

Two Dimensional Arrays in C/C++



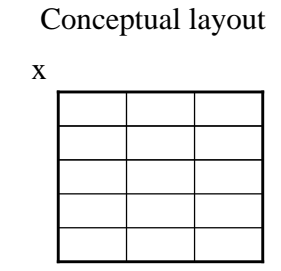
C++ Object Oriented Programming
 Pei-yih Ting
 NTOU CS

Version 1. Fixed dimensions 5 by 3

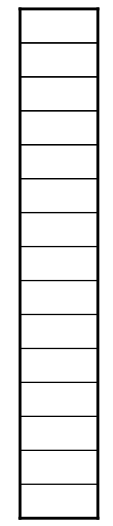
- Both dimensions are fixed
- Allocated either in data segment or in stack
- Example

```
int i, j;
int x[5][3];

for (i=0; i<5; i++)
    for (j=0; j<3; j++)
        x[i][j] = 0;
```



Physical layout



Version 2a. Dynamic allocated 5 by n

- The first dimension is fixed as 5, the second dimension is variable
- Allocated on the stack (x[]) and the heap (x[i][])
- Example

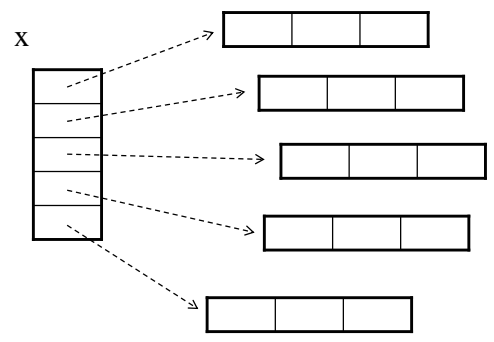
```
int i, j, n=3;
int *x[5];

for (i=0; i<5; i++)
    x[i] = new int[n];

for (i=0; i<5; i++)
    for (j=0; j<n; j++)
        x[i][j] = 0;

for (i=0; i<5; i++)
    delete[] x[i];
```

Conceptual layout



Version 2b. Dynamic allocated m by n

- Both dimensions are variable
- Both allocated on the heap
- Example

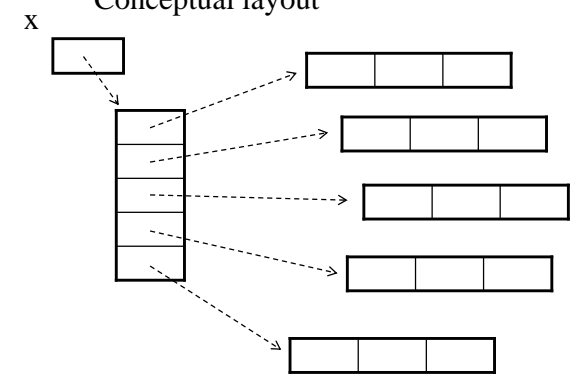
```
int i, j, m=5, n=3;
int **x;

x = new int*[m];
for (i=0; i<m; i++)
    x[i] = new int[n];

for (i=0; i<m; i++)
    for (j=0; j<n; j++)
        x[i][j] = 0;

for (i=0; i<m; i++)
    delete[] x[i];
delete[] x;
```

Conceptual layout



Version 3. Dynamic allocated m by 3

- ◇ The first dimension is variable, the second dimension is fixed as 3
- ◇ Allocated on the heap
- ◇ Example

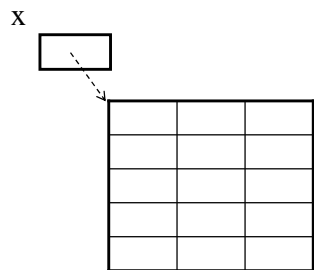
```
int i, j, m=5;
int (*x)[3];
```

```
x = new int[m][3];
```

```
for (i=0; i<m; i++)
    for (j=0; j<3; j++)
        x[i][j] = 0;
```

```
delete[] x;
```

Conceptual layout



5

Version 4. Dynamic allocated m by n

- ◇ Both dimensions are variable
- ◇ Allocated on the heap
- ◇ Example

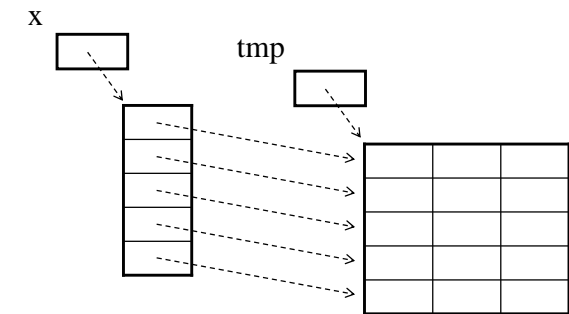
```
int i, j, m=5, n=3;
int **x, *tmp;
```

```
x = new int*[m];
tmp = new int[m*n];
for (i=0; i<m; i++)
    x[i] = &tmp[i*n];
```

```
for (i=0; i<m; i++)
    for (j=0; j<n; j++)
        x[i][j] = 0;
```

```
delete[] x[0];
delete[] x;
```

Conceptual layout



6

Version 5. Dynamic allocated m by n

- ◇ Both dimensions are variable, emulate with 1-D array syntax
- ◇ Allocated on the heap
- ◇ Example

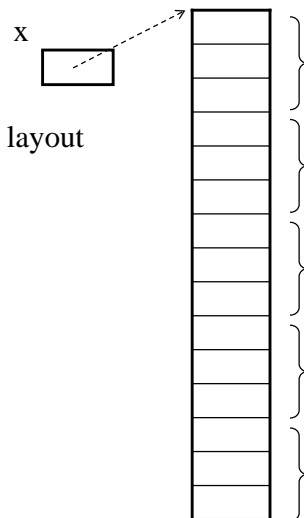
```
int i, j, m=5, n=3;
int *x;
```

```
x = new int[m*n];
```

```
for (i=0; i<m; i++)
    for (j=0; j<n; j++)
        x[i*n+j] = 0; // x[i][j] does not work
                        // (&x[i*n])[j] is OK
```

```
delete[] x;
```

Physical layout



7