İşletim sistemleri Dersi vize proje raporu

Hazırlayan:İbrahim Halil Güneş

Konya Teknik Üniversitesi

Yazılım Mühendisliği

**Soru 1-**

Verilen sorudaki amaç 1'den 1.000.000'e kadar olan sayıları dört farklı kriterle işleyerek sonuçları farklı ArrayList'lere eklemektir

Listeye ekleme kriterleri:

1-Çift Sayılar: 1'den 1.000.000'e kadar olan çift sayıları bulmak.

2-Tek Sayılar: 1'den 1.000.000'e kadar olan tek sayıları bulmak.

3-Asal Sayılar: 1'den 1.000.000'e kadar olan asal sayıları bulmak.

Kod Yapısı ve İşleyişi:

Main sınıfında 1'den 1.000.000'e kadar olan sayılar oluşturulur.Çeşitli kriterlere göre sayıları işleyecek dört farklı Thread oluşturulur.Her bir Thread belirli bir kriterle sayıları işleyerek sonuçları ortak ArrayList'lere ekler.Son olarak, elde edilen sonuçlar konsola yazdırılır.

NumberFinder Sınıfı:

Runnable arayüzünü uygulayan bu sınıf, belirli bir kriterle sayıları işleyen Thread'i temsil eder.isPrime metodu ile verilen sayının asal olup olmadığı kontrol edilir.Her bir Thread, sayıları işlerken sonuçları eklemek için senkronize bir şekilde ArrayList'e erişir.

Sonuçlar:

Çalıştırılan kod ile, 1'den 1.000.000'e kadar olan çift, tek ve asal sayılar belirlenir.Bu sayılar ayrı ayrı oluşturulan ArrayList'lerde saklanır.Her bir Thread'in kendi kriterine göre işlediği sayılar belirlenir.

**Soru 2-**

**1-Sanal Bellek Kavramları**

Sanal bellek Windows'un ram üzerinde o anda işletilmeyen verileri depoladığı bir sabit disk dosyasıdır.[[1]](#footnote-1)

Sanal bellekte, bir programın bilgiyi tanımlamak için kullanabileceği adresler, bellek sisteminin fiziksel depolama sitelerini tanımlamak için kullandığı adreslerden ayrılır ve program tarafından oluşturulan adresler otomatik olarak karşılık gelen makine adreslerine çevrilir.[[2]](#footnote-2)

Sanal belleğin avantajları şunlardır:Uygulamaların daha fazla bellek kapasitesiyle çalışmasını sağlar.Bellek kullanımını optimize eder.Daha az fiziksel bellek ile daha fazla işlem yapılmasına olanak tanır.Ancak, sanal bellek kullanımının aşırı olması performans düşüklüğüne neden olabilir, çünkü sabit disk erişim süresi, fiziksel belleğe göre daha yavaştır. Bu nedenle, sanal bellek kullanımının dengeli bir şekilde yapılması önemlidir.

**2-Sayfa Değiştirme Algoritmaları**

1.İlk giren ilk çıkar algoritması (FIFO): FIFO algoritması işletmelerde de yoğun olarak kendisine uygulama alanı bulmuş ve yaygın olarak kullanılan bir yapıdır. Tampon bölgesine bir blok yerleşimi gerçekleştirileceği zaman FIFO algoritması tampon üzerinde ilk olarak tahsis edilmiş alanları tespit eder. Yerleştirilecek olan blok sayısına göre ilk olarak yerleşen bloktan itibaren ihtiyaç duyulan miktar kadar blok yapısı yer boşaltarak yeni gelen blokların boşaltılan yerlere yerleşimini gerçekleştirir. Işletmelerde çok kullanışlı bir yapı olarak kullanılmasına ragmen bilgisayar sistemlerinde aynı performansı sağladığı söylenemez. FIFO algoritmasına göre gelen pin 60 ve pin 70 istekleri şekil 3 te göründüğü gibi ilk olarak pin edilmiş bölgeler olan 0 ve 2 bölgeleridir. Bu bölgeler gelen pin 60 isteği buffer 0 bölgesine ve pin 70 isteği buffer 2 bölgesine yerleşimleri gerçekleştirilir.

2. En son kullanılan sayfa algoritması (LRU): LRU algoritması tampon bölgesi üzerinde yerleştirme yapacağı zaman tampon bölgesinde arama işlemi gerçekleştirerek en eski boşaltılmış olan yerden itibaren başlayarak ihityaç duyulan miktar kadar ileri doğru gelerek yer değiştirme işlemini icra eder. Bu algoritmada blokların daha önceki yerleştirilme sırası değil, bloklar üzerindeki işlemler sonucu boşaltılma değeri kriter alınarak işlemler gerçekleştirilir. LRU algoritması senaryo üzerindeki gelen 11 ve 12. istekleri en eski unpin edilmiş alan olan buffer 3 (unpin 7) ve buffer 0 (unpin 8) olan bölgeler üzerine yerleştirmektedir. 7 ve 8. sırada gelmiş olan istekler en eski unpin işlemleridir. Bu durumda pin (60) isteği buffer 3, pin (70) isteği buffer 0 bölgesine yerleştirilecektir.

