**MİNİTALK NOTLAR**

**Bit Kaydırma Mantığı Nasıl Çalışır?**

Bit kaydırma (bit shift), bir veya birden fazla bitin belirli bir sayıda pozisyonu sağa veya sola kaydırılması işlemidir. Bu işlem, programlama dillerinde sıklıkla kullanılır ve birçok farklı amaç için kullanılabilir.

Örneğin, bir değerin iki katına çıkarılması veya bölünmesi için kullanılabilir. Ayrıca, bit kaydırma işlemi, belirli bir bitin (0 veya 1) değerini bulmak, belirli bir biti ayarlamak veya sıfırlamak ve bellek kullanımını optimize etmek gibi birçok farklı amaç için kullanılabilir.

Bit kaydırma işlemi, iki tür olabilir: sağa kaydırma (>> operatörü) ve sola kaydırma (<< operatörü). Sağa kaydırma işlemi, belirli bir sayıda bitin sağa doğru kaydırılmasını sağlar. Bu işlem, bir sayının değerini 2^n (n, sağa kaydırma işleminde belirtilen pozisyon sayısıdır) ile bölerek gerçekleştirilebilir.

Örneğin, 11010110 sayısının sağa iki bit kaydırılması, 00110101 sayısına eşittir. Bu işlemde, sağa kaydırılan bitlerin sol tarafındaki boş pozisyonlara sıfır değerleri eklenir.

Sola kaydırma işlemi ise, belirli bir sayıda bitin sola doğru kaydırılmasını sağlar. Bu işlem, bir sayının değerini 2^n (n, sola kaydırma işleminde belirtilen pozisyon sayısıdır) ile çarpma işlemi yaparak gerçekleştirilebilir.

Örneğin, 11010110 sayısının sola iki bit kaydırılması, 01011000 sayısına eşittir. Bu işlemde, kaydırılan bitlerin sağ tarafındaki boş pozisyonlara sıfır değerleri eklenir.

**SIGUSR1 Nedir?**

SIGUSR1, Unix ve Unix benzeri işletim sistemlerinde kullanılan bir sinyal türüdür. "User-defined signal 1" (Kullanıcı tanımlı sinyal 1) anlamına gelir ve özel kullanıcı tanımlı işlemler için kullanılır.

SIGUSR1 sinyali, bir işlem tarafından diğer bir işleme gönderilebilir ve alındığında, alıcı işlem bu sinyali yakalar ve işleyebilir. Bu sinyal, farklı işlemler arasında iletişim sağlamak için kullanılabilir.

Örneğin, bir sunucu programı, bir istemci isteği aldığında SIGUSR1 sinyalini gönderebilir ve bu sinyali alan işlem, istemci isteğini işleyebilir. Ayrıca, bir programın yürütmesi sırasında bazı özel işlemler gerçekleştirilmesi gerektiğinde, SIGUSR1 sinyali kullanılabilir.

SIGUSR1, işletim sistemi tarafından belirtilen sinyaller arasında özel olarak ayrılmıştır ve kullanıcı tarafından tanımlanan sinyaller için önerilir. Bu nedenle, SIGUSR1 ve SIGUSR2, işletim sistemi tarafından özel kullanıcı tanımlı sinyaller olarak ayrılmıştır.

**Signal() Fonksiyonu Nedir?**

**signal()** fonksiyonu, bir sinyal yakalandığında ne yapılacağını belirlemek için kullanılan bir POSIX fonksiyonudur. Bu fonksiyon, yakalanabilecek bir sinyal ve bu sinyal için oluşturulacak bir işlev işaretçisi parametreleri alır. Yakalanabilir bir sinyal olduğunda, işlev işaretçisi çağrılır ve işlevin içeriği çalıştırılır. **signal()** fonksiyonu, programın çalışması sırasında sinyalleri ele almak için kullanılır ve genellikle sinyal yakalama işlevleri oluşturmak için kullanılır.

**Kill() Fonksiyonu Nedir?**

kill() fonksiyonu, Unix ve Unix benzeri işletim sistemlerinde kullanılan bir sistem çağrısıdır. Bu fonksiyon, bir işleme sinyal göndermek için kullanılır.

kill() fonksiyonu, iki parametre alır: hedef işlemin kimliği (PID) ve gönderilecek sinyalin numarası. Bu fonksiyon, bir işlemi sonlandırmak, işlemin çalışma modunu değiştirmek veya işlemle iletişim kurmak gibi birçok farklı amaç için kullanılabilir.

Örneğin, bir program, başka bir programı sonlandırmak için kill() fonksiyonunu kullanabilir. Ayrıca, bir program, diğer bir programı belirli bir işlemi yapmaya zorlamak veya işlemle ilgili bilgi almak için de kill() fonksiyonunu kullanabilir.

kill() fonksiyonu, özellikle Unix ve Unix benzeri işletim sistemlerinde sıklıkla kullanılır ve sistem düzeyinde işlem kontrolü için oldukça önemlidir. Ancak, bu fonksiyonun kötüye kullanılması, ciddi güvenlik açıklarına neden olabilir. Bu nedenle, kill() fonksiyonunun kullanımı, dikkatli bir şekilde yapılmalı ve sadece gerektiği durumlarda kullanılmalıdır.

