**AUTO TÜR BELİRLEYİCİSİ**

**(AUTO TYPE DEDUCTION)**

auto anahtar sözcüğü C dilinde bir nesnenin otomatik ömürlü olduğunu göstermek için kullanılan bir anahtar sözcüktür. Global nesnelerin auto ile bildirimi C kurallarına göre geçersizdir. Bu anahtar sözcük C dilinde ve Modern C++ öncesinde kullanımdan ciddi manada düşmüş bir anahtar sözcüktü. C++11 standartlarıyla birlikte auto anahtar sözcüğü C dilinden farklı anlamlar kazandı ve oldukça yaygın kullanılan bir anahtar sözcük haline geldi. Bu anahtar sözcüğün artık otomatik ömürlü nesnelerle doğrudan bir alakası yok. Bu anahtar sözcük artık tür belirlemekte kullanılmaktadır. Örneğin aşağıdaki gibi bir bildirimde x değişkeninin türü int olarak belirlenir.

auto x = 10;

**Anahtar Notlar:** auto ile bildirilmiş bir değişkene ilk değer vermek zorunludur aksi durumda error oluşur.

auto y;//error

Sabit türlerine bağlı olarak aşağıda bazı tür çıkarımları verilmiştir.

Sabitlerle İlgili Bazı Hatırlatmalar:

1. Sayı eğer nokta içermiyorsa, sonuna herhangi bir ek almamışsa ve int sınırları içerisindeyse int türündendir.
2. Sayı eğer nokta içermiyorsa ve sonuna büyük(L) veya küçük(l) almışsa bu durumda long türündendir.
3. Sayı nokta içermiyorsa ve sonuna büyük(U) veya küçük (u) almışsa unsigned int(unsigned) türündendir.
4. Sayı nokta içermiyorsa ve sonuna büyük(LU) veya küçük (lu) almışsa unsigned long türündendir.
5. Sayı nokta içermiyorsa ve sonuna büyük(LLU) veya küçük llu almışsa unsigned long long türündendir.
6. Sayı nokta içermiyorsa ve sonuna büyük(LL) veya küçük(ll) almışsa long long türündendir.
7. Sayı nokta içeriyorsa ve sonuna büyük(F) veya küçük(f) almışsa float türdendir.
8. Sayı nokta içeriyorsa ve sonuna herhangi bir ek almamışsa double türdendir.
9. Sayı nokta içeriyorsa ve sonuna büyük(L) veya küçük(l) almışsa long double türdendir.

Yukarıdaki önermelere dikkat ederek aşağıdaki auto tür çıkarımlarını inceleyiniz:

auto x = 1000L;//Tür çıkarımı long

auto y = 123U;//Tür çıkarımı unsinged int veya unsigned

auto z = 61891LU;// Tür çıkarımı unsigned long

auto k = 17781LLU;//Tür çıkarımı unsigned long long

auto p = 619919LL;//Tür çıkarımı long long int

auto w = 178.98F; //Tür çıkarımı float

auto g = 23.78; //Tür çıkarımı double

auto v = 234.45L; //Tür çıkarımı long double

Bu bölümde sabitler için tür çıkarımlarını inceledik. İkinci kısımda ise değişken ifadeleri için tür çıkarımının nasıl yapılacağını inceleyiniz. Aşağıdaki örnekte görüldüğü gibi auto ile bildirilen bir değişkene bir fonksiyon atanırsa tür çıkarımı bir fonksiyon göstericisi(function pointer) olarak yapılır. Yani f auto olmadan bildirilseydi void (\*f) (int) şeklinde bildirilmeliydi.

void foo(int x)

{

}

int main()

{

auto f = foo;//tür çıkarımı void (\*) (int)

}

Bir dizi eğer auto ile bildirilmiş değişkene atanırsa bu durumda tür çıkarımı pointer türünden yapılır. Yani aşağıdaki örnekte x değişkeni arr dizisinin ilk elemanını gösteren bir pointerdır.

int main()

{

int arr[]{1,2,3};

auto x = arr; // tür çıkarımı int\*

}

int türünden bir değişkenin adresinin auto ile bir bildirilmiş bir değişkene atanması sonucunda tür çıkarımı tahmin edilebileceği gibi int\* türünden yapılır.

int main()

{

int a = 34;

auto x = &a; // tür çıkarımı int\*

}

**Anahtar Notlar**: const int türünden bir değişkenin auto ile bildirilmiş bir değişkene atanması sonucunda constluk düşer ve tür çıkarımı int olarak yapılır.

int main()

