Tipo astratto di dato

Un tipo astratto di dato è costituito da un insieme di elementi e da una collezione di operazioni eseguibili sugli elementi dell'insieme

Strutture lineari

- Lista
  - Lista unidirezionale
  - Pila
  - Coda
  - Lista bidirezionale

Strutture non lineari

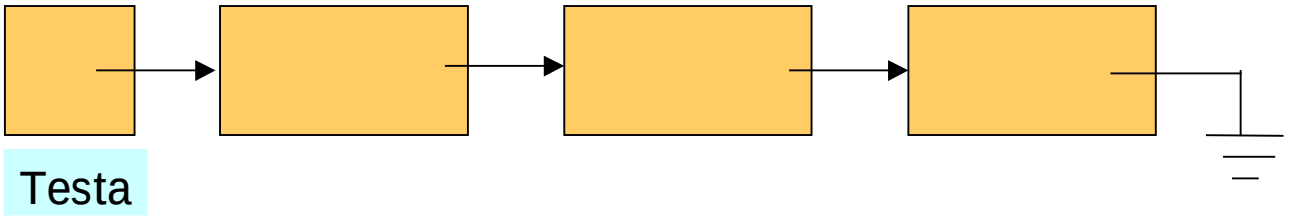
- Grafo
- Albero

Strutture lineari

La lista

Definizione del tipo astratto

- Struttura lineare di elementi omogenei
- Operazioni: *Inserimento*
  Cancellazione
  Ricerca

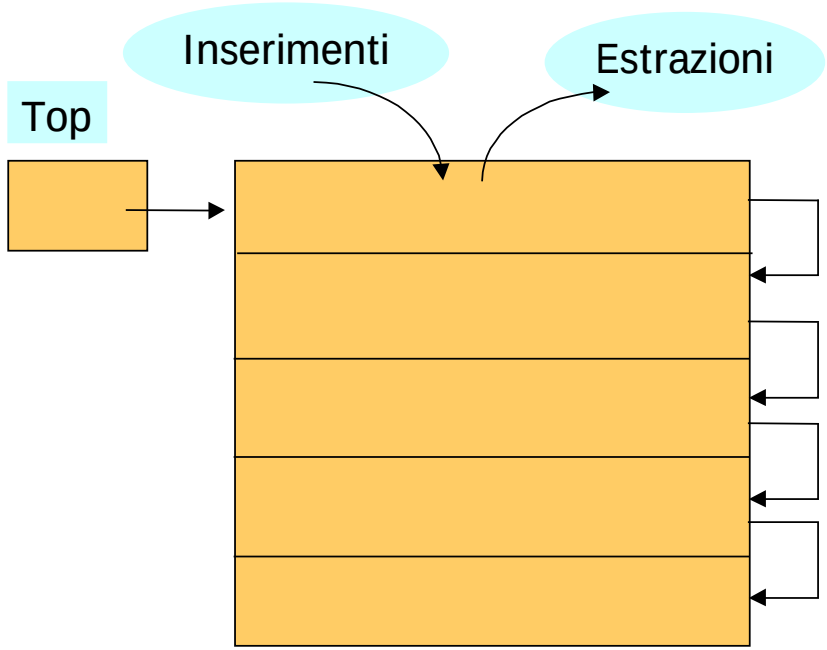


Strutture lineari

La pila (stack)

Definizione del tipo astratto

- Struttura costituita da un numero variabile di elementi omogenei con la proprietà che in qualunque momento sia possibile estrarre soltanto l'ultimo elemento inserito: *Accesso LIFO (Last In First Out)*
- Operazioni: Inserimento (*Push*)
  Estrazione (*Pop*)
  Lettura del contenuto del top (*Top*)