3. Saat yerleşim algoritması (Clock): Clock algoritması tampon bölgesi üzerinde saat önünde bir çevrim gerçekleştirerek işlemlerini gerçekleştirir. Saat çevrimi yönünde arama işlemine başlanarak en son yer değiştirme işlemi gerçekleştirmiş olan buffer tespit edilir ve bu tespit edilen bloktan sonraki ilk tahsis edilmemiş olan blok üzerine yerleşim gerçekleştirilir. Yerleşimi gerçekleştirilecek olan her blok için işlem tekrarlanır. Clock algoritması senaryo üzerinde gelen 11 ve 12 nolu istekler için ilk once en son yerdeğiştirme işlemi gerçekleştirmiş olan bölgeyi tespit eder bu bölge buffer 1 bölgesidir. Bu bölgenin tespitinden sonra unpin durumdaki bölge üzerine yerleştirme işlemini gerçekleştirir. Bu durumda gelen pin 60 isteği buffer 2 ve pin(70) isteği buffer 3 bölgesine yerleştirilecektir.

4. LRM algoritması (least recently made): LRM algoritması tampon bölgesinde sayfa yerdeğiştirme işlemini gerçekleştirirken tamponda bulunan blokların en eski yerleşim zamanından itibaren en az kullanılanları (okuma ve yazma işlemini gerçekleştirenleri) tespit ederek yeni gelen blokları tespit edilenler ile yer değiştirirek işlemleri gerçekleştirir.[[3]](#footnote-3)

**3-Sanal Bellek Hiyerarşisi:**

Hafıza hiyerarşisi çeşitli veri depolama birimlerinin veri iletim hızı/işlem gücüne göre hiyerarşik olarak sıralanmasıdır.Bellek hiyerarşisi çeşitli veri depolama birimlerinin veri iletim hızı/işlem gücüne göre hiyerarşik olarak sıralanmasına verilen addır. İşlemcilerin hızı ve işlem gücü arttıkça bilgisayar mimarisini oluşturan veri saklama birimlerinin işlemciyle arasındaki senkron farkı gitgide artmaktadır. Veri depolama birimleri arasındaki bu tür farklılıkların önüne geçmek, işlemcinin döngülerce diğer birimlerin veri işlemesi/iletmesi sırasında beklemesini engellemek için önbellek (cache) adı verilen geçiş birimleri oluşturulmuştur. Bu hiyerarşide bir bellek türü bir üst düzeydeki bellek türünü önbellek olarak kullanır. Örneğin, yerel diskte kayıtlı bir dosyada işlem yaparken ana bellek işletim sistemi tarafından geçici depolama birimi olarak kullanılabilir. Hiyerarşideki bellek türleri hızlılık, düşük gecikme ve maliyete göre yukarıdan aşağıya doğru şu şekilde sıralanabilir:

– Geçici saklayıcılar (CPU)

– L1 önbellek (SRAM)

– L2 önbellek (SRAM)

– Ana bellek (DRAM)

– Yerel ikincil depolama birimleri

– Uzaktan erişimli ikincil depolama birimler[[4]](#footnote-4)

4- Sayfa Tablosu Analizi:

Sanal bellekte sayfalar, belleği dizinleyen bir tablo kullanılarak yerleştirilirler. Bu yapı **“sayfa tablosu (page table)”** olarak adlandırılır. Bellekte tutulan sayfa tablosu sanal bellek adresinin numarasına göre dizinlenmiştir ve ona karşılık gelen gerçek sayfa numarasını içerir. Her program, sanal adres uzayını, ana bellekteki bellek uzayına dönüştüren kendine ait bir sayfa tablosuna sahiptir. Sayfa tablosu, ana bellekte mevcut olmayan sayfaların kayıtlarını da tutabilir. Her sayfa tablosunda **geçerli bit** (1 veya 0 ) tutulur. Eğer bu bit mantıksal sıfıra eşit ise sayfa ana bellekte mevcut değil demektir ve **“sayfa hatası (page fault)”** oluşur. Eğer bit mantıksal bire işaret ediyorsa sayfa ana bellekte mevcut ve geçerli bir fiziksel adrese sahip demektir.

