T.C.

SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

İŞLETİM SİSTEMLERİ ÖDEV RAPORU

GÖREVLENDİRİCİ KABUĞU

HAZIRLAYANLAR

GRUP NO:32

G191210015-Mustafa ŞİMŞEK

2.A

G201210066-Görkem KARAGÖL

2.A

G191210450-Gökberk Furkan ÇAKIR

2.A

G201210052-Gökhan Özcan

1.B

https://github.com/simsek1462/IsletimOdev

BİZDEN İSTENİLENLER:

Görevlendirici, proses listesinden (giriş.txt) beslenen iki adet kuyruğa sahiptir: Gerçek Zamanlı ve Kullanıcı Proses kuyrukları. Proses listesi, her zaman adımında sürekli işlenir ve gelen prosesler uygun kuyruğa aktarılır. Kuyruklar işleme alınır; tüm gerçek zamanlı prosesler tamamlanmak üzere çalıştırılır ve o anda çalışmakta olan diğer düşük öncelikli prosesler kesilir. Düşük öncelikli geri beslemeli görevlendirici yeniden etkinleştirilmeden önce gerçek zamanlı kuyruktaki prosesler bitirilmelidir. Bir geri besleme kuyruğunun normal çalışması, en yüksek öncelik düzeyindeki prosesleri alır, uygun kuantuma göre işler ve daha sonra önceliğini düşürerek bir alt kuyruğa yerleştirir. Ancak her iş öncelik değerine uygun bir kuyruğa yerleştirilir . Eğer tüm prosesler en alt seviye kuyrukta ise o zaman basit bir çevrimsel sıralı (round robin) algoritma çalıştırılır. Tüm "hazır" yüksek öncelikli prosesler tamamlandığında, geri beslemeli görevlendirici, en yüksek öncelikli ve boş olmayan kuyruğun başındaki prosesin kaldığı yerden devam etmesiyle çalışmasını sürdürür. Bir sonraki zaman adımında, eşit veya daha yüksek önceliğe sahip başka "hazır" prosesler varsa, mevcut proses askıya alınır (veya sonlandırılır).

GELİŞTİRİLEN YAZILIM:

Geliştirmiş olduğumuz yazılım “Process” , “Görevlendirici” , ”TimerControl” ve “Uygulama” olmak üzere 4 adet sınıfa sahiptir. **Process** sınıfında “Giris.txt” adlı dosyada verilen proseslerin özellikleri tutuluyor. Ayrıca prosesin durumunu gösteren ve ekrana yazdıran fonksiyonlarımız bulunuyor. **Görevlendirici** sınıfımız ise zamana bağlı olarak gelen prosesleri gerçek zamanlı ya da kullanıcı zamanlı kuyruklarımıza atıyor. Daha sonra gerçek zamanlı kuyruk boş olmadıkça buradaki prosesleri alıp bitene kadar çalıştırır. Eğer gerçek zamanlı kuyruk boş ise kullanıcı zamanlı kuyrukların en önceliklisi kuantumu 1 olmak üzere çalıştırılır. Askıya alınan proseslerin önceliği düşürülerek uygun kuyruğa aktarılır. Kullanıcı zamanlı kuyrukların ilk iki öncelikli kuyruğu boş ise son kuyruktaki prosesler Round-Robin mantığı ile çalıştırılır. Eğer bir proses askıya alınırsa “askiyaAlinanlar” listesinde tutulur ve her saniye bekleme süreleri güncellenir eğer 20 saniye ulaşan prosesler var ise bunlar zaman aşımına uğratılır. Ancak bir proses askıya alınıp daha sonra yürütüldüyse “askiyaAlinanlar” listesinden çıkarılır. **TimerControl** sınıfımızda ise her saniyede Görevlendirici sınıfının fonksiyonları çağrılır. Bu fonksiyonlar ile her saniyede gelen prosesler uygun kuyruklara atılır , hangi kuyruğun önce işleneceğine karar verilir , askıya alma durumları ve zaman aşımı durumları kontrol edilir. **Uygulama** sıfımızda ise Giris.txt okunur. Her satırdaki değerler parçalanır , proses oluşuturlup değerler prosesin constructor fonksiyonuna verilir. Oluşa prosesler bir arraylist yapısında tutulur. Dosya okuma işleminden sonra görevlendirici nesnemiz oluşur. Oluşan görevlendirici nesnemiz proses listesini alır ve bu listeyi her saniye işler.

ÖDEVDE ZORLANDIĞIMIZ KISIMLAR:

-Konsolda renkli çıktı alma aşamasında zorlandık.

-Zaman aşımına uğratma işleminde zorlandık.

ÖDEVİN BİZE KATTIKLARI:

-CPU-Scheduling mantığını daha iyi anladık.

-Round-Robin algoritmasını daha iyi anladık.