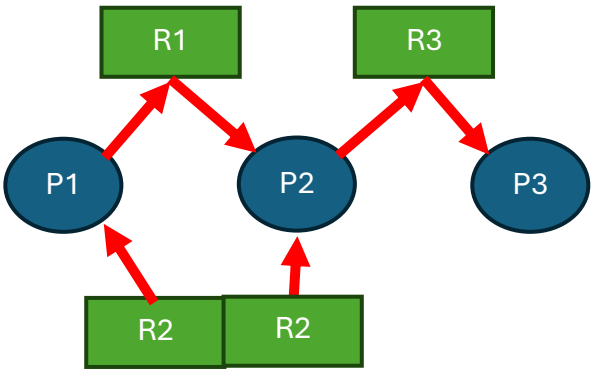
GİRESUN ÜNİVERSİTESİ MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
İŞLETİM SİSTEMLERİ DERSİ FİNAL SINAVI

Adı – Soyadı – Numarası:

Soru 1: Aşağıda durumu verilen kaynak tahsis çizgesini (resource allocation graph) çiziniz.
P1, bir R2 kaynağını tutar ve R1 kaynağını bekler.
P2, R1 kaynağını ve bir R2 kaynağını tutar ve R3 kaynağını bekler.
P3, R3 kaynağını tutar.

Kaynak tahsis çizgesi (resource allocation graph), sistemdeki süreçler ve kaynaklar arasındaki ilişkiyi görsel olarak gösteren bir çizge türüdür. Bu çizgede, süreçler ve kaynaklar düğümlerle temsil edilir ve süreçlerden kaynaklara veya kaynaklardan süreçlere doğru kenarlar çizilir. Süreçler P1, P2 ve P3 olarak temsil edilir. Kaynaklar R1, R2 ve R3 olarak temsil edilir.

P1 --> R1 (bekliyor)
P2 --> R3 (bekliyor)
R2 --> P1 (tutuyor)
R1 --> P2 (tutuyor)
R2 --> P2 (tutuyor)
R3 --> P3 (tutuyor)



Soru 2: Kilitlenme olması için gerekli şartlar nedir? Açıklayınız.

Kilitlenme (deadlock), bir grup sürecin, birbirlerinin kaynaklarını bekleyerek sonsuz bekleme durumuna girdiği durumdur. Kilitlenmenin gerçekleşmesi için dört gerekli ve yeterli şart bulunur.

- Karşılıklı Dışlama (Mutual Exclusion): En az bir kaynak, aynı anda yalnızca bir süreç tarafından kullanılabilir olmalıdır. Yani, kaynaklar paylaşılamaz ve sadece tek bir süreç tarafından tutulabilir.
- Kaynak Tutma ve Bekleme (Hold and Wait): Bir süreç, en az bir kaynağı tutarken aynı zamanda başka kaynakları da bekliyor olmalıdır. Bu durumda süreç, sahip olduğu kaynakları serbest bırakmadan yeni kaynaklar talep eder.
- Geri Alamama, Kesme Yok (No Preemption): Süreçlerden tuttukları kaynaklar zorla geri alınamaz. Kaynaklar, sadece işini tamamlayan süreç tarafından serbest bırakılabilir.
- Döngüsel Bekleme (Circular Wait): Süreçler arasında döngüsel bir bekleme durumu olmalıdır. Yani, P1 süreci bir kaynağı beklerken, bu kaynağı tutan P2 süreci başka bir kaynağı bekler ve bu döngü devam eder, en sonunda son süreç de P1'in tuttuğu kaynağı bekler.

Bu dört şartın hepsi aynı anda gerçekleşirse, sistemde kilitlenme meydana gelir. Bu şartların herhangi birinin ihlal edilmesi, kilitlenmenin oluşmasını engeller.

Soru 3: Bellek yönetiminde fiziksel adres yerine mantıksal adres kullanımının (bellek soyutlaması) getirdiği avantaj ya da dezavantajlar nedir?

Bellek yönetiminde mantıksal adreslerin (logical address) kullanımı, fiziksel adreslere (physical address) doğrudan erişim yerine bir soyutlama katmanı ekleyerek birçok avantaj ve bazı dezavantajlar sunar.

