

7.1 Yaşam alanı (scope)

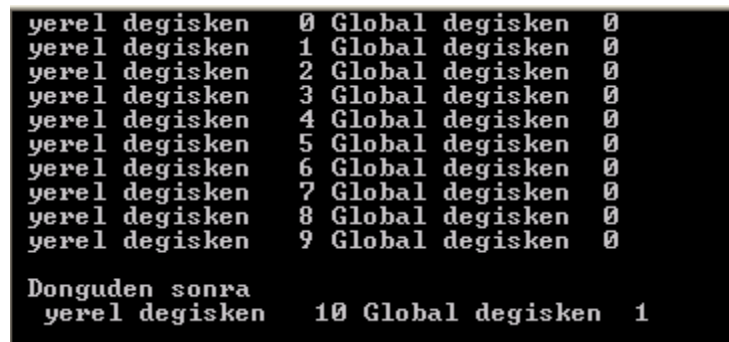
Bir C++ programında global olarak tanımlanan bir değişkenle, yerel olarak tanımlanan bir değişken aynı isme sahip olabilir. Bu durumda yerel olarak tanımlanan değişken global olarak tanımlanan değişkene göre önceliğe sahiptir. Bir başka ifade ile en içte yer alan bloktaki değişken bir üsttekini örter.

Yaşam alanı belirtme (scope Resolution)operatörü(::) kullanılarak örtülmüş olan verilere erişerek kullanmak mümkün olmaktadır. (::) operatörü kısaca kapsam (scope) operatörü olarak da ifade edilmektedir. Aşağıda verilen programı inceleyiniz.

```
#include <iostream.h>
#include <conio.h>
int k = 0;    //global k

int main()
{
    int k;      //yerel k
    for (k = 0; k < 10; ++k) {
        cout<< " yerel degisken  " << k ;
        cout<< " Global degisken  " << ::k <<endl;
    }

    ::k++;
    cout<<endl;
    cout<< " Donguden sonra \n " ;
    cout<< " yerel degisken  " << k ;
    cout<< " Global degisken  " << ::k <<endl;
    getch();
    return 0;
}
```



```
yerel degisken  0 Global degisken  0
yerel degisken  1 Global degisken  0
yerel degisken  2 Global degisken  0
yerel degisken  3 Global degisken  0
yerel degisken  4 Global degisken  0
yerel degisken  5 Global degisken  0
yerel degisken  6 Global degisken  0
yerel degisken  7 Global degisken  0
yerel degisken  8 Global degisken  0
yerel degisken  9 Global degisken  0

Donguden sonra
yerel degisken  10 Global degisken  1
```

İsim uzayı (Name Space)

Takım halinde bir yazılım geliştirmek için programcılar bir araya geldiklerinde, her programcı kendine ait bir takım değişkenler ve fonksiyonlar tanımlar. Programlar bir araya getirildiğinde ise aynı isimde birden fazla değişken ve fonksiyon tanımlaması ile karşılaşılabilir. Bu durumda isim çakışması meydana gelir. Bunu önlemek için C++ de her programcının kendine ait bir isim uzayı tanımlamasına imkan verilmiştir.. Her bir isim uzayındaki değişkenler sadece o isim uzayı içinde tanınırlar. Aşağıda verilen direktif,

using namespace std;

std isim uzayının kullanılacağını belirtir.

cin, cout ve endl fonksiyonları gibi C++ standart kütüphanesinde yer alan tüm değişkenler ve fonksiyonlar bu isim uzayı içinde tanımlanmıştır. Bu direktifin verilmemesi halinde bu fonksiyonları kullanmak için başlarına std yazmak gerekir.

Örnek:

```
Std::cout<<"merhaba"<<endl;  
Std::cin>>x;
```

İsim uzayı tanımlama

Bir isim uzayı aşağıdaki şekilde tanımlanır.

```
namespace tanımlayıcı  
{  
    Değişkenler;  
}
```

Örnek:

```
namespace isimuzayı1  
{  
    int a, b;  
}
```

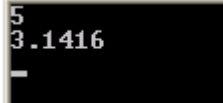
İsimuzayı1 de tanımlanan a ve b değişkenlerine erişmek için aşağıdaki şekilde kullanmak gerekmektedir.

```
isimuzayı1::a=5;  
isimuzayı1::b=7;
```

```
#include <iostream.h>  
#include <conio.h>
```

using namespace std;

```
namespace isimuzayı1  
{  
    int a = 5;  
}  
namespace isimuzayı2  
{  
    double a = 3.1416;  
}  
int main () {  
    cout << isimuzayı1::a << endl;  
    cout << isimuzayı2::a << endl;  
  
    getch();  
    return 0;  
}
```



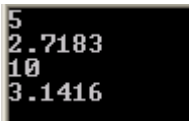
Using bildirimi

Kullanımı kolaylaştırmak ve her değişkeni kullanırken değişken ismi önüne isim uzayı ismini yazmamak için using bildirimi kullanılır. Her bir değişken için using bildirimi ayrı ayrı olarak kullanılabileceği gibi, tüm isim uzayı için de kullanılabilir.

```
#include <iostream.h>
#include <conio.h>
```

```
using namespace std;
namespace isimuzayi1
{
    int x = 5;
    int y = 10;
}
namespace isimuzayi2
{
    double x = 3.1416;
    double y = 2.7183;
}
int main ()
{
    using isimuzayi1::x;
    using isimuzayi2::y;
    cout << x << endl;
    cout << y << endl;
    cout << isimuzayi1::y << endl;
    cout << isimuzayi2::x << endl;

    getch();
    return 0;
}
```



Using bildirimi ile tüm isim uzayı bildirildikten sonra isim uzayı içinde tanımlanmış tüm değişkenleri kullanırken önünde isim uzayı ismini belirtmeye gerek kalmaz. Bu durumda using namespace olarak kullanılır.

```

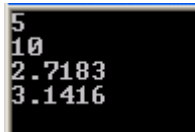
#include <iostream.h>
#include <conio.h>

using namespace std;
namespace isimuzayi1
{
    int x = 5;
    int y = 10;
}
namespace isimuzayi2
{
    double x = 3.1416;
    double y = 2.7183;
}

int main ()
{
    using namespace isimuzayi1;
    cout << x << endl;
    cout << y << endl;
    cout << isimuzayi2::y << endl;
    cout << isimuzayi2::x << endl;

    getch();
    return 0;
}

```



```

5
10
2.7183
3.1416

```

İsimuzayı aşağıdaki gibi de kullanılabilir.

```

#include <iostream.h>
#include <conio.h>
using namespace std;

namespace isimuzayi1
{
    int x = 5;
    int y = 10;
}
namespace isimuzayi2
{
    double x = 3.1416;
    double y = 2.7183;
}

```

```
int main ()
{
    {
        using namespace isimuzayi1;
        cout << x << endl;
        cout << y << endl;
    }

    {
        using namespace isimuzayi2;
        cout << y << endl;
        cout << x << endl;
    }
    getch();
    return 0;
}
```