**13° lezione prog1**

**Come stampare gli indirizzi di un vettore**

Supponendo di avere un vettore popolato con numeri pseudocasuali, per stampare gli indirizzi di ogni cella dell’array si procede nel seguente modo:

-   **Ciclo for che itera fino a <dim(dim-1)**

-   **printf in cui si va  a stampare l’indirizzo con %p e &v[i][j] vado a stampare l’indirizzo allo posizione i,j**

Le matrici(array a due dimensioni righe, colonne) sono formate da vettori riga e vettori colonna, sono posizionate in modo contiguo in memoria. La distanza tra una cella di memoria all’ altra è di 4 byte.

Il nome della matrice è un puntatore alla prima riga, invece, m+1 è un puntatore alla 2° riga. \*(m+1) punta al 1° elemento della seconda riga. Per ottenere il dato si deve deferenziare il puntatore in questo modo:

-   **\*(v+i)** **per saltare alla riga di indice i e deferenzio per ottenere il primo elemento della riga i esima**.

-   **Aggiungo un offset \*(\*(v+i)+j)**

**Questa procedura si può applicare k volte e pertanto avrà k- livelli di derefenziazione. Si può scrivere anche nel seguente modo: \*(v[i] +j)**

**Allocazione dinamica di una matrice (HEAP)**

La differenza tra l’allocazione dinamica e automatica di una matrice è che nella dinamica si utilizza lo **STACK** e nelle dinamica si utilizza lo **HEAP**.

Immagine che contiene testo, schermata, diagramma, Carattere

Descrizione generata automaticamente

Per un Array k-Dimensionale  con k  maggiore uguale a 2 **il modello** **di memoria cambia e  pertanto cambia anche la struttura dei puntatori**. Per .”allocare un array bidimensionale con allocazione automatica. di dimensione N X N si procede nel seguente modo:

* Si va ad allocare dinamicamente in modo programmato con un loop n vettore riga o n vettori colonne HEAP

 si vanno ad allocare n puntatori al tipo (int \*).  Si deduce che ogni elemento del vettore e’ un int\*. Il vettore di puntatori si indicizza con il nome di questo(Il nome del vettore è un doppio puntatore che punta Il primo elemento del vettore). Immagine che contiene testo, Carattere, schermata, bianco

Descrizione generata automaticamente

Si stanno eseguendo due operazioni:

1. **La creazione del vettore tiene puntatori al tipo e quindi la sua  allocazione**
2. **allocazione degli n vettori di m elementi**

 nelle immagini di sopra  ne sono le righe ed m sono le colonne.

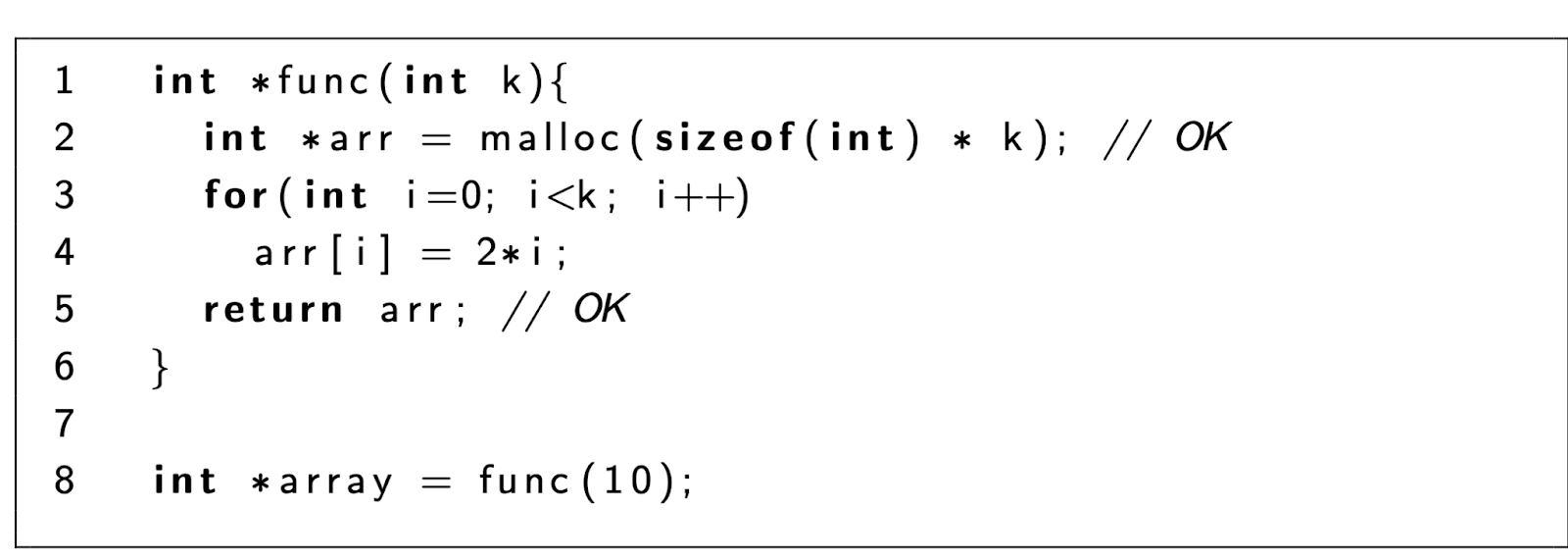
**Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente**

**ci sono diversi modi per accedere agli elementi di una matrice**

Si utilizza quando si devono conservare tanti dati.

**A differenza della allocazione automatica nella dinamica si può avere una funzione che restituisce un puntatore. Si potrebbe fare una funzione che alloca i dati e restituisce il puntatore a quei dati.**



Per de-allocare una matrice bisogna fare un ciclo che parte da 0 fino a n in modo da fare tanti free() nel seguente modo:

for(short i=0; i<N; i++) {free(V[i]);} (tutto il vettore riga)

I puntatori alla fine del suo utilizzo si iniziali iniziano sempre con null  in modo che quando si vanno a utilizzare  si sicuri che il puntatore  non sia stato deallocato.La free si chiama una sola volta se si chiamasse più di una volta se verrebbe un errore grave è un comportamento indefinito.

**Funzioni calloc() e realloc()**

* **calloc() è** la stessa della malloc (appartiene alla stessa famiglia) serve sempre per l’allocazione dinamica ha due parametri

1. **dimensione del singolo elemento da allocare**
2. **numero d elementi totali da allocare (con tutti i byte a 0)**

****

* **realloc()** consente di ridimensionare un blocco di memoria

Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente