 0 striping kullanır, fault tolerance yoktur.

✓ RAID 1 mirroring kullanır, çoklu kopya olduğu için kapasite verimsizdir.

14. 3 pt **File System Implementation** Aşağıdaki file allocation metodlarını rastgele (random) ve sıralı (sequential) erişim işlemleri ve fragmentation bakımından karşılaştırmız:

a) **Contiguous allocation:**

14. Hızlı rastgele erişim, external fragmentation

b) **Linked allocation:**

14. External fragmentation yok, rastgele erişim yavaş

c) **FAT allocation:**

14. same as linked allocation but faster/efficient since it uses TABLE and stores

**Solution:** same as linked allocation but faster/efficient since it uses TABLE and stores it in memory

d) **Indexed allocation:**

14. İyi rastgele ve sıralı erişim, overhead var

15. 3 pt **File System Recovery** - Aşağıdakilerden hangisi file system crash recovery yöntemi DEĞİLDİR?

A. Journaling

B. Soft updates

**C. Disk partition**

D. Log-structured file system

E. Copy-on-write

## Bölüm 4: Protection and Security

16. **3 pt** **Cryptography Basics** - Aşağıdaki eşleştirmelerden hangisi YANLIŞTIR?
- A. AES → Symmetric encryption
  - B. RSA → Asymmetric encryption
  - C. SHA → Hash function
  - D. SSL → Access control model**
  - E. Salted hash → Password storage
17. **3 pt** **Access Control Models** - Aşağıdaki ifadelerden hangileri doğrudur?
- ✓ **DAC’de kullanıcı kendi dosyalarının erişimini kontrol eder.**
  - ✓ **MAC’de sistem (admin) tüm erişimleri kontrol eder.**
  - ✓ **RBAC’de izinler rollere atanır, kullanıcılara değil.**
  - ✓ **Access control list (ACL) DAC implementasyonudur.**
18. **Security Vulnerabilities and Defenses**
- Aşağıdaki güvenlik açıklarını doğru tanımları ve savunma mekanizmalarıyla eşleştiriniz:
- (a) **3 pt** Aşağıdakilerden hangisi **TOCTTOU (Time-of-Check to Time-of-Use, kontrol etme zamanından kullanma zamanına)** açığı ve çözümü için örnektir?
- A. Dosya işlemleri arasında sembolik link değiştirilmesi; çözüm: atomic işlemler, file descriptor kullanımı**
  - B. Buffer sınırlarının aşılması; çözüm: stack canary, ASLR
- (b) **3 pt** **Security Mechanisms** - Aşağıdaki güvenlik mekanizmalarından hangisi buffer overflow koruması için DEĞİLDİR?
- A. Stack canary and Bounds checking
  - B. Address Space Layout Randomization (ASLR)
  - C. Non-executable stack
  - D. Disk encryption (disk şifreleme)**
19. **3 pt** **OS Protection** - Aşağıdaki ifadelerden hangileri doğrudur?
- A. User mode ve kernel mode arasında mode bit ile ayırım vardır.**
  - B. System call’lar user mode’dan kernel mode’a geçiş sağlar.**
  - C. Privileged instructions sadece kernel mode’da çalıştırılabilir.**
  - D. Timer interrupts process’lerin CPU’yu monopolize etmesini engeller.**
20. **University Access Control System** Aşağıda üniversitede bir bölümdeki kullanıcılar ve dosyalar verilmiştir (Consider a university department with the following users and files):
- **Users:** Professor (Prof), Teaching Assistant (TA), Student (Stu)
  - **Files:**
    - `syllabus.txt` - owned by Prof (Profa ait)
    - `exam.pdf` - owned by Prof (Profa ait)
    - `hw.pdf` -owned by TA (TAye ait)
    - `submission.zip` - owned by Stu (Öğrenciye yani Student(Stu)ya ait)

Access requirements (Erişim gereksinimleri):

- `syllabus.txt`: Prof (rwx), TA (r), Stu (r)
- `exam.pdf`: Prof (rwx), TA (r)



- hw.pdf: TA (rwx), Prof (r), Stu (r)
- submission.zip: Stu (rwx), TA (rw), Prof (r)

Permissions(izinler): r = read, w = write, x = execute, - = no access

- (a) 3 pt Complete the access control matrix (erişim kontrol matrisini doldurunuz):

	syllabus.txt	exam.pdf	hw.pdf	submission.zip
Prof	<u>    rwx    </u>	<u>    rwx    </u>	<u>    r    </u>	<u>    r    </u>
TA	<u>    r    </u>	<u>    r    </u>	<u>    rwx    </u>	<u>    rw    </u>
Stu	<u>    r    </u>	<u>    -    </u>	<u>    r    </u>	<u>    rwx    </u>

- (b) 3 pt For each file, write its Access Control List (ACL) (her bir dosya için erişim kontrol listesini yazınız):

- syllabus.txt:     Prof: rwx, TA: r, Stu: r
- exam.pdf:     Prof: rwx, TA: r, Stu: -
- hw.pdf:     TA: rwx, Prof: r, Stu: r
- submission.zip:     Stu: rwx, TA: rw, Prof: r

- (c) 3 pt Design a Role-Based Access Control (RBAC) system for this scenario. Fill in the table (aşağıdaki tabloda verilen rol-tabanlı tabloyu, rol tabanlı erişim kontrol sistemi senaryosu için, tamamlayınız):

Role	Permissions (File: permissions, dosya: izinler)
Instructor (Öğretmen)	<u>syllabus.txt: rwx, exam.pdf: rwx, hw.pdf: r, submission.zip: r</u>
Assistant (Asistan)	<u>syllabus.txt: r, exam.pdf: r, hw.pdf: rwx, submission.zip: rw</u>
Learner (Öğrenci)	<u>syllabus.txt: r, exam.pdf: -, hw.pdf: r, submission.zip: rwx</u>

Assign users to roles (Kullanıcıları rollere atayınız):

- Prof →     Instructor
- TA →     Assistant
- Stu →     Learner