Başarılar.. Dr. Öğr. Üyesi Sercan KÜLCÜ 25.06.2024

Mantıksal adresleme, süreçlerin birbirlerinin bellek alanlarına erişimini engeller. İşletim sistemi, her sürecin yalnızca kendi mantıksal adres alanında çalışmasını sağlayarak bellek korumasını gerçekleştirir. Mantıksal adresleme, programcıların fiziksel bellek detaylarından soyutlanmasını sağlar. Bu soyutlama, uygulamaların taşınabilirliğini artırır ve farklı donanım yapılandırmalarında çalışabilir hale getirir. Mantıksal adresleme, süreçlerin belirli bellek alanlarını paylaşmalarını sağlar. Örneğin, kütüphaneler ve dinamik bağlama (dynamic linking) bu şekilde etkinleştirilir. Mantıksal adresleme, bellek parçalanmasını (fragmentation) azaltarak bellek kullanımını daha verimli hale getirir. Sayfalama (paging) ve kesimleme (segmentation) gibi teknikler, bellek tahsisini ve kullanımını optimize eder. Sanal bellek kullanımı, fiziksel bellek miktarının ötesinde bellek kullanımına izin verir. Bu, disk üzerindeki bellek alanının RAM gibi kullanılmasını sağlar.

Mantıksal adresleme ve sanal bellek yönetimi, ek işlemci döngüleri ve bellek erişim süreleri gerektirir. TLB (Translation Lookaside Buffer) ve sayfa tabloları (page tables) gibi yapılar, bellek erişimlerinde ek gecikmelere neden olabilir. Sayfalama, kesimleme ve bellek koruma mekanizmaları gibi yapılar, işletim sisteminin karmaşıklığını artırır. Mantıksal adresleme ve sanal bellek yönetimi, ek bellek alanı gerektirir. Sayfa tabloları ve segment tabloları gibi yapılar, ek bellek tüketimine yol açar.

Soru 4: Dinamik yer değiştirme (relocation) için kullanılan taban ve limit yazmaçlarının (base and limit registers) görevi nedir?

Dinamik yer değiştirme (dynamic relocation) için kullanılan taban ve limit yazmaçları, bir sürecin bellekteki mantıksal adres alanının fiziksel bellekte güvenli ve izole bir şekilde kullanılmasını sağlar.

Taban yazmacı, bir sürecin mantıksal adreslerinin fiziksel adreslere çevrilmesinde kullanılır ve sürecin bellekte başladığı fiziksel adresi tutar. Bu yazmaç, mantıksal adreslerin fiziksel bellekte nereye yerleştirileceğini belirler. Bir süreç bir bellek adresine erişmek istediğinde, süreç tarafından verilen mantıksal adres, taban yazmacının değeri ile toplanır. Bu toplam, fiziksel adresi oluşturur.
Fiziksel adres = Mantıksal adres + Taban yazmacının değeri

Limit yazmacı, bir sürecin mantıksal adres alanının boyutunu belirler ve sürecin kendi bellek alanının dışına çıkmasını engeller. Bu yazmaç, sürecin erişebileceği maksimum adresi belirler. Bir süreç bir bellek adresine erişmek istediğinde, süreç tarafından verilen mantıksal adres, limit yazmacının değeri ile karşılaştırılır. Eğer mantıksal adres, limit yazmacının değerinden büyükse, bu bellek erişimi geçersiz kabul edilir ve bellek koruma hatası (protection fault) oluşturulur.

Soru 5: Paylaşımlı kütüphane (shared library) kullanmanın avantajı nedir?

Aynı paylaşımlı kütüphaneyi kullanan birden fazla program, kütüphaneyi bellek üzerinde paylaşarak bellek kullanımını optimize eder. Bu paylaşım sayesinde, bellek tüketimi azalır ve sistemin genel performansı artar, özellikle bellek sınırlı sistemlerde büyük avantaj sağlar. Paylaşımlı kütüphaneler merkezi olarak yönetilebilir, bir kütüphanede hata düzeltmesi veya iyileştirme yapılması gerektiğinde, sadece kütüphane dosyasının güncellenmesi yeterlidir. Paylaşımlı kütüphaneler kullanıldığında, yürütülebilir dosyalar (executables) daha küçük olur çünkü kütüphane kodları yürütülebilir dosyanın içine dahil edilmez. Programların belleğe yüklenme süresi hızlı olabilir, çünkü kütüphane kodları zaten bellek üzerindeyse, yeniden yüklenmez. Paylaşımlı kütüphaneler, kodun modüler bir şekilde yapılandırılmasına olanak tanır.