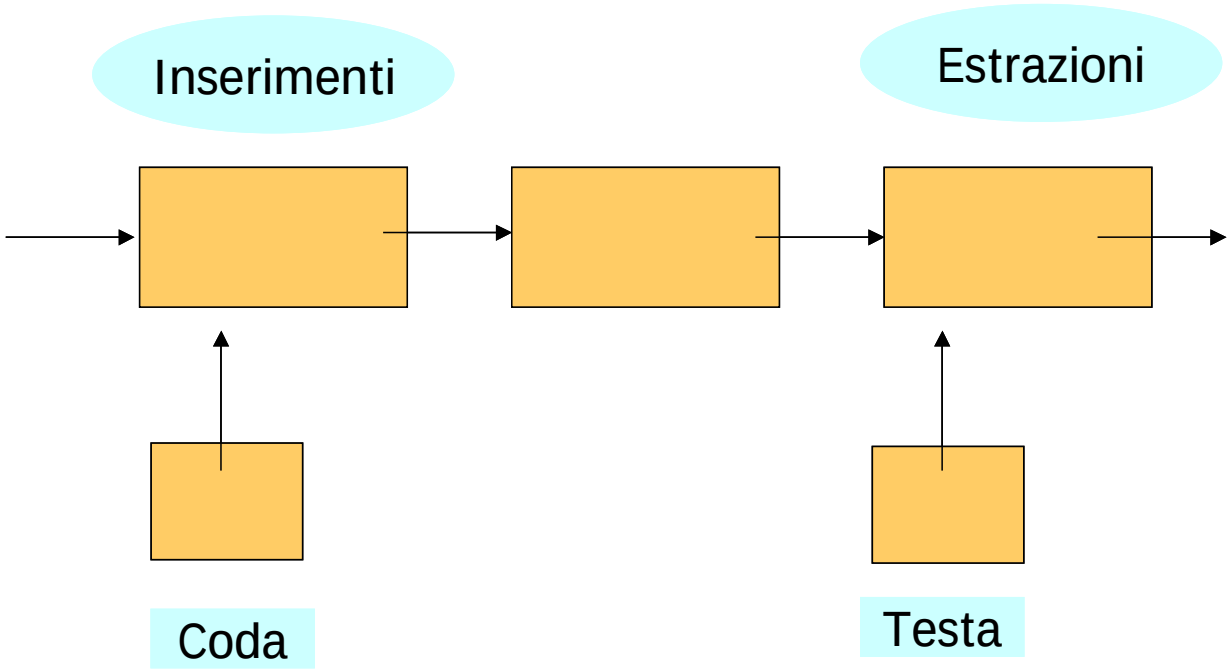


Strutture lineari

La coda

Definizione del tipo astratto

- Struttura costituita da un numero variabile di elementi omogenei con la proprietà che il primo elemento inserito sarà il primo ad essere estratto: *Accesso FIFO (First In First Out)*
- Operazioni: Inserimento
  Estrazione



