**Bil212 Veri Yapıları**

**Lab3**

**İşletim Sistemi Zamanlayıcısı**

**Kuyruk(Queue)**

**Zamanlayıcı(Scheduler):**

İşletim sistemi zamanlayıcısı (scheduler), bir bilgisayarın işlemci kaynaklarını etkin bir şekilde yönetmek ve çeşitli işlemlerin adil ve verimli bir şekilde çalışmasını sağlamak amacıyla kullanılan kritik bir bileşendir. Zamanlayıcı, çeşitli zamanlama algoritmaları kullanarak işlemlerin önceliklerine, ihtiyaçlarına ve sistemin genel performans hedeflerine göre kararlar verir. Sunumlardaki örnekte Round-Robin (RR) zamanlayıcı kuyruk veri yapısıyla nasıl implement edildiği gösterilmiştir.

RR nasıl çalışır?

Belli bir süre zarfında cpu daki iş bitmese dahi tekrar kuyruğa alınır ve kuyruktaki bekleyen iş cpu ya yüklenir. Bu süreye timekuantum (tk) denir. Örneğin tk = 4 birim olsun. job1 toplam süre 6 birim, job2 toplam süre 3 birim olsun. job1 cpu ya yüklendikten 4 birim zaman sonra yerini job2 ye bırakır, kendisi kuyruğa tekrar girer. job2, 3 birim olan çalışma süresi tk den küçük olduğu için cpu job2 yi bitirir ve yerine job1 yüklenir. Böylece ilk önce job2 sonra job1 tamamlanmış olur.

**Lab Çalışması**

Bu labda değişen cpu sayısına ve hızına göre gelen iş parçalarının çalışmasını programlayan bir zamanlayıcı sınıfı hazırlanacaktır. Bu zamanlayıcı içerisinde kuyruk veri yapısı kullanılacaktır.

**Simülasyon Ortamı:**

Simulasyon main metodda gerçekleşecektir. Main metodda bir for döngüsü bulunacak ve döngünün her bir devri bir cpu birim zamanı olarak değerlendirilecektir. Her cpu birim zamanında yeni bir iş parçası zamanlayıcı tarafından okunacak ve uygun kuyruğa eklenecektir. Her cpu için bir kuyruk tutulmalıdır. Cpu lar Round-Robin yöntemine göre kendi kuyruğundaki iş parçalarını çalıştıracaktır.

**Sınıflar ve Metodlar:**

***scheduler***

her bir cpu için bir iş kuyruğu tutulmalıdır. Bunu haricinde istediğiniz değişkenleri tutabilirsiniz. İmplement edilmesi gereken metodlar:

**init( )** 3 parametre alır.

1. Toplam cpu sayısı
2. cpu ların hızı yani bir zaman biriminde kaç zamanlık iş bitirebildiği
3. tk (timekuantum) sayısı

**addJob( )** job nesnesini scheduler daki uygun cpu kuyruğuna yükler.

**run ( )** simulasyonu asıl gerçekleştirildiği metoddur. Her bir çalışmasında bir birim zamanlık iş yapılır. Kuyruktan cpu ya iş aktarımı, tk yı geçen işlerin tekrar kuyruğa alınıp yerine yeni iş parçasının yüklenmesi burada gerçekleşir.

timeline() metoduna yardmcı olacak verileri burda loglayabilirsiniz.

**allJobsComplete( )** input dosyasındaki satırlar bittiğinde iş parçası ekleme döngüsünden çıkıp bütün işlerin run ile bitirilmesi gerekir. Bu durumda ikinci bir döngüde bu metod kullanarak bütün işler bitinceye kadar run() çalıştırılır.

**timeline( )** örnek çıktıdaki gibi o ana kadarki her bir log ekrana bastırılır.

**toString( )** örnek çıktıdaki gibi cpu lar alt alta, cpu kuyrukları da onların yanında olacak şekilde ekrana bastırılmalıdır. cpu nun yanındaki ilk iş, o anda çalışan işi, yanındakiler de kuyruğu gösterir.

**run metodu açıklama**

bu kısım da puanlamaya dahil olacaktır.

***job***

her bir iş için isim ve toplam çalışma süresi değişkenini tutmalısınız. Bunlar haricinde istediğiniz değişkenleri tanımlayıp bunlara erişme ve değiştirme metodlarını tanımlayabilirsiniz.

