Allocazione dinamica di una matrice

- Le funzioni di allocazione dinamica di memoria (malloc e calloc) restituiscono il puntatore ad una zona di memoria **non strutturata**

- Assegnando il valore di ritorno ad un puntatore di un tipo **T**, la zona di memoria **acquista** la struttura del tipo **T**.

```
typdef
   struct {
      int x;
      double y;
      char s[9];
   } T
```

```
T* p = (T *)malloc(sizeof(T));
```

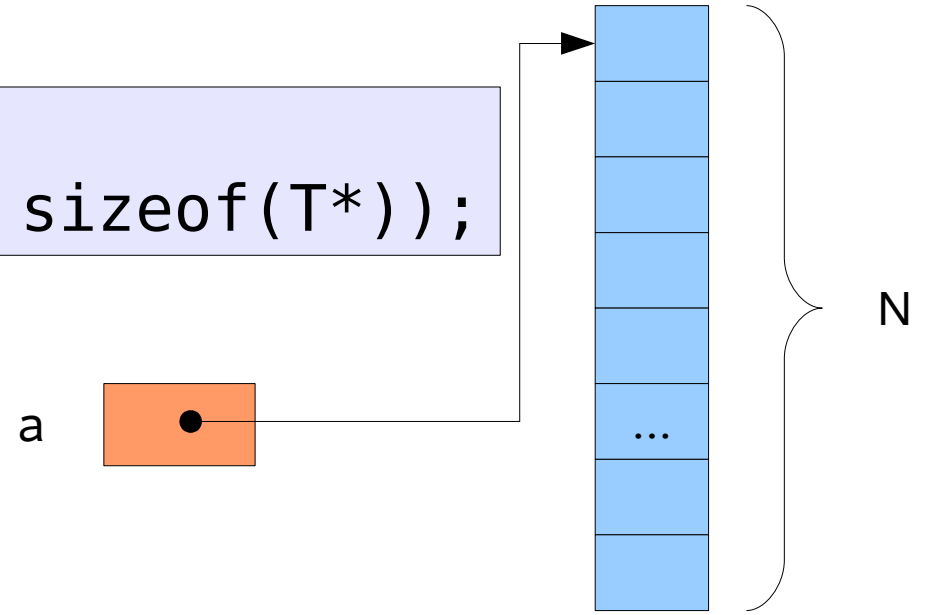
Allocazione dinamica di una matrice

- Se **T** è una struct o un array unidimensionale non ci sono problemi ad accedere ai componenti: es.
  - `p->x` se **T** è struttura avente un campo di nome x
  - `p[5]` se **T** è ad es. un intero (p punta quindi ad un array di interi)
- Il problema sorge se vogliamo allocare della memoria per una **matrice N x M**
- **Non è possibile fare il casting ad un tipo array**

Allocazione dinamica di una matrice Soluzione 1

- Vogliamo allocare una matrice N x M di elementi di tipo T
- si alloca un array di N puntatori a T e si assegna il puntatore a una variabile a di tipo T **

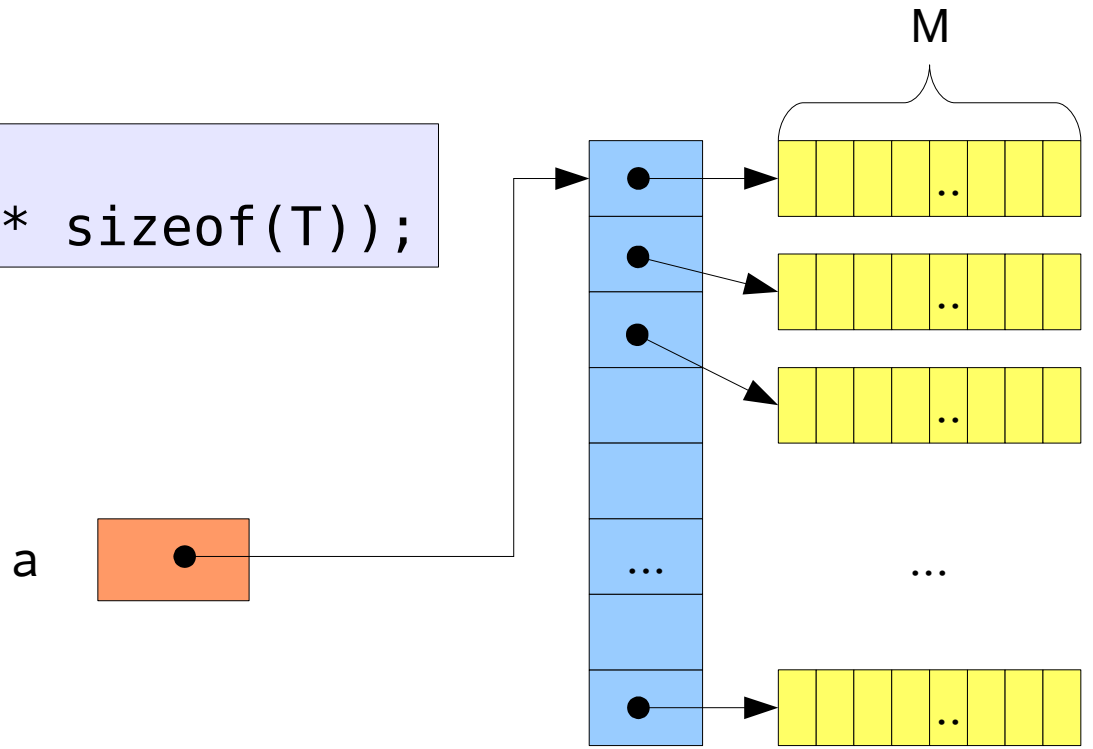
```
T **a;
a=(T**)malloc(N * sizeof(T*));
```



