**2023.09.13**

Namespace ( İsim alanları )

**void** x**();**

**int** main**()**

**{**

**::**x**()** ***// :: scope resolution operator***

**}**

Namespace nasıl oluşturabiliriz:

* Bir namespace içinde olmalıyız
* Bir fonksiyonun içinde olmaz
* Bir sınıf tanımı içinde olmaz
* Sonuna noktalı virgül koymamamız gerek yok

***// namespace scope***

**namespace** nec

**{**

**int** x **=** 10**;** ***// namespace içinde tanımlanmış global bir değişken***

**void** f**(int);**

**void** f**(double);**

***// function overloding***

**}**

**namespace** a

**{**

**int** x**;**

**}**

**namespace** b

**{**

**int** x**;**

**}**

**int** main**()**

**{**

a**::**x **=** 5**;**

b**::**x **=** 4**;**

**}**

***// ikiside aynı namespace kümülatif olarak birikirler***

**namespace** emre

**{**

**int** a**;**

**int** b**;**

**}**

**namespace** emre

**{**

**int** c**;**

**}**

***// unnamed namespace***

**namespace**

**{**

**int** a**,** b**,** c**;**

**}**

Bir isim alanı içindeki bir ismin nitelenmeden bulunması için:

* using declaration
* using (namespace) direcvtive
* ADL (argument dependent lookup) argümana bağlı isim arama

**Using declaration:**

using std::cout, std::cin;

1) Using bildiriminin bir kapsamı vardır o kapsam içinde etkin olur.

2) Bildirime konu isim bildiriminin yapıldığı isim alanına enjekte edilir

ve o kapsamda tanımlanmış gibi etki eder.

**namespace** emre

**{**

**int** x **=** 5**;**

**}**

**int** main**()**

**{**

**using** emre**::**x**;**

**int** y **=** x**;**

**}**

**namespace** ali

**{**

**int** x**;**

**}**

**namespace** veli

**{**

**using** ali**::**x**;**

**}**

**int** main**()**

**{**

**using** veli**::**x**;**

**}**

**Using namespace direcvtive**

using namespace direcvtive bir bildiirimdir. Tüm bildirimlerde

olduğu gibi bu bildirimin de bir kapsamı vardır ve bu bildirim bildirmin

kapsamında etkindir.

**namespace** ali

**{**

**int** x**,** y**,** z**;**

**}**

**int** main**()**

**{**

**using** **namespace** ali**;**

x **=** 5**;**

y **=** 5**;**

z **=** 456**;**

***// main scope içinde böyle kullanabiliriz***

**}**

**namespace** ali

**{**

**int** x **=** 99**,** y**,** z**;**

**}**

**int** main**()**

**{**

**using** **namespace** ali**;**

std**::**cout **<<** x **<<** "\n"**;** ***// 99 yazar***

**int** x **=** 10**;**

std**::**cout **<<** x **<<** "\n"**;** ***// 10 yazar***

**}**

**namespace** ali

**{**

**int** x **=** 99**;**

**}**

**namespace** veli

**{**

**int** x **=** 99**;**

**}**

**int** main**()**

**{**

**using** **namespace** ali**;**

**using** **namespace** veli**;**

x **=** 4**;** ***// hatalıdır ambigious olur***

**}**

**namespace** ali

**{**

**void** foo**(int);**

**}**

**namespace** veli

**{**

**void** foo**(int,** **int);**

**}**

**int** main**()**

**{**

**using** **namespace** ali**;**

**using** **namespace** veli**;**

foo**(**4**);**

foo**(**4**,** 4**);**

***// function overloading olur***

**}**

**int** g **=** 10**;**

**namespace** ali

**{**

**using** **::**g**;** ***// global scope'taki g yi tanımlamadık***

**}**

**namespace** veli

**{**

**using** ali**::**g**;**

**}**

**int** main**()**

**{**

**++**g**;** ***// 11***

**++**ali**::**g**;** ***// 12***

**++**veli**::**g**;** ***// 13***

std**::**cout **<** "g = " **<<** g **<<** "\n"**;** ***// 13***

**}**

**ADL (argument dependent lookup) argümana bağlı isim arama**

Bir fonksiyona nitelenmemiş bir isimle çağrı yapıldığında eğer fonksiyona gönderilen argümanlardan biri bir namespace içinde tanımlanan türe ilişkin ise söz konusu fonksiyon ismi o namespace içinde de aranır.

**namespace** Nec

**{**

**class** Erg

**{**

**};**

**void** foo**();**

**void** bar**(int);**

**void** baz**(**Erg**);**

**}**

**int** main**()**

**{**

nec**::**Erg e**;**

baz**(**e**)** ***// hata yok***

foo**();** ***// hatalıdır***

bar**(**12**);** ***// hatalıdır***

endl**(**std**::**cout**)**

***// endl std namespace içinde de aranır***

**}**

**namespace** neco

**{**

**class** Nec

**{**

**public:**

**void** foo**(int);**

**}**

**void** func**(**Nec**);**

**}**

#include "nec.h"

**int** main**()**

**{**

neco**::**Nec x**;**

x**.**foo**(**12**);**

func**(**x**);**

**}**

**namespace** ali**::**veli**::**can

**{**

**int** x **=** 5**;**

**}**

**namespace** ali

**{**

**int** a**;**

**}**

**namespace** ali**::**veli

**{**

**int** v**;**

**}**

***// inline with namespace***

**namespace** ali

**{**

**inline** **namespace** old\_version

**{**

**class** Myclass **{};**

**}**

namaspace new\_version

**{**

**class** Myclass**{};**

**}**

**}**

**int** main**()**

**{**

ali**::**Myclass ***// old\_version namespce'deki Myclass kullanılır***

**}**