Çok Boyutlu Diziler
Önbellek Performans İyileştirmesi

BIL-304: Bilgisayar Mimarisi

Dersi veren öğretim üyesi:
Dr. Öğr. Üyesi Fatih Gökçe

Ders kitabına ait sunum dosyalarından adapte edilmiştir: http://csapp.cs.cmu.edu/

Adapted from slides of the textbook: http://csapp.cs.cmu.edu/

Çok boyutlu (Nested) diziler

- **Tanımlama**

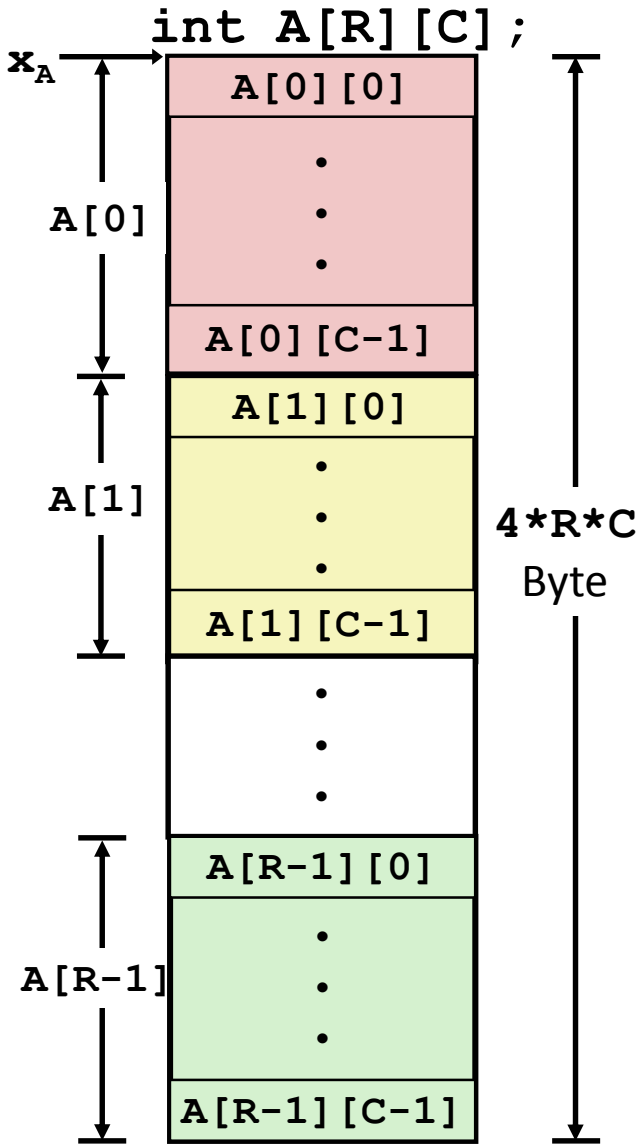
  T `A[R][C];`

  - *T* veri tipinde 2-boyutlu dizi
  - *R* satırlar(rows), *C* sütunlar (columns)
  - *T* veri tipindeki elemanlar *K* byte'lık alana ihtiyaç duyarlar

- **Dizi boyutu**
  - *R* * *C* * *K* byte

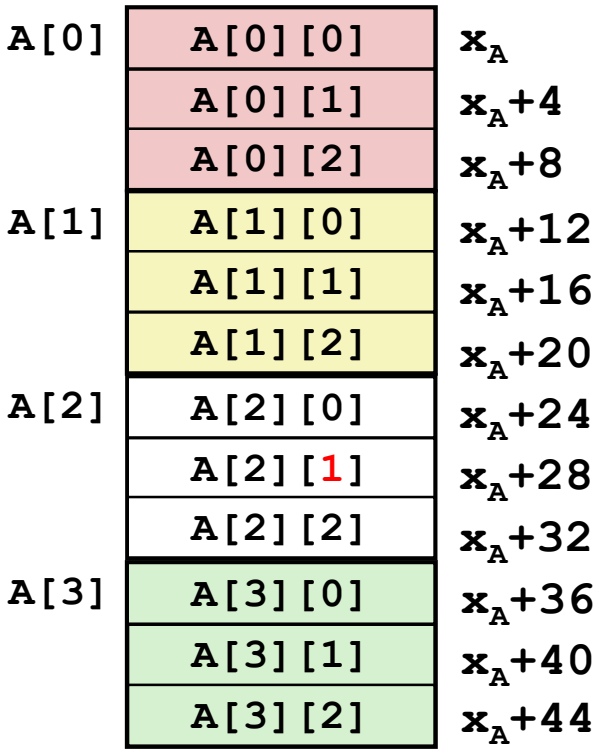
- **Dizilim**
  - Satır öncelikli sıralama

$$\begin{bmatrix} A[0][0] & \cdots & A[0][C-1] \\ \vdots & & \vdots \\ A[R-1][0] & \cdots & A[R-1][C-1] \end{bmatrix}$$