Allocazione dinamica di una matrice Soluzione 1

- per ogni elemento a[i] dell'array puntato da m (che sarà di tipo T*) si alloca un array di M elementi di tipo T il cui indirizzo viene posto in a[i]

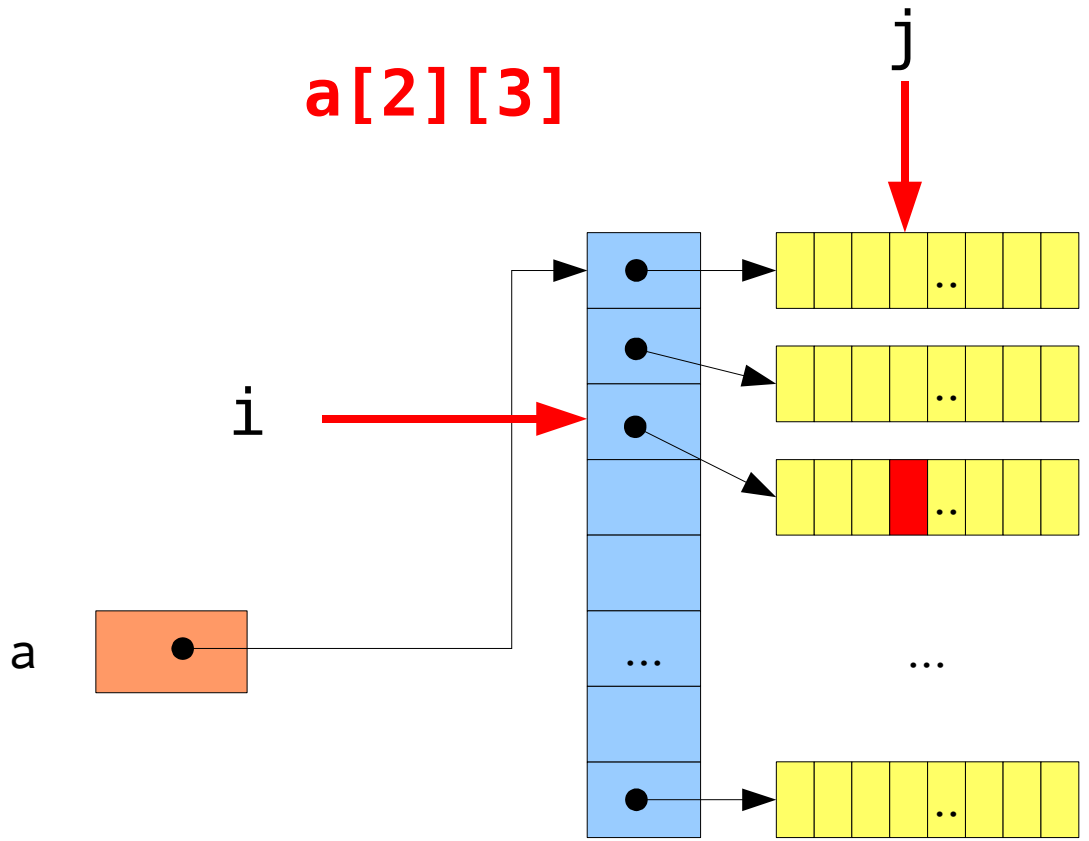
```
for (i=0; i<N; ++i)
   a[i]=(T*)malloc(M * sizeof(T));
```