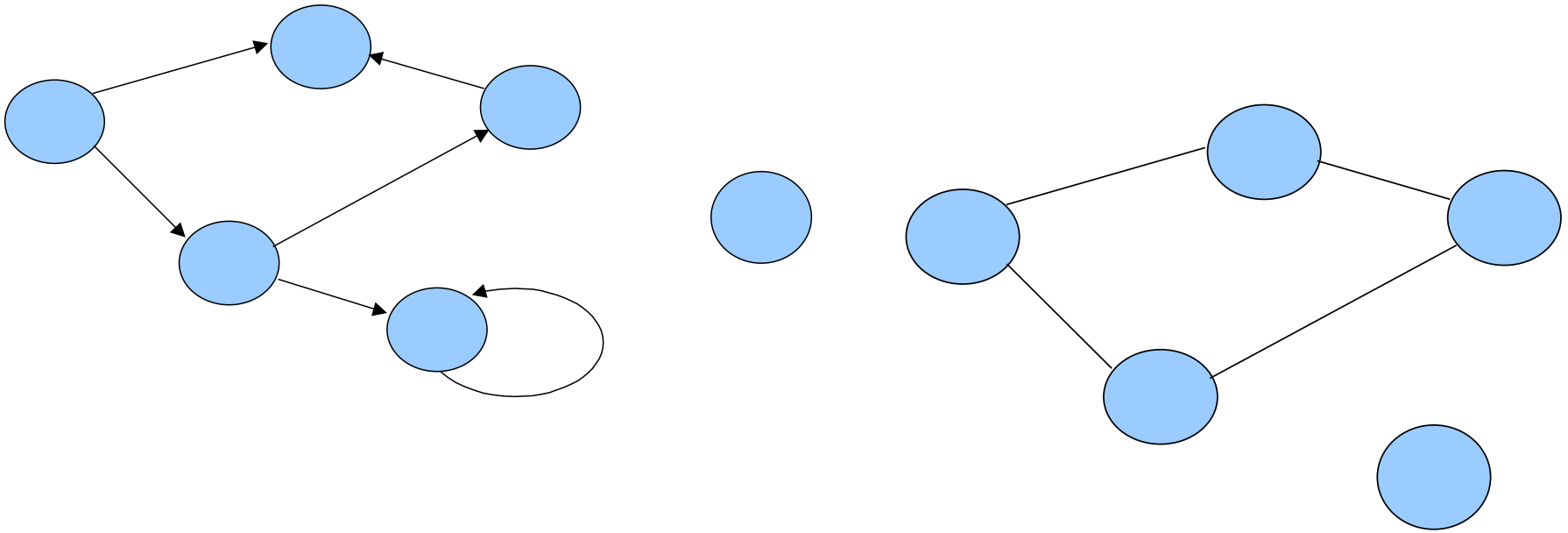
Strutture non lineari

Il grafo

Definizione.
Un grafo G=<V,A> consiste in un insieme V finito di elementi detti nodi o vertici e da un insieme A di archi che collegano coppie di nodi.

Caratterizzazione:

- Grafo orientato
- Grafo non orientato



Strutture non lineari

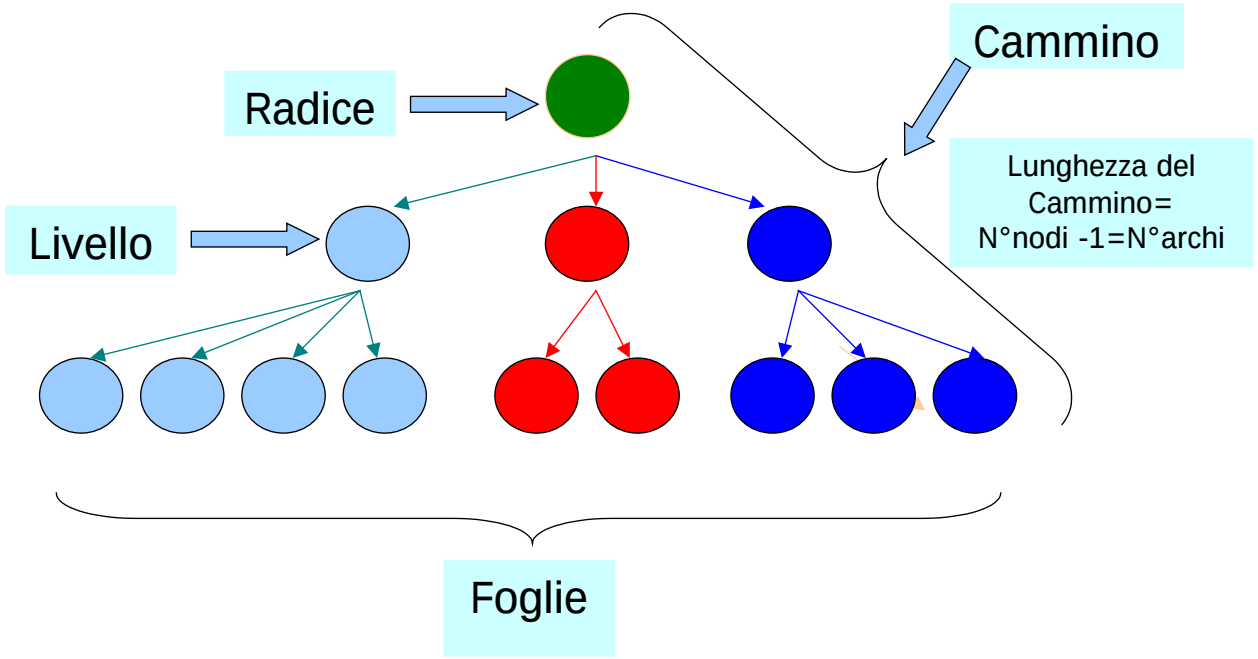
L'albero

Definizione.

