

**Ödevi Sisteme Yükleme için Son Tarih = 12/05/2025 Pazartesi 23:00**

**Geç gönderimler ve ilgili kurallara uymayan gönderimler KESİNLİKLE kabul edilmeyecektir! Tüm kuralları dikkatli bir şekilde okuyunuz.**

**Diğer Kurallar:**

- Her öğrenci ödevi kendisi yapmalıdır. Birlikte yapılan ödevler veya internetten alınan hazır kodlar (dosya okuma ve java dilindeki hazır veri yapıları hariç) **kopya** olarak değerlendirilecektir. Kontrol sonucu **kopya** tespit edilen ödevlere ödev notu olarak 0 verilecektir. Dersin öğretim elemanları kopya ödev gönderen öğrenciler hakkında **disiplin sürecini** başlatma hakkını saklı tutar.
- Dersin öğretim elemanları ödevi gönderen öğrencileri çağırıp (veya zoom üzerinden) çözümleri hakkında soru sorma ve anlattırma hakkını saklı tutar.
- Ödevler, UBYS (**Eders Değil!**) BSM304 İşletim Sistemleri sayfasındaki Ödevler sekmesinden sisteme yüklenmelidir. Mail yolu ile gönderilen ödevler kabul edilmeyecektir ve değerlendirmeye alınmaz.
- Uygulamanızın çalışıp çalışmadığı ve uygulamanızdaki eksik kısımlar vb. hakkında bilgi veren 1-2 satırlık kısa bir açıklama içeren .txt uzantılı metin dosyası (**açıklama.txt**) oluşturup, bunu proje klasörü ile birlikte sisteme yüklemeniz gerekmektedir.
- Proje klasörünü (açıklama içeren “açıklama.txt” dosyasıyla birlikte) sıkıştırıp (\*.zip) uzantılı tek bir dosya sisteme yükleyiniz. ( ( \*.rar) uzantılı dosyaların sisteme yüklenmesinde sıkıntı yaşanmaktadır ve ödeviniz değerlendirmeye alınmaz!)
- Proje klasörü yerine, sisteme tek bir class dosyası, tek bir java dosyası vb. farklı şekillerde yükleme yapanların ödevleri değerlendirilmeyecek ve 0 ile notlandırılacaktır.
- Projenizi “\_ÖğrenciNo\_Ad\_Soyad” (örn. \_2013510001\_Ali\_Bilir) şeklinde isimlendiriniz. Ayrıca, projede tanımladığınız her bir sınıfın ismi de öğrenci numaranız ile başlamalıdır (örn. \_2013510001\_Proses gibi).
- Kodunuza KESİNLİKLE **yorum satırları eklemeyiniz**.
- Son teslim tarihi ve saatine kadar ödevini sisteme yüklemeyenlerin ödevleri değerlendirmeye alınmayacaktır ve 0 ile notlandırılacaktır.

**Başarılar.**

#### **Ödev Açıklaması:**

Bu ödevde, Eclipse (Netbeans DEĞİL!) ve Java dilini kullanarak, prosesler arası iletişim yöntemlerinden biri olan adlandırılmamış pipe yöntemini simüle eden ve aşağıda belirtilen işlemleri yapacak **dinamik bir konsol uygulaması** geliştirmeniz beklenmektedir. Diğer hususlar şu şekildedir:

- Uygulama çalıştığında ilk olarak “girdi.txt” adındaki dosyayı okumalıdır. “girdi.txt” dosyasında iki kısım bulunmaktadır. İlk kısım olan prosesler kısmında, proseslere ilişkin bilgiler (producer proseslerin sayısı ve isimleri, consumer proseslerin sayısı ve isimleri) yer almaktadır. İkinci kısım olan olaylar kısmında ise write ve read çağrıları, bu çağrıların hangi proses tarafından kaçınıcı saniyede yapıldığı yer almaktadır. Olaylar kısmındaki çağrılar zamana (saniyeye) göre sıralı şekilde verilecektir. Belirli bir saniyede sadece bir çağrı olabilir. Ödevle birlikte örnek bir dosya sisteme yüklenecektir. Kendinizin de formata uygun farklı dosyalar oluşturup uygulamanızı test etmenizde yarar vardır.

