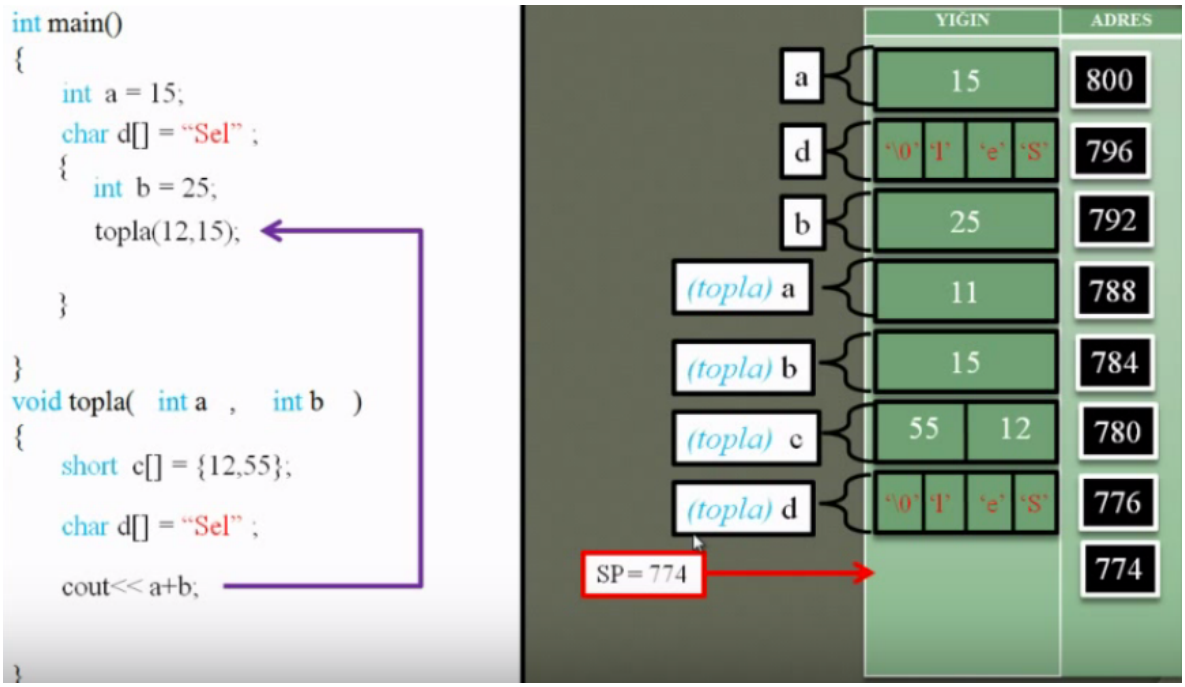
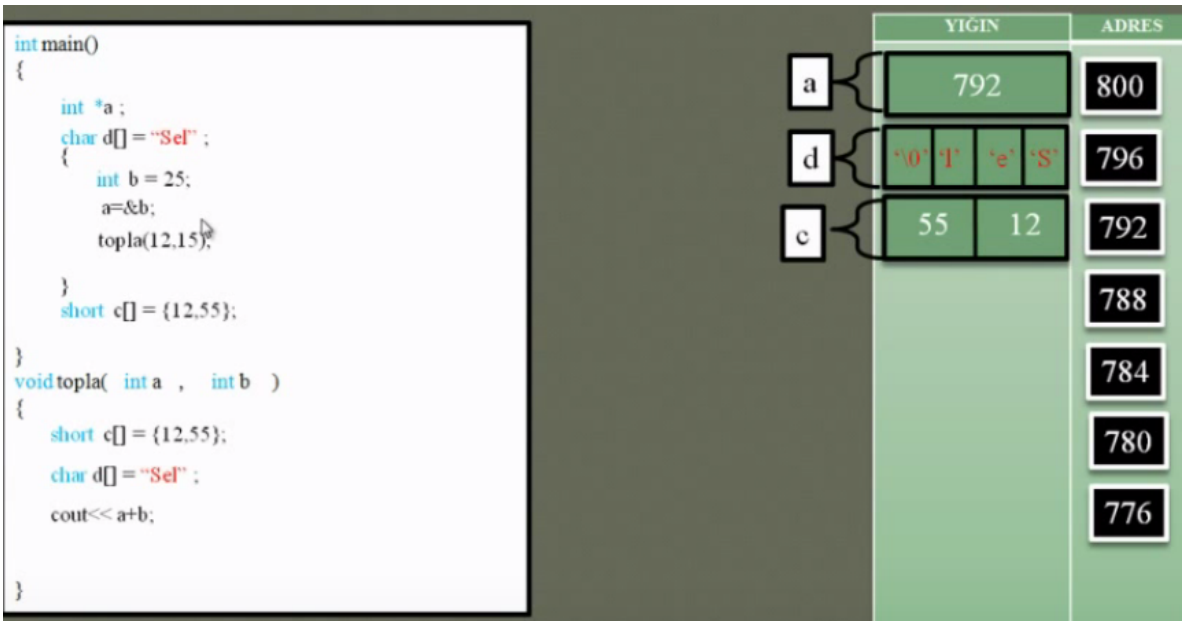
Pointer II Programlama TV
Atanmış ve Atanmamış Global değişkenler **Heap**'de. Dinamik değişkenler için **Heap** kullanılır.