Dato un insieme prefissato E di elementi. Un albero:

- può essere vuoto
- consistere di un solo elemento $e \in E$ detto nodo
- consistere di un nodo $e \in E$ collegato mediante archi diretti a un numero finito di altri alberi

- il livello e la profondità (o altezza)
- il grado di ingresso e di uscita
- il grado massimo di uscita
- il bilanciamento



Strutture non lineari

L'albero

Operazioni

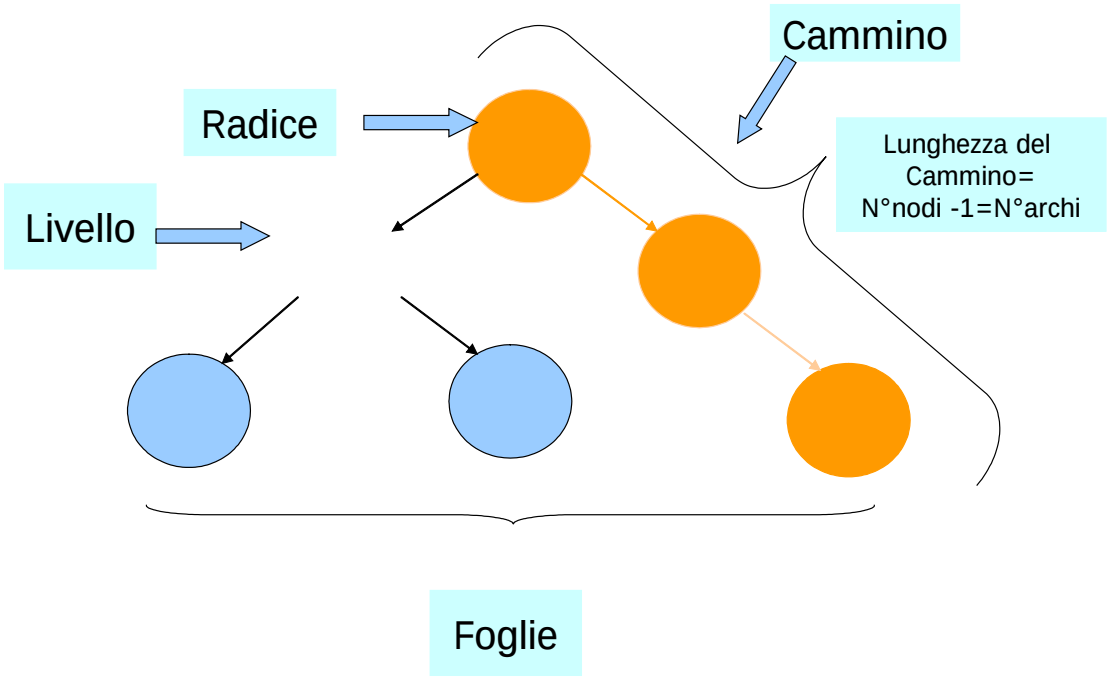
- Inserimento
- Ricerca
- Visita (Rappresentazione lineare o linearizzazione) (cfr. appunti "Linearizzazione alberi")
- Cancellazione
- Bilanciamento

Strutture non lineari

L'albero binario

Operazioni

- Inserimento
- Ricerca
- Visita (Rappresentazione lineare o linearizzazione) (cfr. appunti "Linearizzazione alberi")
  - ☞ Postordine
  - ☞ Preordine
  - ☞ Inordine
- Cancellazione
- Bilanciamento



Realizzazione di strutture lineari

Allocazione statica della memoria

La pila e la coda possono essere rappresentate nella memoria di un calcolatore utilizzando un array.