- Örnek bir dosyanın ekran görüntüsü aşağıda yer almaktadır. Dosyadaki her bir değer arasında bir boşluk bulunmaktadır. Dosyadaki prosesler kısmının satır formatı şu şekildedir:

proses\_türü proses\_sayısı prosesin\_ismi prosesin\_ismi

**Örnek:** producer 2 p1 p18 (Bu örnekte toplamda 2 tane producer proses bulunmaktadır. Producer proseslerin isimleri p1 ve p18 dir. Benzer şekilde, başka prosesler varsa, onlar da yan yana yazılır.)

Dosyadaki olaylar kısmı 2 farklı satır formatına sahiptir. Producer proses için satır formatı şu şekildedir:

prosesin\_ismi çağrının\_yapıldığı\_saniye mesaj

**Örnek:** p1 3 "merhaba consumer" (Bu örnekte, p1 isimli producer proses "merhaba consumer" mesajı için 3. Saniyede write çağrısı yapmaktadır.)

Consumer proses için satır formatı şu şekildedir:

prosesin\_ismi çağrının\_yapıldığı\_saniye

**Örnek:** c4 5 (Bu örnekte, c4 isimli consumer proses 5. Saniyede read çağrısı yapmaktadır.)

3. İşletim sisteminin kernel'i, producer proses ve consumer proses olmak üzere 3 ayrı tip proses olmalıdır. Kernel proses bir tanedir, yani 2 veya daha fazla kernel proses olamaz. Öte yandan, producer ve consumer prosesler birden fazla olabilir. Producer proseslerin ismi p harfiyle consumer proseslerin ismi ise c harfiyle başlamaktadır; harften sonra sayı gelmektedir.
4. Pipe için uygun bir veri yapısı kullanılmalıdır ve bu veri yapısı kernel'in içinde yaratılmalıdır. Verilerin (mesajların) veri yapısına eklenmesi ve silinmesi işlemleri kernel proses tarafından gerçekleştirilmelidir. Mesajlar veri yapısının yazma ucundan yazılır ve okuma ucundan okunur. Veri yapısının kapasitesi 3 tür, yani en fazla 3 tane mesaj tutabilir. Öte yandan, mesajlar sadece string türündedir ve bir mesaj en fazla 20 karakter uzunluğunda olabilir. Burada bahsedilen pipe yapısı için, java dilinde bulunan hazır veri yapılarından uygun olan bir tanesini kullanabilirsiniz, lakin ek kontroller-fonksiyonlar tanımlamanız gerekebilir. (**Not:** Burada sizden istenen pipe yapısı Unix-Linux işletim sistemlerindeki gerçek pipe yapısından farklıdır!)
5. Producer prosesler sadece write çağrısı consumer prosesler ise sadece read çağrısı yapabilmektedir. Bu çağrılar işletim sisteminin kernel prosesine yapılmaktadır ve kernel proses bu çağrıları işlemektedir. Eğer tünelde veri yoksa ve bir proses read çağrısı yaparsa, çağrıyı yapan proses bloklanır ve beklemeye geçer. Pipe'a veri yazıldığında blokaj sona erdirilir ve okuma işlemi gerçekleşir. Eğer tünel tamamen doluysa ve bir proses write çağrısı yaparsa, çağrıyı yapan proses bloklanır ve beklemeye geçer. Pipe'ta yer açıldığında blokaj sona erdirilir ve yazma işlemi gerçekleşir.
6. Uygulamanız her saniyedeki durumu ekrana çıktı olarak vermelidir. Çıktı mesajları doğru saniyede ekrana yazdırılmalıdır. Örneğin, "3. Saniye: p1 prosesi write çağrısı yaptı." mesajından 1 saniye sonra "4. Saniye: c4 prosesi read çağrısı yaptı. Okunan mesaj: "merhaba consumer"" mesajı ekrana yazdırılmalıdır. Süreleri ayarlamak için timer veya thread.sleep() benzeri yapıları kullanabilirsiniz. (Örnek bir ekran görüntüsü aşağıda verilmiştir. Sizin uygulamanızın da aynı formatta ekrana çıktı üretmesi gerekmektedir. Formatı değiştirmeyiniz.)
7. Projenizi yaparken nesneye yönelik programlama tekniğini göz önünde bulundurunuz. Dolayısıyla, projenizde belirli sayıda ve uygun şekilde sınıflar ve metotlar tanımlanmalıdır.

