Doğukan Arkan
15253507

Main() method'umda Thread class'ından extend edilen SchedulingAlgorithms class'ından thread dizisi oluşturdum ve bu dizi elemanlarını start() komutu ile başlattım. Daha sonra main() thread'in diğer thread'leri beklemesini join() komutu yardımıyla yaptım.

Process class'ımda oluşturulan process'lerin özellikleri bulunuyor. Bu özelliklerin atama işlemleri için de constructor oluşturdum.

Tüm işlemleri ise SchedulingAlgorithms class'ında gerçekleştirdim. Bu class'ı Thread class'ından extend ve Comparable class'ında implement ettim. Thread yapısı gereği run() method'unu @Override olacak şekilde tanımladım ve thread.start() işlemi yapıldıktan sonra çalışmasını sağladım. Process bilgilerini txt dosyasından okuyarak process'leri oluşturdum ve ArrayList tipindeki processList içerisine ekledim. Daha sonra her farklı algoritma için açtığım 7 method'u switch-case yapısıyla çağırdım. Her method'da da sonuçları farklı txt dosyalarına yazdırıp main() method'unda bu dosyaları okuttum. Bir process'in bittiğini ise Finished özelliğinde tuttum. Yine her algoritma türü için nanosaniye cinsinde zamanlarını hesapladım.

First Come First Served algoritmasında process'leri ArrivalTime değerlerine göre sıralayarak problemi çözdürdüm.

Nonpreemptive Shortest Job First algoritmasında process'leri bu sefer BurstTime değerlerine göre sıralayarak CPU'ya atama işlemini gerçekleştirdim. Böylece çalışma süresi kısadan uzuna doğru process'ler çalıştı.

Preemptive Shortest Job First algoritmasında ise yaptığım tek fark her CPU Burst Time değerinde process'leri tekrardan BurstTime değerlerine göre sıralayarak çalıştırmak oldu.

Preemptive Priority algoritmasında Priority değerleri işin içine giriyor. Bu yüzden process'leri Priority değerlerine göre sıraladım ve hesaplattım.

Round Robin algoritmalarında ise process'leri FCFS algoritmasındaki gibi ArrivalTime değerlerine göre sıraladıktan sonra her Time Quantum biriminde process'leri kontrol ederek işlemleri gerçekleştirdim.

Grafiklere yapabileceğim yorum ise; process sayısının 10'a katlanmasına rağmen nonpreemptive olarak çalışan FCFS ve SJF algoritmalarında sadece 45 nanosaniye artış gözlemlendi. Fakat preemptive olarak çalışan SJF algoritması çok büyük bir değişim göstererek 94 nanosaniyeden 310 nanosaniyelere çıktı. Tüm grafiklere bakarsak 8 time quantum değerine sahip RR algoritmasının en hızlı şekilde çalıştığı yorumunu yapabiliriz. Genel olarak bakıldığında nonpreemtive olarak çalışan algoritmaların daha düşük sürede çalıştığı yorumunu yapabilirim. Tabii ki bu her zaman böyle olacak diye bir şey yoktur. Bu örnekte process'in değerlerini rastgele oluşturdum.