{

const int a = 56;

auto x = a; // tür çıkarımı int

}

int& türünden bir değişkene bağlanmış bir referans eğer auto ile bildirilmiş bir değişkene atanırsa bu durumda referanslık düşer ve tür çıkarımı int olarak yapılır.

int main()

{

int x = 35;

int& r = x;

auto k = r;//tür çıkarımı int

}

Bir değişkenin türü const int& ise bu durumda hem referanslık hem constluk düşer son durumda tür çıkarımı int olarak yapılır.

int main()

{

int x = 35;

const int& r = x;

auto k = r;//tür çıkarımı int

}

**Anahtar Notlar:** Referanslarla auto anahtar sözcüğü birlikte kullanımı biraz daha farklı sonuçlar doğurabilir. Örneğin bu durumda constluk da referanslık da düşmek ve tür çıkarımı const int& şeklinde yapılır.

int main()

{

int x = 35;

const int& r = x;

auto& k = r;//tür çıkarımı int

}

Yazılar için tür çıkarımı tahmin edilebileceği gibi const char\* türünden yapılır çünkü C++ dilinde C dilinden farklı olarak yazı türleri char[] değil const char[] türündendir bu sebeple tür çıkarımları bu şekilde yapılır. Hatırlarsak C dilinde char\* türünden bir yazının değiştirme girişimi syntax hatası olmasa da bir undefined behaviour’dır. Bu yüzden zaten C++’ta yazı karakterlerini değiştirme girişimini engellemek için C++ dilinde yazıların const char[] türünden belirlenmiştir bu sayede değiştirme girişimi syntax hatası statüsüne yükseltilmiştir.

auto str = “mustafa”; //tür çıkarımı const char\*

**Anahtar Notlar :** Çok boyutlu dizilerde ise tür çıkarımı biraz daha farklı şekilde yapılmaktadır. Örneğin aşağıdaki örnekte tür çıkarımı int\*[20] olarak yapılmaktadır.

int m[10][20];

auto x = m; //tür çıkarımı int\*[20]

Eğer yukarıdaki örnekteki x değişkeni auto& ile bildirilseydi bu durumda tür çıkarımı

int (&)[10][20] olarak yapılacaktı.

int m[10][20];

auto& x = m; //tür çıkarımı int(&)[10][20]

Çok boyutlu olmayan bir dizide ise doğrudan o diziye referans olucaktır. Örneğin aşağıdaki örnekte tür çıkarımı int(&)[3] şeklinde yapılacaktır.

int arr[]{1,2,3};

auto& x = arr;//tür çıkarımı int(&)[3]

Auto Anahtar sözcüğünün sağladığı bazı kolaylıklar:

1. **Daha Kısa ve Daha Okunabilir Kod**: Özellikle karmaşık türleri olan ifadeler veya döngülerde, türünü açıkça belirtmek yerine "auto" kullanarak kodun daha kısa ve daha okunabilir olmasını sağlayabilirsiniz.

2. **Tür Değişikliklerinden Bağımsızlık:** Bir değişkenin veya ifadenin türünü değiştirmeniz gerektiğinde, "auto" kullanarak kodunuzun bu değişikliğe otomatik olarak adapte olmasını sağlayabilirsiniz.

3. **Tür Tekrarını Azaltma:** Büyük ve karmaşık türleri yazmak yerine, derleyicinin bu türü çıkararak kodunuzu daha temiz ve daha az hata yapma olasılığı yüksek hale getirebilirsiniz.

4. **Şablon Programlamayı Kolaylaştırma:** C++'ın şablon sistemini kullanırken, "auto" kullanarak daha genel ve daha esnek kodlar yazabilirsiniz. Bu, kodunuzun farklı veri türleriyle çalışmasını kolaylaştırır.

5. **Uzun Tür Adlarını Azaltma:** Bazı modern C++ türleri, uzun ve karmaşık isimlere sahip olabilir (örneğin, `std::vector<std::pair<int, std::string>>`). "auto" kullanarak bu türleri daha kolay ve daha hızlı bir şekilde değişkenlere atayabilirsiniz.

6. **Algoritmalar ve Lambda İfadeleriyle Daha İyi Entegrasyon:** C++11'den itibaren "auto", özellikle algoritmalar ve lambda ifadeleriyle kullanıldığında kodun daha esnek ve anlamlı olmasını sağlar.

Özetle, "auto" anahtar sözcüğü, C++'ta kodun okunabilirliğini artırmak, kod tekrarını azaltmak ve genel olarak daha esnek ve temiz kod yazmak için güçlü bir araçtır.

auto anahtar sözcüğüyle ilgili diğer kısımları ilerde ele alacağınız.