Soru 6: Bellekte boş alanların bir veriyapısında tutulması gerekir. İki adet veri yapısı öneriniz.

Serbest blok listesi (free block list), serbest bellek bloklarının boyut ve adres bilgilerini içeren bir liste yapısıdır. Bu yapı, tipik olarak blok boyutlarına veya adreslerine göre sıralanabilir. Uygun algoritmalarla,

belirli boyuttaki bloklara hızlıca erişim sağlanabilir. Bellek blokları serbest bırakıldığında veya tahsis edildiğinde liste güncellenebilir. Liste, bellek bloklarının dinamik olarak eklenip çıkarılmasını destekler. Uygun boyuttaki bir blok bulunması zaman alabilir, ancak optimize edilebilir. Liste yapısını tutmak için ekstra bellek alanı gerekir.

Bit eşlem (bitmap), bellek bloklarının dolu veya boş olup olmadığını gösteren bitlerden oluşan bir dizidir. Her bit, belirli bir bellek bloğunun durumunu temsil eder (0: boş, 1: dolu). Çok küçük bir bellek kullanarak büyük bellek alanlarının yönetimini sağlar. Uygun algoritmalarla, boş bloklar hızlıca bulunabilir. Bit eşlem yapısı, uygulama ve yönetim açısından oldukça basittir. Bit eşlem genellikle sabit boyutlu bloklarla çalışır, bu da bellek parçalanmasına yol açabilir. Çok büyük bellek alanları için bit eşlem yönetimi karmaşık olabilir.

Soru 7: Bellekte harici ve dahili parçalanma nedir?

Harici parçalanma, bellekte yeterince birçok küçük boşluk (fragment) bulunmasına rağmen, bu boşlukların gereksinim duyulan büyük bir boşluğu karşılayamayacak kadar dağılmış olması durumudur. Uygulamaların talep ettiği bellek blokları serbest veya ayrılmış olabilir, ancak bu boşluklar uygulamaların ihtiyacı olan büyük blokları sağlayacak şekilde birleştirilemez. Bellek tahsisi ve serbest bırakma işlemleri sonucunda, boşluklar bellekte dağınık bir şekilde dağılmış olabilir. Harici parçalanma durumunda, bazı durumlarda küçük boşluklar kullanılabilirken, büyük bloklar için yeterli boşluk sağlanamayabilir. Bellek Yeniden Düzenleme (Memory Compaction): Bellekteki blokların yeniden düzenlenmesi ve boşlukların birleştirilmesiyle büyük bloklar için yeterli boşluk sağlanabilir. İleri Yönlü Tahsis (Forward Allocation): Bellekteki bir sonraki boş bloğun bir sonraki talep edilen blok için kullanılması.

Dahili parçalanma, bellekte tahsis edilen bir bloğun gerçek ihtiyacından daha büyük olması durumudur. Bu durumda, tahsis edilen blok içinde kullanılmayan veya gereksiz boşluklar bulunur. Bellekte belirli bir boyutta bir blok talep edilir, ancak bu blok gereksinimlerin altında kalan küçük bir veri setiyle doldurulur. Dinamik bellek tahsisi sırasında, talep edilen bloğun genellikle belirli bir minimum boyutta olması gerekliliği vardır, bu da gereksiz boş alanlara neden olabilir. Talep edilen bloğun gerçek boyutuna daha yakın olan bir boyutla tahsis edilmesi gerekir.

Soru 8: Sayfa Tablosu'nun (Page Table) görevi nedir?

Sayfa Tablosu (Page Table), sanal bellek yönetimi sürecinde fiziksel bellek (RAM) üzerindeki yerleşimleri kontrol etmek ve sanal adresler ile fiziksel adresler arasındaki eşleşmeyi sağlamak için kullanılan bir veri yapısıdır. Sanal bellek, bir programın kullanabileceği daha büyük bir bellek alanını temsil eder. Her program sanal bellek kullanır ve bu sanal bellek parçaları sayfalara (page) bölünür. Sayfa tablosu, bu sanal sayfaların fiziksel bellekteki (RAM) karşılıklarını (çerçeveler) tutar. Sayfa tablosu, her sanal sayfa numarasını (virtual page number) fiziksel bellekteki bir çerçeve numarasına (frame number) eşler. Sayfa tablosu, hangi sanal sayfaların hangi fiziksel çerçevelere yerleştirildiğini ve böylece bellek yönetimi için hangi kaynakların kullanıldığını gösterir. Sayfa tablosu, bellek erişim haklarını (read-only, read-write, execute vs.) kontrol etmek için kullanılabilir.

