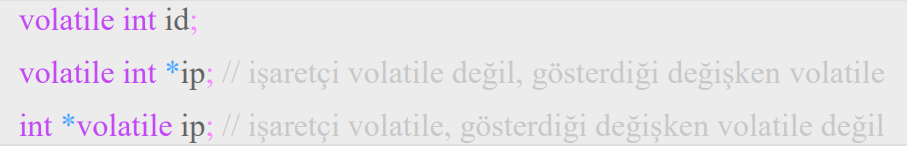
**Depolama Sınıfları**

**volatile değişken tanımlayıcısı**

Eğer bir değişken bildiriminin başına volatile ifadesini getirirsek, derleyici bu değişken değerinin program içinde herhangi bir zamanda, program içinde yer alan kodlar dışında, farklı kaynaklar tarafından değiştirilebileceğinianlar. Burada bahsi geçen kaynaklar, bir kesme (interrupt), harici kesme, doğrudan bellek erişimi veya paylaşılan kaynaklar olabilir. Normal değişkenler kullanıldığında, derleyici iyileştirme işlemi uygular ve değişkeni bellek yerine yazmaca ve ön belleğe yükler. Bir değer volatile olarak tanımlandığında, derleyicinin herhangi bir iyileştirme yapmaması ve değişken değerinin bellekten okunması sağlanır. Bir değişkenin önbellekten register’a aktarılmasını engeller ve değişkenin her erişimde bellekten okunması sağlanır.

****

**static belirleyicisi**

Yalnızca bir kez başlatılır. Derleyici, programın sonuna kadar değişkenle devam eder. Statik değişkenler, fonksiyonun içinde veya dışında tanımlanabilir. Blok için yereldirler. Statik değişkenlerin varsayılan değeri sıfırdır. Statik değişkenler, program kapanıncaya kadar bellekte kalmaya devam eder.

**const değişken tanımlayıcısı**

Değişken bildiriminin başına const ifadesini getirdiğimizde, program değişkenin değerini hiç bir şekilde değiştiremez. const değişkenlere bir ilk değer verilebilir. Bir fonksiyona geçirilen bir parametrenin, fonksiyon tarafından değiştirilmesini önlemek için const değişken tanımlayıcısını kullanabiliriz. Böylece fonksiyona aktardığımız değişken değerinin korunmasını garantiye alırız.

**auto değişken tanımlayıcısı**

auto değişken tanımlayıcısı, bir fonksiyon veya kod bloğu içinde tanımlanan bütün lokal değişkenler için ön tanımlı belirleyici olarak kullanılır. Bir değişken için, bir depolama sınıf belirleyicisi tanımlanmadığında, otomatik olarak auto değişken tanımlayıcısı sahip olacaktır. Auto olarak bildirimi yapılan değişkenlerin kapsamı, tanımlandıkları fonksiyon veya kod bloğu içerisinde olup, bu fonksiyon veya kod bloğunun dışından erişim sağlanmaz. Auto olarak bildirimi yapılan değişkenlerin ömrü, tanımlandıkları fonksiyon veya kod bloğu sona erdiğinde sona erer.

**extern değişken tanımlayıcısı**

C dilinde yazılan uzun programlar derleme zamanını çok artıracağı için, genellikle uzun programlar iki veya daha fazla dosyaya bölünerek derlenir. Bu sistemden aynı zamanda farklı amaçlarla kullanılan kodların düzenlenmesi içinde faydalanılır. Bir proje içinde yer alan bu dosyalar tek bir komutla ayrı ayrı derlendikten sonra birleştirilerek tek bir çalışan .exe uzantılı dosya oluşturulur. Bu durumda, bir dosya içinde tanımladığınız global değişkenler diğer dosya içinde tanınmazlar. Eğer bir dosya içinde tanımladığımız global değişkenlerin diğer dosyalar içinde geçerli olmasını istersek, diğer dosyalarda yer alan bütün fonksiyonların dışında yaptığımız değişken tanımlamalarının baş tarafına extern ifadesini getirmemiz gerekir. Bir değişkeninin başında extern ifadesini kullanmak, o değişkenin projede yer alan kod dosyalarının birinde tanımlandığını ve o değişkene extern ifadesini kullandığımız dosya içinden erişim sağlamak istediğimizi gösterir.

**register değişken tanımlayıcısı**

Diğer bir önemli tip değiştirici register olarak adlandırılır ve bu sözcük sadece int ve char tipi değişkenlere uygulanır. Bir değişken register tipinde tanımlanırsa bu değişken, RAM bellekte değil, mikroişlemcinin registerinde yer alır. Bu nedenle register tipinde tanımlanmış değişkenlerin kullanıldığı programlar diğer programlara göre daha hızlı çalışır. Çünkü registerde bulunan değişkenlere, bellekte bulunan değişkenlerden daha hızlı ulaşır.

Bir değişkene hızlı erişim için auto belirteçi yerine register belirteci kullanmak daha iyidir.

Çünkü, register değişkenleri, register bellekte (CPU belleği) depolanırken auto değişkenleri bellekte depolanır. Sınırlı sayıda değişkenler register bellekte depolanabilir. Yani çok sık kullanılan kayıtlar için register belirteç kullanabilirsiniz. Bu sayede yazdığınız programlar daha hızlı olacaktır.