Örnek bir ekran görüntüsü aşağıda verilmiştir:

girdi.txt dosyası okundu.

0. Saniye:

1. Saniye:

2. Saniye:

3. Saniye: p1 prosesi write çağrısı yaptı.

4. Saniye: c4 prosesi read çağrısı yaptı. Okunan mesaj: "merhaba consumer"

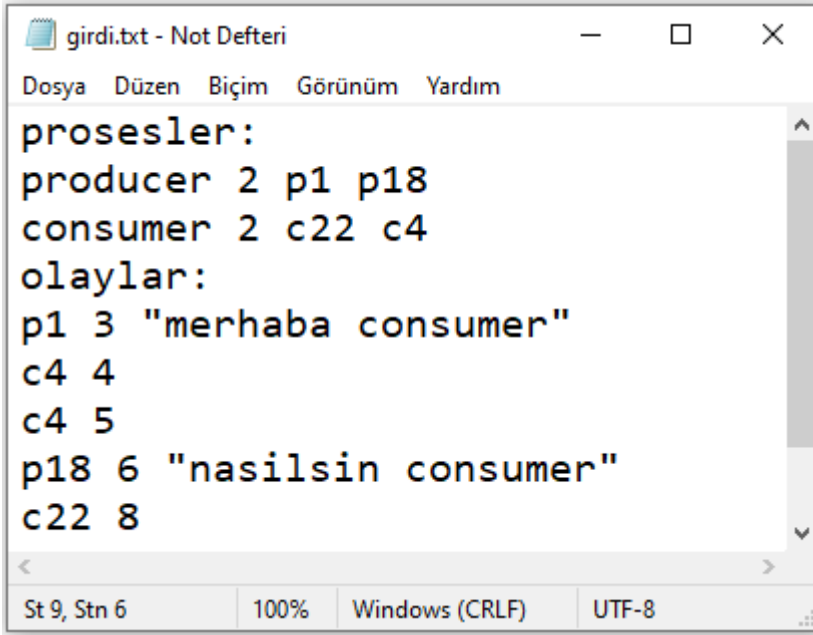
5. Saniye: c4 prosesi read çağrısı yaptı. c4 prosesi bloklandı ve kuyruğa alındı.

6. Saniye: p18 prosesi write çağrısı yaptı. c4 prosesi uyandırıldı. Okunan mesaj: "nasilsin consumer"

7. Saniye:

8. Saniye: c22 prosesi read çağrısı yaptı. c22 prosesi bloklandı ve kuyruğa alındı.

Örnek bir girdi.txt dosyasına ilişkin ekran görüntüsü aşağıda verilmiştir:



```
girdi.txt - Not Defteri
Dosya  Düzen  Biçim  Görünüm  Yardım
prosesler:
producer 2 p1 p18
consumer 2 c22 c4
olaylar:
p1 3 "merhaba consumer"
c4 4
c4 5
p18 6 "nasilsin consumer"
c22 8
St 9, Stn 6    100%    Windows (CRLF)    UTF-8
```