Soru 9: Adres alanı: 32 bit olsun. Sayfa boyutu: 4 KB olsun. Sayfa tablosunda her bir eleman 32 bit (4 byte) yer kaplasın. Bu durumda sayfa tablosunun boyutu ne kadardır?

Adres alanı: 32 bit
Sayfa boyutu: 4 KB (4096 byte)
Sayfa tablosunda her bir eleman 32 bit (4 byte) yer kaplar.

4 KB'lik sayfa boyutu olduğunda, adres alanı içinde $2^{32-12} = 2^{20}$ tane sayfa bulunur (her biri 4 KB).
Toplam sayfa sayısı = 2^{20}
Her bir sayfa tablosu öğesi = 4 byte
Sayfa tablosunun toplam boyutu = 2^{20} x 4 byte = 4 Mbyte

Soru 10: Talebe bağlı sayfalama (demand paging) özelliği nedir?

Talebe bağlı sayfalama (demand paging), sanal bellek yönetimi içinde bir stratejidir ve modern işletim sistemlerinin çoğunda kullanılır. Bu özellik, fiziksel bellek (RAM) kullanımını optimize eder ve yalnızca gerektiği zaman sayfaları (veya blokları) belleğe yükler. Talebe bağlı sayfalama, sanal belleğe erişim talepleri doğrultusunda sayfaların (veya blokların) fiziksel belleğe yüklenmesini gerektirir. Bir programın çalıştırılması sırasında, işletim sistemi yalnızca gerektiği zaman sayfa yükler. Bu işlem sayesinde, programın tamamı bellekte olmadan başlatılabilir ve kullanıcının gerçekten eriştiği sayfalar bellekte tutulur. Bir sayfa talebi geldiğinde, sayfa tablosu kullanılarak ilgili sayfanın fiziksel bellekteki konumu belirlenir ve diskten RAM'e taşınır.

Soru 11: Sayfa yer değiştirme (Page Replacement) algoritmalarından dört tanesini açıklayarak yazınız.

Optimal sayfa yer değiştirme algoritması, ideal bir durumda hangi sayfanın diskten çıkarılması gerektiğini belirlemek için kullanılır. Algoritma, her bir sayfa erişiminde gelecekteki kullanımı tahmin ederek en uzun erişim süresine sahip olan sayfayı seçer.

İkinci şans sayfa yer değiştirme algoritması, kullanılan ve kullanılmayan sayfaları ayırt etmek için "referans" veya "ikinci şans" biti kullanır. Algoritma, fiziksel bellekte bulunan her sayfa için bir referans biti bulundurur. İlk önce işaretlenmiş olup olmadığı kontrol edilir ve eğer işaretlenmişse, işareti kaldırılır ve sayfa döngüsü devam eder.

En eski kullanılan (LRU – Least recently used) sayfa yer değiştirme algoritması, en son erişilen sayfanın en az kullanılan sayfa olduğunu varsayar. Yani, en son kullanılan sayfaların en erken döndürülmesini bekler. Bu algoritma, fiziksel bellekteki her sayfaya bir zaman damgası atayarak çalışır ve her erişimde bu damgayı günceller.

Çok kullanılan (MRU – Most recently used) sayfa yer değiştirme algoritması, en son erişilen sayfanın en çok kullanılan sayfa olduğunu varsayar. Yani, en son kullanılan sayfaların en geç döndürülmesini bekler. Bu algoritma, LRU algoritmasının tam tersidir.

Çalışma kümesi (Working Set) yer değiştirme algoritması, programın çalışma kümesini belirlemek için bir zaman penceresi (sliding window) kullanır. Bu algoritma, programın gereksinim duyduğu sayfaların birbirleriyle etkileşimini dikkate alır.

Soru 12: Metin (text) ve ikili (binary) dosya tipi farkları nedir?