**Önemli:**

1. **sürücü ile çalışmayan gönderiler değerlendirilmeyecektir !!!**
2. **iş parçasının çalışması ne demek olduğunu anlamayanları yarınki laba gelmeleri gerekiyor.**
3. **yazılan private metodların nerede nasıl kullanıldığı açıklanmalıdır.**
4. **diğer metodlar run( ) metodunu etkilediği için onlar düzgün çalışmadığı takdirde diğer kısımlardan da alınacak puan yüzdesi azalacaktır. Örneğin init() metodu ile istenilen sayıda cpu ayarlanamıyorsa run() da alacağınız puan %10 a inecektir. Aynı durum toString için de geçerlidir.**
5. **run( ) metodu için yapılacak açıklama lab ın %20 sini oluşturur. Hangi değişkenleri ne için kullandığınızı, run içinde çalışan private metodlarınızı ve benzeri bilgileri açıklamalısınız.**

**Varsayımlar:**

* input dosyasındaki her bir satırda iş parçasını temsil eden isim ve iş parçasının süresi çifti bulunacaktır.
* İnput dosyasındaki iş parçacığı sırası varış zamanını ifade eder.
* iş parçacıkları cpu kuyruklarına eklenirken öncelikli olarak kuyruk boyutu az olana atama yapılır. Bunlarında aynı olduğu durumda indisi küçük olan cpu ya atama yapılacaktır.
* Bir iş tam bitmediği için cpu dan alınıp kuyruğa tekrar ekleneceği sırada scheduler kuyruğa yeni iş ekleyecek ise yeni iş sıraya sonradan eklenir.
* tk dolduğu için kuyruğa alınan iş sadece kendi cpu sunun kuyruğuna geri eklenebilir.
* cpu hızı 1 den büyük olduğunda bir run() çağrısı içinde en fazla bir iş parçası çalışabilir. Örneğin; kalan zaman 1 olan bir iş parçası hızı 3 olan cpu ile çalıştırılacağı zaman işi bitirdikten sonra yeni iş almak için diğer run() çağrısı beklenir. cpu 1 birimlik işi bitirince kalan 2 birim zamanda boşta bekler.

**Yüklenecekler:**

*Scheduler.java* adlı dosya ile implementasyonunuzu UZAK a yükleyebilirsiniz.