Allocazione dinamica di una matrice Soluzione 1

- Si accede ad ogni elemento della matrice utilzzando la notazione `a[i][j]`

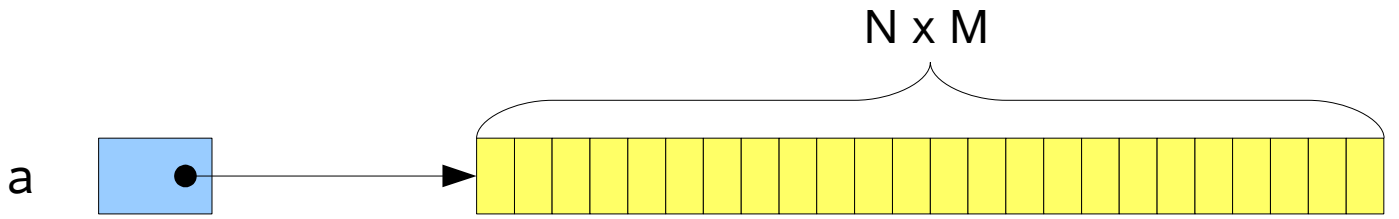
```
for (i=0; i<N; ++i)
   for (j=0; j<M ++j)
      a[i][j]=....;
```



Allocazione dinamica di una matrice Soluzione 2

- Si alloca una zona di N x M elementi di tipo T e la si assegna ad una variabile a di tipo (void *)

```
void *a;
a=malloc(N * M * sizeof(T));
```

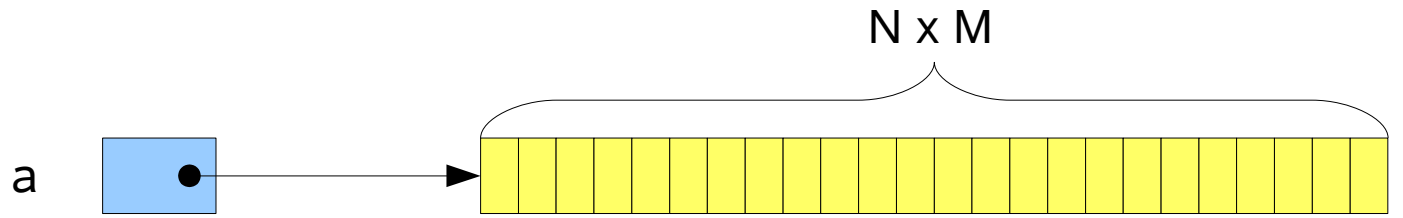


Allocazione dinamica di una matrice Soluzione 2

- Si passa il puntatore a ad una funzione che avrà come parametri le dimensioni della matrice e una matrice a dimensioni variabili di tipo T

```
f(N,M,a)
```

```
... f(int n, int m, T a[n][m]){
      ...
}
```

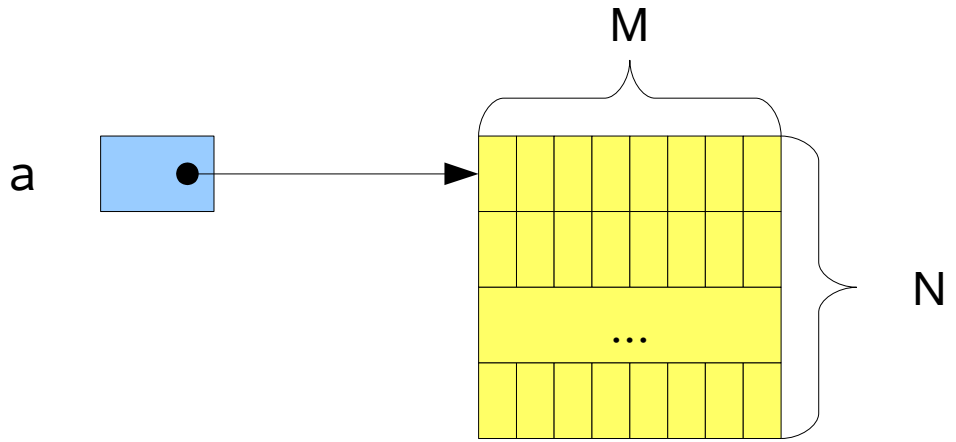


Allocazione dinamica di una matrice Soluzione 2

- Internamente alla funzione, la zona di memoria **acquista** la struttura di un matrice. Il **mapping** degli elementi è delegato al compilatore.

```
f(N,M,a)
```

```
... f(int n, int m, T a[n][m]){
      ...
}
```



Allocazione dinamica di una matrice Soluzione 2

- Si accede normalmente agli elementi della matrice come **a[i][j]**

```
...  f(int n, int m, T a[n][m]){
     ...
      for (i=0; i<n; ++i)
         for (j=0; j<m; j++)
            a[i][j]=...
}
```