TİP DÖNÜŞÜMLERİ

Üstü Kapalı Tip Dönüşümleri

C dilinde üstü kapalı veri tipi dönüşümü (implicit type casting) ya da standart tip dönüşümüdür (standart type casting), otomatik tip dönüşümü olarak da adlandırılır ve programcının açık talimatlar vermesine gerek kalmadan bir veri tipini otomatik olarak başka bir uyumlu tipe dönüştürmesi işlemidir. Şu durumlarda gerçekleşir;

- Dönüşüm farklı veri tiplerinin değerleri üzerinde gerçekleştirilir.
- Farklı bir veri tipi bekleyen bir fonksiyona bir argüman geçirmeniz halinde gerçekleşir.
- Bir veri tipinin değerini başka bir veri tipinin değişkenine atama durumunda gerçekleşir.

Otomatik tip dönüşümde bellekte az yer kaplayan veri tipinden tanımlanmış değişkenler, bellekte çok yer kaplayan değişkenlere atanmaya (widening casting) çalışıldığında otomatik olarak çok yer kaplayanın veri tipine dönüştürülür.

```
//...
char c;   //1 bayt
int i;    //4 bayt
long l;   //8 bayt
float f;  //4 bayt
double d; //8 bayt
c=65;
i=c; //int veri tipine atanmadan önce; char, int veri tipine dönüştürülür.
l=i; //long veri tipine atanmadan önce; int, long veri tipine dönüştürülür.
f=12.50;
d=f; //double veri tipine atanmadan önce; float, double veri tipine dönüştürülür.
//...
```

Ayrıca bellekte çok yer kaplayan veri tipinden tanımlanmış değişkenler, bellekte az yer kaplayan değişkenlere atanmaya (narrowing casting) çalışıldığında, tip dönüşümü otomatik olarak yapılır ancak veri kaybı söz konusu olur.

```
//...
char c;
int i;
float f;
i=4095; //i=0x00000FFF -> 0x00,0x00,0x0F,0xFF olarak 4 bayt
c=i;    /*c=0xFF -> c değişkenine int veri tipinden char veri tipine en anlamsız
          baytı atanır.*/

f=12.50;
/*
Kayan noktalı bir sayıdan tamsayıya atama yapılıyor ise tam kısmı tamsayıya çevrilir ve
sonrasında atama yapılır.
*/
i=f;    //i=12
//...
```

Bilinçli Tip Dönüşümü

Bazen belli işlemleri daha kontrollü yapmak amacıyla programcı yapılan işlemde işlenen veri tipini bir başka ver tipine kasıtlı olarak değiştirir. Bu tür dönüşümüne bilinçli tür dönüşümü (explicit type casting) denir ve tekli tip dönüştürme (unary cast) işleci (operator) ile yapılır.

Diğer tekli işleçler gibi ifadenin (expression) önünde kullanılır ve parantez içerisinde dönüştürülmek istenen veri tipi yazılarak `int j=(int) 12.8/4;` şeklinde kodlanır. Bu işleç, diğer `+`, `-`, `!`, `--`(pre-decrement), `++`(pre-increment) gibi işleçlerle aynı önceliklidir.

Aşağıda buna ilişkin bir örnek program verilmiştir;

```
#include<stdio.h>
int main() {
   float x = 9.9;
   int y = x+3.3;
   int z= (int)(x)+3.3; // x int veri tipine dönüştürülmüş ve ardından toplama yapılmıştır.

   printf("x: %f\n",x);  // x: 9.900000
   printf("y: %d\n",y);  // y: 13
   printf("z: %d\n",z);  // z: 12
   return 0;
}
```

Üstünlük ve zayıflıklar

Tür dönüşümünün üstünlükleri;

- İşlemlerde Esneklik: Farklı veri tiplerini içeren işlemleri gerçekleştirmede esneklik sağlar. Programcının verileri bir tipten diğerine açıkça dönüştürmesini sağlayarak karışık tip aritmetiğini ve diğer işlemleri kolaylaştırır.
- Uyumluluk: Farklı veri tipleri arasında uyumluluğun sağlanmasına yardımcı olur. Programcının verileri belirli bir bağlamda kullanmadan önce uyumlu bir tipe dönüştürmesini sağlayarak veri uyumsuzluğu hatalarını önler.
- Hassasiyet Kontrolü: Hassasiyet kontrolünün kritik olduğu durumlarda, programcının özellikle sayısal işlemlerde veri tipleri arasında dönüşüm yaparak istenen hassasiyeti açıkça belirtmesini sağlar.
- Açıklık: Tip dönüşümü, programcının veri tipini değiştirme niyetini belirterek kodu daha açık hale getirir. Bu, kod okunabilirliğini artırabilir ve işlenen veri tipiyle ilgili kafa karışıklığını azaltabilir.

Tür dönüşümünün zayıf yönleri;

- Hassasiyet Kaybı: Tip dönüştürmenin en büyük dezavantajlarından biri hassasiyet kaybı potansiyelidir. Örneğin, kayan noktalı bir sayıyı tam sayıya dönüştürürken kesirli kısım kesilir ve bu da bilgi kaybına yol açar.
- Çalışma Zamanı Yükü: Bilinçli tip dönüştürme genellikle program yürütme sırasında dönüştürmenin gerçekleştirilmesi gerektiğinden çalışma zamanı yüküne neden olur. Bu ek işlem, özellikle tip dönüştürmenin sık olduğu durumlarda performansı etkileyebilir.
- Derleyici Uyarıları ve Hataları: Yanlış veya güvenli olmayan tip dönüştürme, derleyici uyarılarına veya hatalarına yol açabilir. Örneğin, uyumsuz tipler arasında dönüştürme yapmaya çalışmak veya geçersiz dönüştürme sözdizimi kullanmak, hata ayıklaması zor olabilecek sorunlara yol açabilir.
- Tanımsız Davranış Potansiyeli: Bazı durumlarda, tip dönüştürme, özellikle uyumsuz tipler arasında dönüştürme yaparken veya dönüştürülen değer hedef tipin aralığının dışında olduğunda tanımsız davranışa yol açabilir. Bu, programda öngörülemez duruma neden olabilir.
- Kod Bakım Zorlukları: Bilinçli tip dönüşümüne kodun bakımı ve anlaşılması, özellikle karmaşık veya iç içe ifadelerde gerçekleştirildiğinde, daha zor hale gelebilir. Bu, artan kod karmaşıklığına ve azalan okunabilirliğe yol açabilir.
- Taşınabilirlik Endişeleri: Bilinçli tip dönüşümüne farklı platformlar veya derleyiciler arasında daha az taşınabilir olabilir. C'de tip dönüşümünün davranışı değişebilir ve belirli veri tipi boyutları hakkındaki varsayımlar tüm ortamlarda geçerli olmayabilir.