Sayfa Tablosu Oluşturulurken Kullanılan Teknikler:

1.Sayfa tablosunun boyutlarını sınırlayan bir sınır kaydı tutmak. Eğer sanal sayfa numarası sınır kaydının sınırını aşarsa kayıtlar sayfa tablosuna eklenmelidir. Bu teknik, bir işlem daha fazla alana ihtiyaç duyduğunda sayfa tablosuna büyüme yeteneği kazandırır. Sonuç olarak sanal adres uzayı yalnızca ihtiyaç duyulduğunda büyük tutulacaktır.

2.Bölmelere ayırmak (segmentation): Çoğu dil iki boyutları büyüyebilen iki ayrı alan gerektirdiği için tek boyutta büyümenin yetersiz olduğu durumlarda sayfa tablosu ikiye bölünür. İki ayrı sayfanın farklı sınırlara sahip olması desteklenir. İki sayfa tablosu kullanımı adres uzayını da ikiye parçaya (segment) böler. Sınır kaydı her iki parça için de tutulur. MIPS mimarisi de bu tekniği desteklemektedir.

3.Sanal adrese bir hesaba dayalı adresleme (hashing) işlevi eklenerek sayfa tablosu veri yapısının sadece gerçek sayfa sayısı kadar boyutta olması sağlanabilir.Bu yapıya ters çevrilmiş (inverted) sayfa tablosu adı verilir.

4.Birden fazla seviyeli sayfa tabloları kullanılabilir.

5.Sayfa tabloları kullanımı için gerekli ana bellek miktarının azaltılması sayfa tablolarının tekrar sayfalanması ile sağlanabilir.[[5]](#footnote-5)

**5-Bellek Fragmentasyonu:**

Bellek Fragmentasyonu

Bellek fragmentasyonu, bellekteki boş alanların, kullanılan alanlar tarafından parçalara ayrılmasıyla oluşan bir durumu ifade eder. Bu parçalar, tek bir büyük boş alana dönüştürülemez, bu da belleğin etkin bir şekilde kullanılmasını engeller.

Bellek fragmentasyonu, özellikle dinamik bellek tahsis edildiğinde ve bu alanların zamanla serbest bırakılması durumunda ortaya çıkar. Bellek yönetimi stratejileri, bu tür fragmentasyonları minimize etmek için tasarlanmıştır.

İç Fragmentasyon:

İç fragmentasyon, bir bellek bloğunun (genellikle sabit bir boyuta sahip olan) tam olarak kullanılmadığı durumu ifade eder. Örneğin, bir bellek bloğu 512 byte ise ve bir süreç sadece 300 byte alan kullanıyorsa, bu blokta 212 byte boşta kalır. Bu boşluk, iç fragmentasyon olarak adlandırılır ve belleğin etkin bir şekilde kullanılmasını engeller.

Dış Fragmentasyon:

Dış fragmentasyon, bellekteki boş alanların birleştirilmesiyle elde edilemeyecek kadar küçük parçalara ayrılması durumudur. Örneğin, birçok küçük boş alan varsa ve bir süreç için yeterli büyüklükte bir alan oluşturmak istendiğinde, bu boş alanların toplamı gereken boyutta bir alan oluşturacak kadar büyük olmayabilir. Bu, dış fragmentasyon olarak adlandırılır ve yeni bellek tahsislerini zorlaştırır.

Bellek yönetimi algoritmaları ve stratejileri, bu tür fragmentasyonları minimize etmek veya önlemek için tasarlanmıştır. Ancak, bellek kullanımı ve serbest bırakma işlemleri sırasında dikkatli bir yönetim gerektirir.

1. “Virtual Memory-Virtual Ram - Sanal Bellek Nedir?,” accessed January 5, 2024, https://www.firatboyan.com/virtual-ram-sanal-bellek-nedir.aspx. [↑](#footnote-ref-1)
2. Peter J. Denning, “Virtual Memory,” *ACM Computing Surveys* 2, no. 3 (September 1970): 153–89, https://doi.org/10.1145/356571.356573. [↑](#footnote-ref-2)
3. Ünal Çavuşoğlu and Ahmet Zengi̇N, “BELLEK YÖNETİMİNDE SAYFA DEĞİŞİM ALGORİTMALARININ PERFORMANS ANALİZİ,” no. 1 (n.d.). [↑](#footnote-ref-3)
4. “COS 40 Memory.Pdf,” accessed January 5, 2024, https://ckk.com.tr/ders/bilgisayarorg/COS%2040%20Memory.pdf. [↑](#footnote-ref-4)
5. “Sayfa tablosu,” in *Vikipedi*, November 13, 2023, https://tr.wikipedia.org/w/index.php?title=Sayfa\_tablosu&oldid=30550959. [↑](#footnote-ref-5)