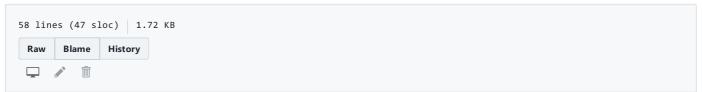




Cplusplus_Ders_Notlari / c_cpp_fark_2.md

Fetching contributors...



Tür Dönüşümlerine İlişkin Farklılıklar

- C dilinde farklı türden adresler arasında otomatik tür dönüşümü var, C++ dilinde yok.
- C'de adres türleri ile tamsayı /gerçek sayı türleri arasında otomatik tür dönüşümü var, C++ dilinde yok.

```
void foo()
{
  int x = 10;
  double dval = 1.2;

int *ptr = x; //C'de geçerli C++'da geçersiz
  ptr = &dval; //C'de geçerli C++'da geçersiz
  int y = ptr; //C'de geçerli C++'da geçersiz
}
```

Böyle otomatik tür dönüşümlerine izin vermek şüphsiz C dilinde de doğru değil. Ancak C derleyicisinin kontrol yükümlülüğü yok. Böyle otomatik tür dönüşümleri C derleyicilerinin hemen hepsinde lojik kontrole takılır ve tipik olarak bir uyarı mesajı alırız.

T herhangi bir tür olmak üzere, C 'de (const T *) türünden (T *) türüne otomatik tür dönüşümü var. C++ dilinde yok.

```
void foo(void)
{
  const int x = 10;
  int y = 20;

int *ptr = &x; //C'de geçerli C++'da geçersiz
  const int *p = &y;

ptr = p; //C'de geçerli C++'da geçersiz
}
```

C 'de (void *) türünden diğer adres türlerine otomatik tür dönüşümü var. C++ dilinde yok.

```
#include <stdlib.h>

void foo(size_t n)
{
  int *p = malloc(n * sizeof(int)); //C'de geçerli C++'da geçersiz
  //...
}
```

C 'de diğer aritmetik türlerde enum türlerine otomatik tür dönüşümü var. C++ 'ta yok. C 'de farklı enum türleri arasında otomatik tür dönüşümü var. C++ 'ta yok.

```
enum Pos {Off, On, Hold, StandBy};
enum Color { Red, Green, Black};

void foo(int val)
{
  enum Pos pos = val;
  enum Color c1 = pos; //C'de geçerli C++'da geçersiz
  enum Color c2 = Off; //C'de geçerli C++'da geçersiz
  int x = pos; //C'de de C++'da da geçerli
}
```

© 2020 GitHub, Inc.

Terms

Privacy

Security

Status Help

Contact GitHub

Pricing

API

Training

Blog

About