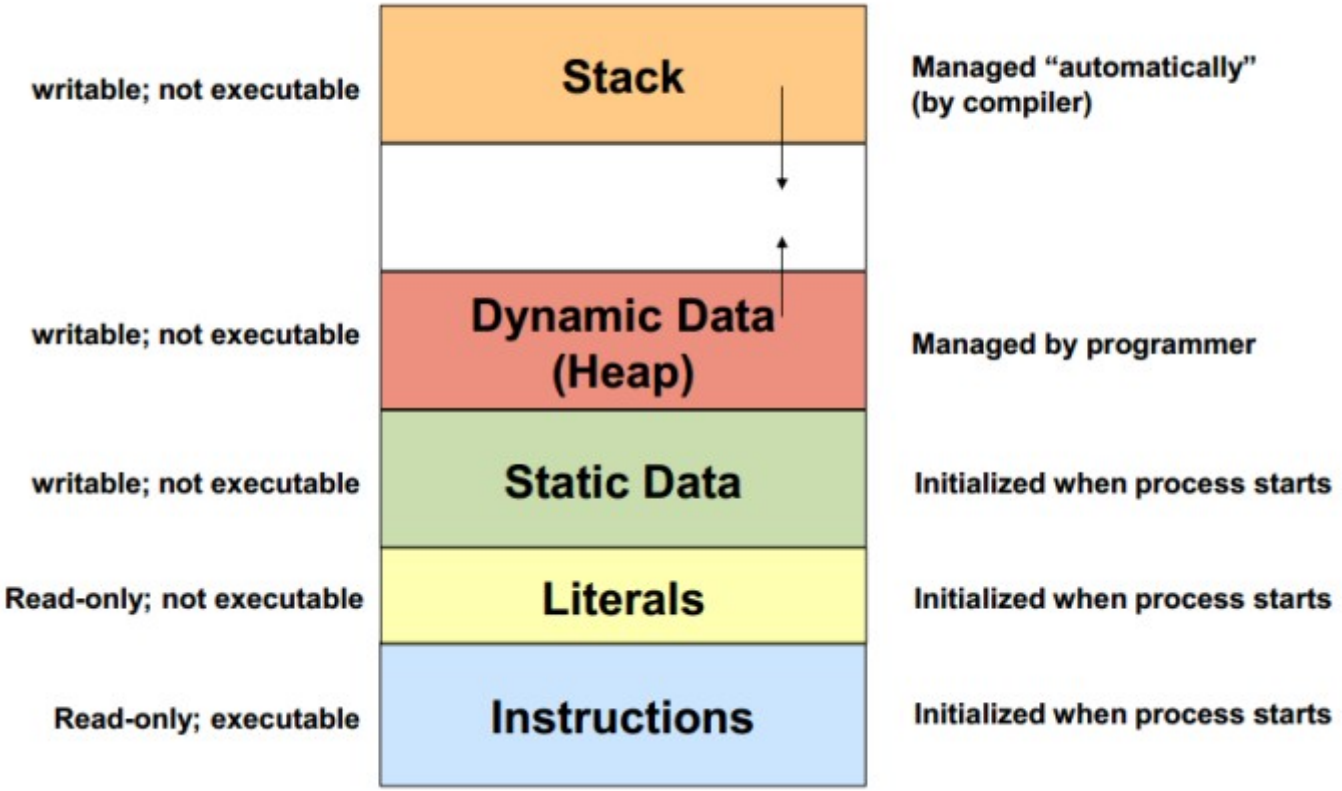
Allocazione dinamica della memoria

Corso di Elementi di Programmazione

Salvatore Venticinque

Università
degli Studi
della Campania
Luigi Vanvitelli

Allocazione delle memoria di un Processo



Variabili statiche

La dichiarazione di variabili corrispondenti a tipi semplici e struttrati ….

…. non possono mutare le loro caratteristiche in fase di esecuzione e pertanto sono dette statiche.

Problematiche

define N 100

[…]

int v[N]

[…]

Non conoscendone l'effettivo utilizzo a tempo di compilazione si sovradimensiona N

- Problemi
  - Spreco di memoria quando il riempimento << N
  - Limiti di utilizzo se il riempimento >N

Allocazione dinamica

- Solo il puntatore si dichiara staticamente:
  - int v *, n;
- Lo spazio per gli elementi si alloca quando serve:
  - printf(“quanti elementi vuoi inserire?”);
  - scanf(“%d”,&n);
  - [….]

Funzioni per l’allocazione dinamica

- Sono accessibili includendo il file header stdlib.h: **#include <stdlib.h>**
- Le principali sono quattro:
  - malloc()
  - calloc()
  - free()
  - ralloc()

malloc()

Il prototipo della funzione:

void* malloc(dim_totale);

```
int* v, n
printf(“quanti elementi vuoi inserire?”);
scanf(“%d”,&n);
v=(int*)malloc(n * sizeof(int));
```

Sia calloc() che malloc() restituiscono NULL se non riescono ad allocare la quantità di memoria richiesta.

calloc()

Il prototipo della funzione:

void* calloc(n_elementi,dim_elemento);

```
int* v, n
printf(“quanti elementi vuoi inserire?”);
scanf(“%d”,&n);
v=(int*)calloc(n , sizeof(int));
```

Sia calloc() che malloc() restituiscono NULL se non riescono ad allocare la quantità di memoria richiesta.

free()

- Il prototipo della funzione free() è:

  void **free**(void ptr);

- Dealloca lo spazio di memoria puntato da ptr
  - il cui valore proveniva da una precedente malloc() o calloc() o realloc().
- Se ptr è NULL nessuna operazione viene eseguita.