Metin dosyaları, insan tarafından okunabilir karakterleri (harfler, rakamlar, semboller, boşluklar vb.) içerir. Genellikle ASCII veya Unicode gibi karakter kodlamalarıyla kodlanır. Satır sonu karakterleri (newline, carriage return gibi) ile ayrılmış satırlardan oluşur. Düz metin içerir ve metin düzenleyiciler veya metin işleme programları tarafından kolayca düzenlenebilir.

İkili dosyalar, bilgisayar tarafından doğrudan anlaşılabilir verileri içerir. Genellikle işlenmiş veri, resim, ses, video gibi yapılandırılmış veri türlerini içerir. Belirli bir yapıya veya veri türüne göre düzenlenmiş bit

düzeyinde verileri içerir. Özel kodlama veya yapılandırma ile saklanır ve genellikle doğrudan insanlar tarafından okunması zor veya imkansızdır. Daha küçük boyutlarda olabilir çünkü veri türlerine veya yapılarına göre optimize edilmiş veri saklama yöntemleri kullanılabilir.

Soru 13: Dosya özniteliklerinden beş tanesini yazınız.

Dosya öznitelikleri, bir dosyanın veya dizinin taşıdığı özellikler veya meta verilerdir. Bu öznitelikler dosya sistemi tarafından yönetilir.

- Dosya İsmi (File Name): Dosyanın adı veya ismi.
- Dosya Türü (File Type): Dosyanın normal dosya mı yoksa sembolik bağlantı (symbolic link) mı olduğunu belirtir.
- Dosya Boyutu (File Size): Dosyanın byte cinsinden boyutu.
- Oluşturma Zamanı (Creation Time): Dosyanın oluşturulduğu zaman.
- Son Erişim Zamanı (Last Access Time): Dosyanın son erişildiği zaman.
- Son Değişiklik Zamanı (Last Modification Time): Dosyanın içeriğinin son değiştirildiği zaman.
- Son Metadata Değişiklik Zamanı (Last Metadata Change Time): Dosya veya dizinin metadata (izinler, sahiplik vb.) son değiştirildiği zaman.
- Dosya İzinleri (File Permissions): Dosyanın veya dizinin sahibi, grup ve diğer kullanıcılar için okuma, yazma ve çalıştırma izinleri.
- Dosya Sahibi (File Owner): Dosyanın sahibi kullanıcı.
- Dosya Grubu (File Group): Dosyanın atanmış olduğu grup.
- Dosya İçerik Türü (File Content Type): İlgili uygulamaların dosyanın içeriğini nasıl işlemesi gerektiğini belirten etiket.
- Dosya Erişim Hakları (File Access Control Lists - ACLs): Gelişmiş erişim kontrolü sağlamak için ek izinler ve kısıtlamalar.

Soru 14: Bir dosyaya erişmek için adres (path) kullanılır. Mutlak ve bağıl adres nedir?

Mutlak adres, dosya veya dizinin kök dizininden başlayarak tam yolunu belirtir. Mutlak adresler, dosyanın veya dizinin tam yerini belirler ve genellikle sistem genelinde benzersizdir.
Örneğin, UNIX/Linux işletim sistemlerinde mutlak adres / ile başlar ve dizinler arası ilerler:

- /home/user/documents/example.txt
- /var/log/syslog

Windows işletim sisteminde mutlak adres C:\ veya başka bir sürücü harfi ile başlar:

- C:\Users\user\Documents\example.txt
- D:\Program Files\program.exe

Bağıl adres, mevcut çalışma dizininden dosya veya dizine olan yolunu belirtir. Bağıl adresler genellikle mevcut süreç veya uygulama tarafından çalışma dizini belirlenir ve bu dizinden dosyalara erişim sağlar.
Örneğin, bir çalışma dizini ~/documents/ olarak varsayıldığında:

- example.txt (~/documents/example.txt anlamına gelir)
- subfolder/image.jpg (~/documents/subfolder/image.jpg anlamına gelir)

Soru 15: Diskte bulunan bloklar dosyalara nasıl tahsis edilir? Var olan yöntemleri kısaca açıklayınız.

Diskte bulunan blokların dosyalara tahsisi, dosya sistemleri tarafından yönetilen bir süreçtir. Dosya sistemleri, dosyaları disk üzerindeki boş alanlara yerleştirmek ve bu alanları etkin bir şekilde kullanmak için çeşitli yöntemler kullanır.