**Getpid() Fonksiyonu Nedir?**

**getpid()** fonksiyonu, bir işlemin benzersiz kimliğini (Process ID - PID) almak için kullanılan bir POSIX fonksiyonudur. Bir işlem, bir işletim sistemi altında çalışan bir programdır ve her işlemin benzersiz bir PID'si vardır. Bu fonksiyon, işlemi yürüten işletim sistemi tarafından atanmış olan PID'yi döndürür. **getpid()** fonksiyonu, birçok işlem yönetimi işlemi için gereklidir ve çoğu zaman bir işlemi tanımlamak veya işlemle ilgili bilgileri kaydetmek için kullanılır.

**Pause() Fonksiyonu Nedir?**

**pause()** fonksiyonu, işlemi geçici olarak askıya almak ve belirli bir sinyal alınıncaya kadar beklemek için kullanılan bir POSIX fonksiyonudur. Bu fonksiyon, işlemi askıya alır ve sinyal alınıncaya kadar bekler. Bir sinyal alındığında, işlem **pause()** fonksiyonundan çıkar ve sinyali işlemek için belirlenmiş olan sinyal yakalama işlevi çağrılır. **pause()** fonksiyonu, programların belirli bir sinyali beklemesi gereken durumlarda kullanılır ve sinyalleri yakalayan işlevlerle birlikte kullanılır.

**Usleep() Fonksiyonu Nedir?**

**usleep()** fonksiyonu, C dili standart kütüphanesinde yer alan bir fonksiyondur. Bu fonksiyon, mikrosaniye cinsinden belirli bir süre beklemek için kullanılır. Fonksiyonun kullanımı için **unistd.h** başlık dosyası dahil edilmelidir.

**usleep()** fonksiyonu, bir parametre olarak mikrosaniye cinsinden bir bekleme süresi alır. Fonksiyon, verilen süre boyunca programı askıya alır ve ardından çalışmaya devam eder. Bu fonksiyon, programların zamanlama ve eşzamanlılık işlemleri için kullanılır.

**usleep()** fonksiyonu, özellikle mikrodenetleyicilerde bulunan **delay()** veya **wait()** gibi işlevlere benzer şekilde kullanılabilir. Ancak, **usleep()** fonksiyonu, daha kesin bekleme süreleri sağlayabilir. Bu nedenle, programların zamanlama ve eşzamanlılık işlemlerinde tercih edilir.

**Global Değişken Nedir?**

Global değişken, program içinde tanımlandığı andan itibaren geçerli olan ve programın herhangi bir yerinden erişilebilen bir değişkendir. Global değişkenler, programın çalışması sırasında bellek üzerinde sabit bir yer kaplarlar ve herhangi bir fonksiyon çağrısından bağımsız olarak kullanılabilirler.

Global değişkenler, programcılar tarafından genellikle fonksiyonlar arasında paylaşılan verileri saklamak veya programın farklı bölümlerinde kullanılan ortak değişkenleri tanımlamak için kullanılır. Ancak global değişkenlerin aşırı kullanımı, programın karmaşıklığını artırabilir ve programın bakımını zorlaştırabilir. Bu nedenle, global değişkenlerin program içindeki kullanımı dikkatli bir şekilde yapılmalıdır.

Global değişkenler, programcılar tarafından programın başlangıcında tanımlanır ve değerleri programın çalışması sırasında değiştirilebilir. Global değişkenler, static anahtar kelimesi ile de tanımlanabilir, bu durumda sadece program içinde kullanılabilirler ve program dışındaki diğer dosyalar tarafından erişilemezler.

Global değişkenlerin kullanımı, programcıların dikkatli bir şekilde program tasarlamalarını ve global değişkenlerin gereksiz kullanımından kaçınmalarını gerektirir. Global değişkenlerin ölçülü kullanımı, programların daha anlaşılır ve bakımı daha kolay olmasını sağlayabilir.

**PID Nedir?**

PID (Process IDentifier), işlem kimliği anlamına gelir ve her çalışan işleme benzersiz bir sayısal kimlik tanımlayıcısı olarak atanır. Bu kimlik, işlemin sistemdeki diğer işlemlerden ayrılmasını sağlar ve işlem hakkında bilgi toplamak veya işlemleri kontrol etmek için kullanılabilir.

PID numaraları genellikle işletim sistemi çekirdeği tarafından atılır ve sistemdeki her işleme benzersiz bir PID atanır. Bu numaralar, bir işlemi diğerlerinden ayırt etmek için kullanılır ve işlem kontrolü, işlem izleme veya kaynak kullanımı takibi gibi amaçlarla kullanılabilir. PID numaraları, çoğu işletim sistemi tarafından 16 bit, 32 bit veya 64 bit uzunluğunda bir tamsayı olarak temsil edilir.

Bir işlemin PID numarası, işlemi sonlandırmak veya duraklatmak gibi işlem kontrolü işlemlerini gerçekleştirmek için kullanılabilir. Ayrıca, bir işlemin PID numarası, işlem kaynakları veya işlem bellek kullanımı gibi bilgileri toplamak için kullanılabilir.

Bir işlemin PID numarası, genellikle işletim sistemi tarafından atandığı için değiştirilemez. Ancak, bazı durumlarda bir işlem, bir başka işlem için bir PID numarası talep edebilir. Bu durum, genellikle işlem grupları veya alt işlemler gibi karmaşık işlem yapıları oluşturmak için kullanılır.