Cosa succede se uso malloc e non free ?
Ad esempio in una funzione ricorsiva?

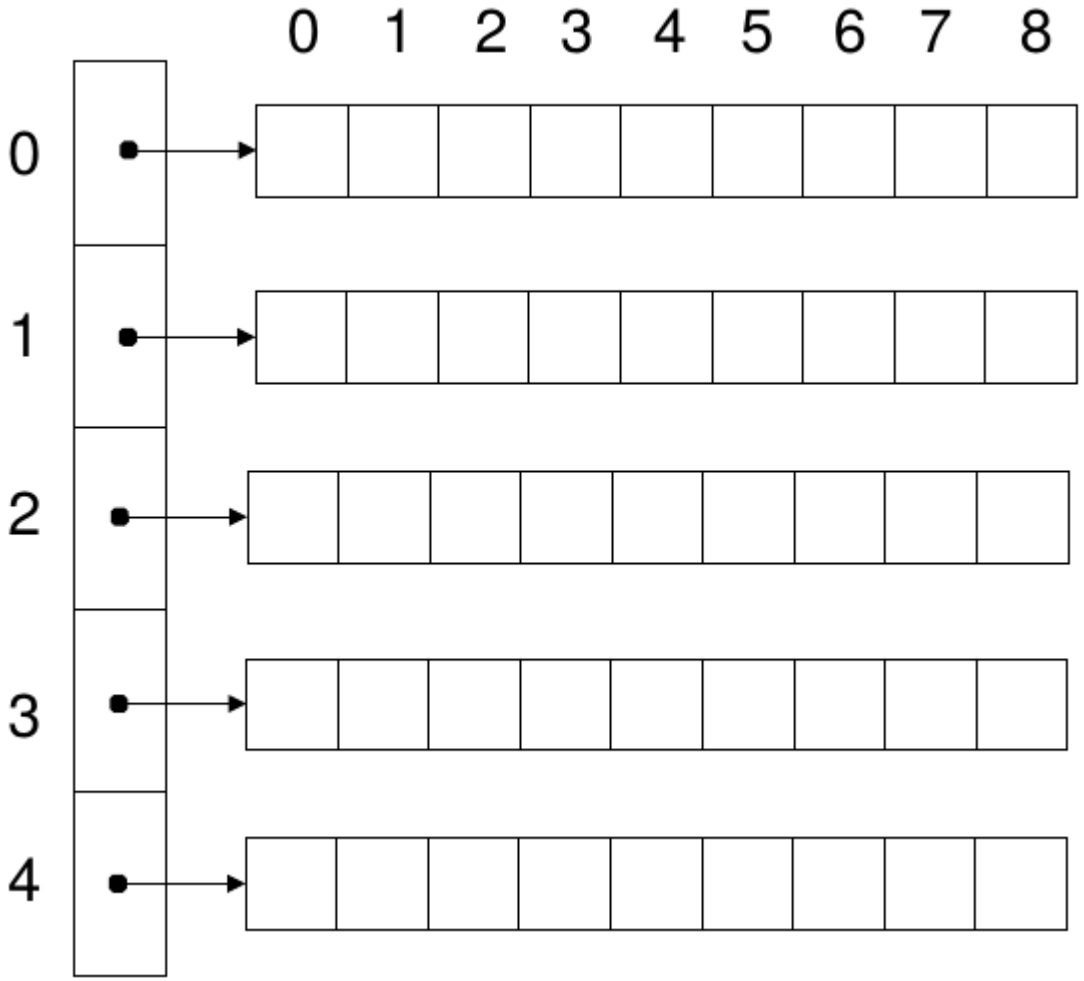
realloc()

- Il prototipo della funzione realloc() è:

  void* **realloc**(void ptr , int dim_totale);

- Modifica la dimensione di un blocco di memoria puntato da ptr a dim_totale bytes.
- Se ptr è NULL, l'invocazione è equivalente ad una malloc(dim_totale).
- Se dim_totale è 0, l'invocazione è equivalente ad una free(ptr).
- Naturalmente il valore di ptr deve provenire da una precedente invocazione di malloc() o calloc() o realloc().

Alllocare una matrice



Allocazione matrice

- La matrice deve essere vista come array di array
- Ogni riga è un array di elementi di dimensione pari al numero di colonne
- [...]

Allocazione di una riga

- int* r = malloc(sizeof(int)* c);

- Quante righe?

Array di righe

- La matrice quindi è un array di righe
  - Un array di puntatori

```
int* * m;
m= (int**) malloc(r*sizeof(int*));
```

Inizializzazione array righe

```
for(int i=0;i<r;i++)
  {
     m[i] = (int*)malloc(c*sizeof(int))
   }
```

Esercizio

- Realizzare una lista linkata di struct
  - Ogni elemento contiene il puntatore all'elemento successivo
  - riconoscerò la fine se next_ele e NULL