Tek Bağlantılı Liste: Dosyanın blokları birbirine bağlıdır. Her blok, bir sonraki bloğun adresini içerir. Bu yöntemde dosya blokları diskte rastgele yerleşebilir ve dosya sistemine göre dosyanın tüm bloklarını sırasıyla bulmak mümkündür.

Çift Bağlantılı Liste: Her blok, bir önceki ve bir sonraki bloğun adresini içerir. Bu şekilde, dosya hem ileri hem de geri yönde taranabilir.

İndekslenmiş Bağlantılı Liste: İlk birkaç blok, dosyanın indeks blokları olarak kullanılır. İndeks blokları, dosyanın geri kalan bloklarının adreslerini içerir.

Düz İndeksleme (Indexed Allocation): Her dosya için bir indeks bloğu oluşturulur. Bu indeks bloğu, dosyanın tüm bloklarının disk üzerindeki adreslerini içerir. Küçük dosyalar için idealdir ancak büyük dosyalar için çok fazla disk alanı gerektirebilir.

İkinci Düzey İndeksleme (Two-level Indexed Allocation): Birincil indeks blokları, ikincil indeks bloklarının adreslerini içerir. İkincil indeks blokları, dosyanın blok adreslerini tutar. Bu yöntemle büyük dosyaların yönetimi daha etkili olabilir.

Üçüncü Düzey İndeksleme (Three-level Indexed Allocation): İki seviyeli indeksleme yöntemine benzer şekilde çalışır, ancak daha büyük dosyalar için kullanılır. Birinci seviyedeki indeks blokları, ikinci seviyedeki indeks bloklarının adreslerini ve bu bloklar da üçüncü seviyedeki blokların adreslerini tutar.

Soru 16: Tablo (table) kullanılarak tahsisin avantaj ve dezavantajları nelerdir?

Tablo (table) kullanılarak yapılan tahsis, disk bloklarının bir tablo aracılığıyla yönetilmesi anlamına gelir. Bu yöntem, özellikle dosya tahsis tablosu (FAT - File Allocation Table) gibi dosya sistemlerinde yaygındır. Disk bloklarının yönetimi merkezi bir tablo aracılığıyla yapılır, bu da blokların yerlerini ve durumlarını izlemeyi kolaylaştırır. Tablo tabanlı tahsis yöntemleri, genellikle basit ve anlaşılır veri yapıları kullanır, bu da dosya sistemi uygulamalarını ve bakımını kolaylaştırır. Dosya blokları bitişik olmasa bile tabloda izlenebilir, bu da disk alanının daha verimli kullanılmasını sağlar. Dosya büyüdüğünde, tablodaki uygun boş bloklar kullanılarak dosya genişletilebilir. Bu, dosya boyutunun esnek bir şekilde artırılmasına olanak tanır. Tablo yapısı, dosyanın herhangi bir bloğuna doğrudan erişimi sağlar. Bu, doğrusal aramaya göre daha hızlı erişim sağlar. Bloklar arasındaki bağlantılar merkezi tabloda saklandığı için, hata tespiti ve kurtarma işlemleri daha kolaydır. Büyük disklerde veya çok sayıda dosya içeren sistemlerde, tahsis tablosu büyük boyutlara ulaşabilir. Bu, tablonun bellekte tutulmasını ve yönetilmesini zorlaştırabilir. Tahsis tablosunun kendisi de disk alanı kaplar. Çok büyük tablolarda bu alan önemli olabilir. Tahsis tablosunun bozulması durumunda, dosya sisteminin tamamı etkilenebilir.

Soru 17: Güç kesintilerine karşı dosya sistemini korumanın yöntemleri neler olabilir?

Günlükleme, dosya sisteminde yapılan her değişikliğin bir günlük (journal) dosyasına kaydedilmesi yöntemidir. Bu sayede, bir güç kesintisi veya sistem çökmesi durumunda, sistem yeniden başlatıldığında yarıda kalan işlemler tamamlanabilir veya geri alınabilir.

Copy-on-Write, verinin doğrudan üzerine yazmak yerine, yeni veriyi yeni bir konuma yazar ve ardından referansları günceller. Bu yöntem, veri tutarlılığını sağlar.

Verilerin birden fazla kopyasını farklı disklerde saklamak. RAID 1 (Mirroring) bu yöntemi kullanır.

Soru 18: Bellek Eşlemeli Giriş Çıkışın avantajları ve dezavantajları nelerdir?

Bellek eşlemeli giriş çıkış (Memory-mapped I/O) yönteminde, G/Ç aygıtlarına erişim, cihazların bellekteki belirli adreslere eşlenmesiyle yapılır. Bu yöntem, G/Ç aygıtlarını doğrudan bellek adres alanına dahil eder, böylece CPU, G/Ç aygıtlarına erişim için standart bellek erişim komutlarını kullanabilir.

G/Ç aygıtlarına erişim, bellek erişimi gibi olduğu için, yazılım geliştiricilerinin aygıtlarla etkileşim kurmasını basitleştirir. Bu durum, düşük seviyeli G/Ç işlemlerini yönetmeyi kolaylaştırır. G/Ç işlemleri için ek komut setine ihtiyaç duyulmaz. CPU, bellek işlemleriyle G/Ç aygıtlarına erişebilir, bu da komut kümesini sadeleştirir. Bellek erişim hızında G/Ç işlemleri yapılabilir, bu da yüksek hızlı veri transferleri için avantaj sağlar. Aygıt sürücülerinin yazılması ve yönetimi daha kolaydır.

G/Ç aygıtlarının bellek alanını kullanması, bellek adres alanı tahsisinde karmaşıklığa neden olabilir ve adres çakışmaları yaşanabilir. Bellek erişim hataları G/Ç aygıtlarına zarar verebilir. Bu durum, bellek ve G/Ç aygıtları arasında koruma ve güvenlik sorunlarına yol açabilir. Bellek eşlemeli G/Ç desteklemek için donanımın uygun şekilde tasarlanması gerekir. Bu da ek maliyet ve tasarım karmaşıklığı getirebilir.

Soru 19: RAID ne işe yarar? RAID konfigürasyonları neler?

RAID (Redundant Array of Independent Disks), birden fazla fiziksel disk sürücüsünü birleştirerek veri depolama güvenilirliğini, performansını ve kapasitesini artırmak için kullanılır.

RAID 0 (Striping) Veriler, bloklar halinde iki veya daha fazla disk arasında bölünerek yazılır. Her disk, veri bloğunun bir kısmını içerir.

RAID 1 (Mirroring) Veriler, iki veya daha fazla diske aynı anda kopyalanır (aynalanır). Her disk, tam veri kopyasını içerir.

RAID 5 (Striping with Parity) Veriler, üç veya daha fazla disk arasında bölünerek yazılır. Parite bilgisi, verilerin yeniden oluşturulabilmesi için disklere dağıtılır.

RAID 6 (Striping with Double Parity) RAID 5'e benzer, ancak parite bilgisi iki farklı diske yazılır. Bu, iki diskin aynı anda arızalanmasına karşı koruma sağlar.

RAID 10 (RAID 1+0 veya RAID 0+1) RAID 1 ve RAID 0 kombinasyonudur. Veriler önce aynalanır (RAID 1) ve sonra bu aynalanmış veriler şeritlenir (RAID 0).

Soru 20: CD üzerinde olan çizilmelere karşı veriler nasıl korunur?

CD'ler (Compact Disc) üzerinde olan çizilmeler verilerin okunmasını zorlaştırabilir veya imkansız hale getirebilir. Verilerin korunması ve okunabilirliğin artırılması için CD'lerde çeşitli hata düzeltme ve veri koruma teknikleri kullanılır.

CD'ler, hata düzeltme kodlarını kullanarak verileri korur. Bu kodlar, verilerin okunabilirliğini sağlamak için önemli bir rol oynar. Cross-Interleaved Reed-Solomon Code (CIRC): CD'lerde yaygın olarak kullanılan hata düzeltme kodudur.

Parite Bitleri ve EDC (Error Detection Code): CD'lerde, verilerin doğru okunmasını sağlamak için parite bitleri ve hata tespit kodları kullanılır.

Veri Karıştırma (Interleaving): Veriler CD üzerine yazılırken karıştırılır. Bu, verilerin birbiriyle bağlantılı olmayan küçük bloklar halinde düzenlenmesini sağlar.