Çok boyutlu (Nested) diziler

```
int A[4][3];
```



Çok boyutlu (Nested) diziler

- **Dizideki elemanların adresleri**

  T `D[`***R***`][`***C***`];`

  - *T* veri tipinde 2-boyutlu D dizisi için
  - `D[i][j]` elemanının adresi:

    $\&D[i][j] = x_D + L(C \cdot i + j)$

    L : ilgili veri tipinin kaç byte olduğu

    C : dizinin sütun sayısı

Çok boyutlu (Nested) diziler

```
int A[4][3];
```

A[0]	A[0][0]	x_A
	A[0][1]	x_A+4
	A[0][2]	x_A+8
A[1]	A[1][0]	x_A+12
	A[1][1]	x_A+16
	A[1][2]	x_A+20
A[2]	A[2][0]	x_A+24
	A[2][1]	x_A+28
	A[2][2]	x_A+32
A[3]	A[3][0]	x_A+36
	A[3][1]	x_A+40
	A[3][2]	x_A+44

```
# %rdi=x_A, %rsi=i, %rdx=j
leaq (%rsi,%rsi,2), %rax      # %rax = 3i
leaq (%rdi,%rax,4), %rax      # %rax = x_A+12i
movl (%rax,%rdx,4), %eax      # %eax = M[x_A+12i+4j]

# x_A+12i+4 = x_A+4(3i+j)
```

Çok boyutlu (Nested) diziler

Soru:

```
#define M …
#define N …
long P[M][N];
long Q[N][M];
long topla(long i, long j){
   return P[i][j]+Q[j][i];
}
```

Kaynak kodu ile verilen kod parçası uygun diğer fonksiyonlar eklenerek derlendiğinde **topla** fonksiyonu için aşağıdaki assembly kodu elde ediliyorsa M ve N'nin değeri nedir?

```
# %rdi=i, %rsi=j
topla:
   leaq 0(,%rdi,8), %rdx
   subq %rdi, %rdx
   addq %rsi, %rdx
   leaq (%rsi,%rsi,4), %rax
   addq %rax, %rdi
   movq Q(,%rdi,8), %rax
   addq P(,%rdx,8), %rax
   ret
```

Çok boyutlu (Nested) diziler

Cevap:

```
# %rdi=i, %rsi=j
topla:
   leaq 0(,%rdi,8), %rdx       # %rdx = 8i
   subq %rdi, %rdx             # %rdx = 7i
   addq %rsi, %rdx             # %rdx = 7i+j
   leaq (%rsi,%rsi,4), %rax    # %rax = 5j
   addq %rax, %rdi             # %rdi = 5j+i
   movq Q(,%rdi,8), %rax       # %rax = M[x_Q+8(5j+i)] → 5, Q dizisinin
                               # sütun sayısı olan M'ye eşittir
   addq P(,%rdx,8), %rax       # %rax += M[x_P+8(7i+j)] → 7, P dizisinin
                               # sütun sayısı olan N'ye eşittir

   ret
```

Böylece M = 5 ve N = 7'dir.

Önbellek performans iyileştirme:

Soru:
Aşağıdaki fonksiyon önbelleği etkin kullanmakta mıdır? **a** dizisinin elemanlarına 1-adımlık hafıza erişimleri ile erişebilmesi için gerekli değişikliği yapınız.

```
int carp(int a[M][N][T])
{
   int i, j, k, carpim = 1;

   for (i = 0; i <= T-1; i++) {
      for (j = 0; j <= M-1; j++ {
         for (k = 0; k <= N-1; k++) {
            carpim *= a[j][k][i];
         }
      }
   }
   return carpim;
}
```

Önbellek performans iyileştirme:

Cevap:

```
int carp(int a[M][N][T])
{
   int i, j, k, carpim = 1;

   for (j = 0; j <= M-1; j++) {
      for (k = 0; k <= N-1; k++) {
         for (i = 0; i <= T-1; i++) {
            carpim *= a[j][k][i];
         }
      }
   }
   return carpim;
}
```