**Örnek Çalışma1:**

**input: cpu sayısı:1 cpu hızı:1 tk:4**

P1 3

P2 2

P3 6

P4 1

**çıktı:**

**Scheduler dump at time 1:**

**CPU 0: P1 (Remaining Time: 3)**

**Scheduler dump at time 2:**

**CPU 0: P1 (Remaining Time: 2) P2 (Remaining Time: 2)**

**Scheduler dump at time 3:**

**CPU 0: P1 (Remaining Time: 1) P2 (Remaining Time: 2) P3 (Remaining Time: 6)**

**Scheduler dump at time 4:**

**CPU 0: P2 (Remaining Time: 2) P3 (Remaining Time: 6) P4 (Remaining Time: 1)**

**Scheduler dump at time 5:**

**CPU 0: P2 (Remaining Time: 1) P3 (Remaining Time: 6) P4 (Remaining Time: 1)**

**Scheduler dump at time 6:**

**CPU 0: P3 (Remaining Time: 6) P4 (Remaining Time: 1)**

**Scheduler dump at time 7:**

**CPU 0: P3 (Remaining Time: 5) P4 (Remaining Time: 1)**

**Scheduler dump at time 8:**

**CPU 0: P3 (Remaining Time: 4) P4 (Remaining Time: 1)**

**Scheduler dump at time 9:**

**CPU 0: P3 (Remaining Time: 3) P4 (Remaining Time: 1)**

**Scheduler dump at time 10:**

**CPU 0: P4 (Remaining Time: 1) P3 (Remaining Time: 2)**

**Scheduler dump at time 11:**

**CPU 0: P3 (Remaining Time: 2)**

**Scheduler dump at time 12:**

**CPU 0: P3 (Remaining Time: 1)**

**Timeline:**

**Job P1 arrived at time 1**

**Job P1 started at time 1**

**Job P2 arrived at time 2**

**Job P3 arrived at time 3**

**Job P1 finished at time 4**

**Job P2 started at time 4**

**Job P4 arrived at time 4**

**Job P2 finished at time 6**

**Job P3 started at time 6**

**Job P4 started at time 10**

**Job P4 finished at time 11**

**Job P3 finished at time 13**

**Process finished with exit code 0**

**Örnek Çalışma2:**

**input: cpu sayısı:1 cpu hızı:1 tk:1**

P1 3

P2 2

P3 6

P4 1

**çıktı:**

**Scheduler dump at time 1:**

**CPU 0: P1 (Remaining Time: 3)**

**Scheduler dump at time 2:**

**CPU 0: P1 (Remaining Time: 2) P2 (Remaining Time: 2)**

**Scheduler dump at time 3:**

**CPU 0: P2 (Remaining Time: 2) P1 (Remaining Time: 1) P3 (Remaining Time: 6)**

**Scheduler dump at time 4:**

**CPU 0: P1 (Remaining Time: 1) P3 (Remaining Time: 6) P2 (Remaining Time: 1) P4 (Remaining Time: 1)**

**Scheduler dump at time 5:**

**CPU 0: P3 (Remaining Time: 6) P2 (Remaining Time: 1) P4 (Remaining Time: 1)**

**Scheduler dump at time 6:**

**CPU 0: P2 (Remaining Time: 1) P4 (Remaining Time: 1) P3 (Remaining Time: 5)**

**Scheduler dump at time 7:**

**CPU 0: P4 (Remaining Time: 1) P3 (Remaining Time: 5)**

**Scheduler dump at time 8:**

**CPU 0: P3 (Remaining Time: 5)**

**Scheduler dump at time 9:**

**CPU 0: P3 (Remaining Time: 4)**

**Scheduler dump at time 10:**

**CPU 0: P3 (Remaining Time: 3)**

**Scheduler dump at time 11:**

**CPU 0: P3 (Remaining Time: 2)**

**Scheduler dump at time 12:**

**CPU 0: P3 (Remaining Time: 1)**

**Timeline:**

**Job P1 arrived at time 1**

**Job P1 started at time 1**

**Job P2 arrived at time 2**

**Job P2 started at time 3**

**Job P3 arrived at time 3**

**Job P4 arrived at time 4**

**Job P1 finished at time 5**

**Job P3 started at time 5**

**Job P2 finished at time 7**

**Job P4 started at time 7**

**Job P4 finished at time 8**

**Job P3 finished at time 13**

**Process finished with exit code 0**

**Örnek Çalışma3:**

**input: cpu sayısı:2 cpu hızı:1 tk:4**

P1 3

P2 2

P3 6

P4 1

**çıktı:**

**Scheduler dump at time 1:**

**CPU 0: P1 (Remaining Time: 3)**

**CPU 1:**

**Scheduler dump at time 2:**

**CPU 0: P1 (Remaining Time: 2)**

**CPU 1: P2 (Remaining Time: 2)**

**Scheduler dump at time 3:**

**CPU 0: P1 (Remaining Time: 1) P3 (Remaining Time: 6)**

**CPU 1: P2 (Remaining Time: 1)**

**Scheduler dump at time 4:**

**CPU 0: P3 (Remaining Time: 6)**

**CPU 1: P4 (Remaining Time: 1)**

**Scheduler dump at time 5:**

**CPU 0: P3 (Remaining Time: 5)**

**CPU 1:**

**Scheduler dump at time 6:**

**CPU 0: P3 (Remaining Time: 4)**

**CPU 1:**

**Scheduler dump at time 7:**

**CPU 0: P3 (Remaining Time: 3)**

**CPU 1:**

**Scheduler dump at time 8:**

**CPU 0: P3 (Remaining Time: 2)**

**CPU 1:**

**Scheduler dump at time 9:**

**CPU 0: P3 (Remaining Time: 1)**

**CPU 1:**

**Timeline:**

**Job P1 arrived at time 1**

**Job P1 started at time 1**

**Job P2 arrived at time 2**

**Job P2 started at time 2**

**Job P3 arrived at time 3**

**Job P1 finished at time 4**

**Job P2 finished at time 4**

**Job P3 started at time 4**

**Job P4 arrived at time 4**

**Job P4 started at time 4**

**Job P4 finished at time 5**

**Job P3 finished at time 10**

**Process finished with exit code 0**

**Örnek Çalışma4:**

**input: cpu sayısı:1 cpu hızı:2 tk:2**

P1 3

P2 2

P3 6

P4 1

**Scheduler dump at time 1:**

**CPU 0: P1 (Remaining Time: 3)**

**Scheduler dump at time 2:**

**CPU 0: P1 (Remaining Time: 1) P2 (Remaining Time: 2)**

**Scheduler dump at time 3:**

**CPU 0: P2 (Remaining Time: 2) P3 (Remaining Time: 6)**

**Scheduler dump at time 4:**

**CPU 0: P3 (Remaining Time: 6) P4 (Remaining Time: 1)**

**Scheduler dump at time 5:**

**CPU 0: P3 (Remaining Time: 4) P4 (Remaining Time: 1)**

**Scheduler dump at time 6:**

**CPU 0: P4 (Remaining Time: 1) P3 (Remaining Time: 2)**

**Scheduler dump at time 7:**

**CPU 0: P3 (Remaining Time: 2)**

**Timeline:**

**Job P1 arrived at time 1**

**Job P1 started at time 1**

**Job P2 arrived at time 2**

**Job P1 finished at time 3**

**Job P2 started at time 3**

**Job P3 arrived at time 3**

**Job P2 finished at time 4**

**Job P3 started at time 4**

**Job P4 arrived at time 4**

**Job P4 started at time 6**

**Job P4 finished at time 7**

**Job P3 finished at time 8**

**Process finished with exit code 0**