Pointerlar da dahi, yerel (lokal) değişkenler ve fonksiyon çağrıları için **Stack(Yığın)** kullanılır.
Stack kullanabilmek için **SP (Stack Pointer)** ihtiyaç var, içinde adres tutar.

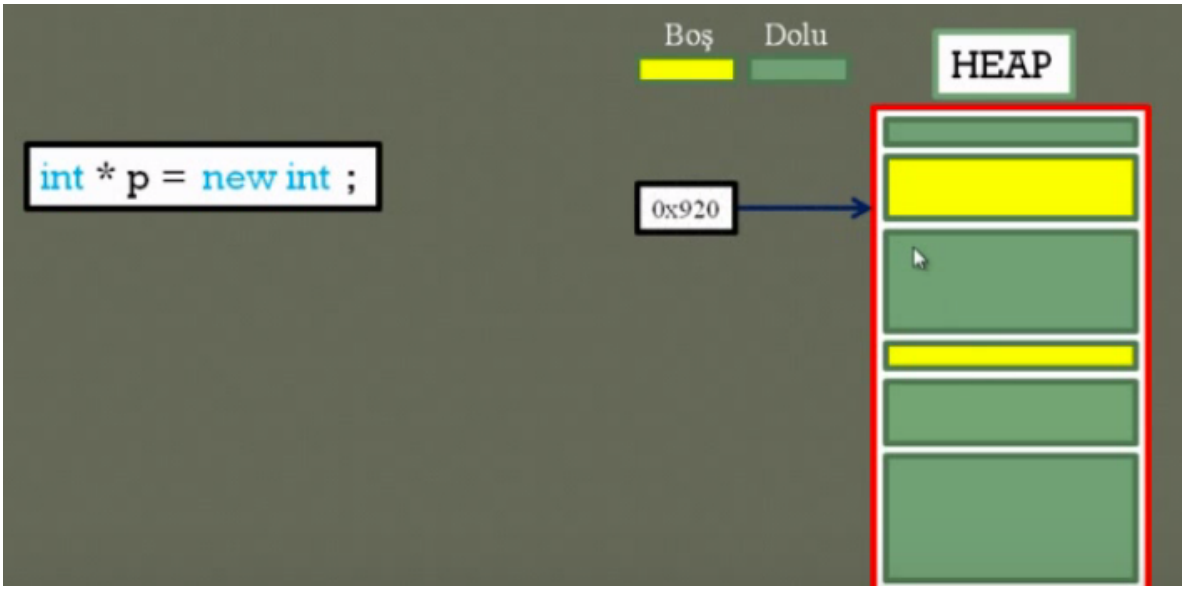


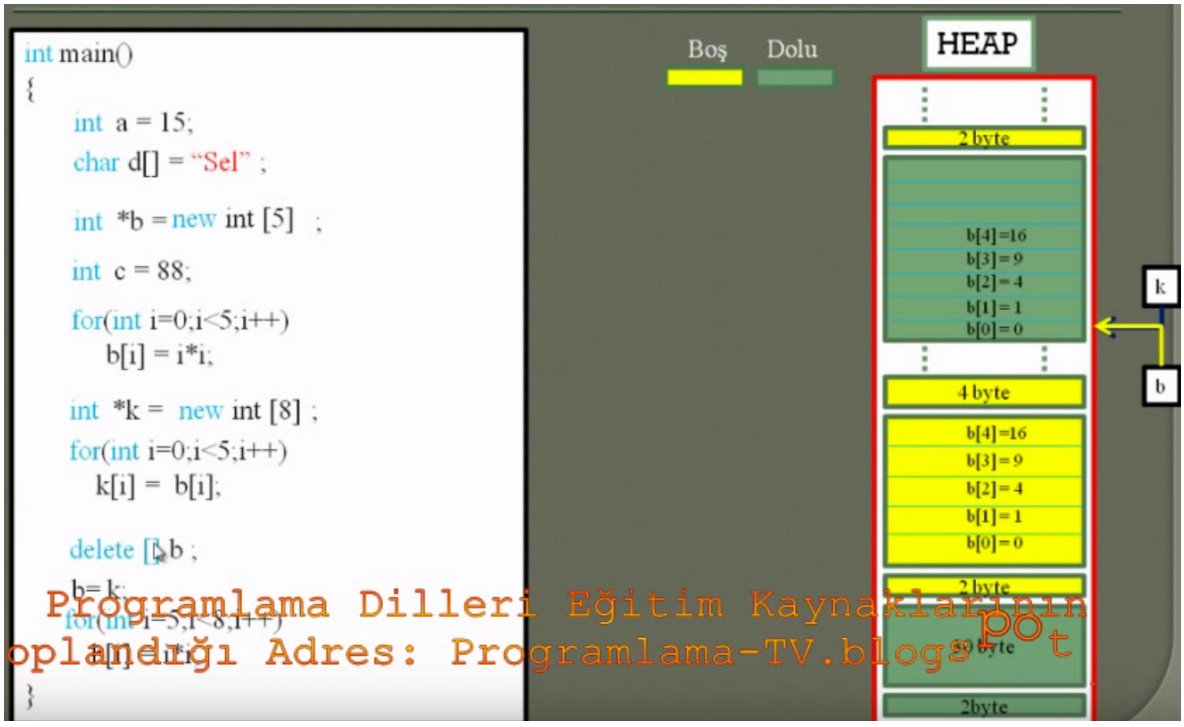
a pointerı b'nin adresini tutuyordu. Fonksiyon okunduktan sonra oluşturalan değişkenlerin etki alanı kalktığı için yeni tanımda
(short c [] = {12,55};) b'nin adresine (792) c değişkeni yazıldı ve a pointerı artık onu gösteriyor. Bu korkunç bir hataya sebep olabilir.



-Yığın'ın (**Stack**) kontrolü programcıda değildir.
-Bu alanda yaratılan değişkenler etki alanları bittiğinde hafızadan kaldırılırlar.
-Global değişkenler bu dez avantajı ortadan kaldırmaktadır.
-Fakat global değişkenler de hafızadan hiç kaldırılmamaktadır.
-**HEAP** bunlara çözüm olarak üretilmiştir.

- **HEAP** hafıza bölgesinde değişken oluşturabilmek için **new** operatörünü kullanmak gerekiyor. (Sadece pointerlar ile erişebiliriz.)
- **int** *p = **new int;** (Derleyici arka planda HEAP'den dört baytlık alan alıyor, o alanın başlangıç adresini pointer'a atıyor.)
- **HEAP** hafıza alanının kontrolü programcıya aittir.





- **Void Pointers:** Yalnızca adres saklamak için kullanılır. Bu yüzden diğer göstericiler arasında atama işlemlerinde kullanılabilir

int tam = 66; **char** y='Z', *pc=&y;

```
void * veri;
veri = &tam;
cout << * (int *)veri << endl;
cout << (void*)pc << endl;
```

```
#include <iostream>
using namespace std;

int dizi_topla(int A[], int n) {
    int *p, toplam = 0;

    for (p = A; p<&A[n] ;p++)
        toplam+= (*p);
    return toplam;
}

int main (){
    int A[5] = { 3,5,6,8,10 };

    cout<<"Toplam:" << dizi_topla(A, 5)<<endl;
    return 0;
}
```

```cpp
#include <iostream>
using namespace std;

void swap(int &a, int &b) {

    int temp;

    temp = a;
    a = b;
    b = temp;
}

int main (){
    int x = 10, y = 20;

    swap(x, y);
    cout<<"x = " << x << "  y = "<< y <<endl;

    return 0;
}
```