Svantaggi derivanti dall'uso di strutture realizzate mediante array:

- E' necessario definire a priori la cardinalità delle informazioni trattate
- Non è possibile gestire una quantità variabile di dati in funzione di esigenze note solo durante l'esecuzione del programma, variabili durante l'esecuzione

Realizzazione di strutture lineari

Allocazione dinamica della memoria

- Gestione dinamica della memoria per memorizzare una quantità variabile di dati in funzione di esigenze note solo durante l'esecuzione del programma e modificabili durante l'esecuzione

Caratteristiche dell'allocazione dinamica

L'allocazione dinamica consente di modificare la struttura dati in fase di esecuzione, permettendo di:

- Aggiungere un nuovo elemento nell'area dati di un programma in fase di esecuzione
- Eliminare l'elemento di memorizzazione in fase di cancellazione del dato stesso

Riferimento a dati gestiti dinamicamente

Il riferimento ai nuovi elementi di memorizzazione non può avvenire mediante identificatori

Soluzione

- Uso di puntatori
- Creazione di un nuovo elemento e restituzione di un riferimento al dato stesso

Allocazione di memoria

MALLOC

La funzione *malloc* consente l'allocazione della memoria necessaria a contenere una variabile del tipo specificato.

Sintassi della malloc:

puntatore=malloc(sizeof (Tipo del dato));

Restituisce nel puntatore l'indirizzo di memoria dell'area creata in memoria.

Rilascio o deallocazione di memoria

FREE

La funzione *free* consente la deallocazione della memoria che è stata allocata mediante la funzione malloc

Sintassi della free:

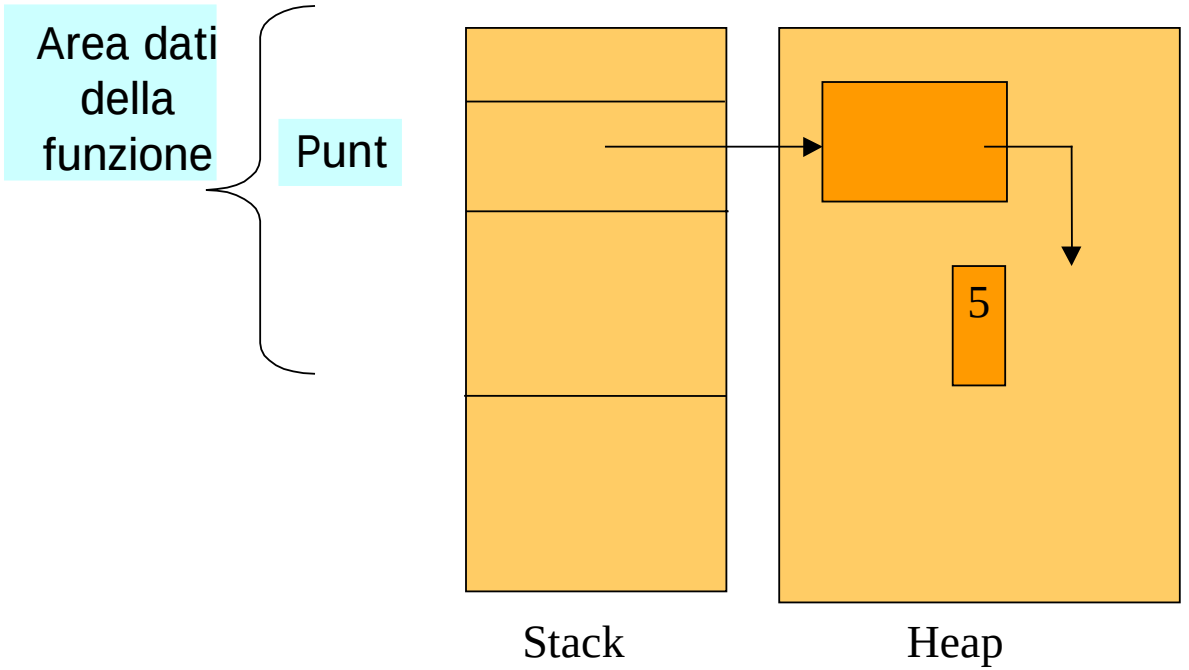
free(puntatore);

Elimina dal puntatore l'indirizzo di memoria dell'area precedentemente creata rendendola disponibile per altre operazioni di malloc

Gestione della memoria della macchina astratta di un programma

Durante l'esecuzione dei programmi la memoria della macchina astratta dei programmi viene partizionata in due aree disgiunte:

- *Stack* contiene le aree dati corrispondenti alle variabili dichiarate nelle funzioni a livello globale e locale mediante i record di attivazione
- *Heap* contiene le variabili create dinamicamente



Inconvenienti derivanti da una errata gestione dell'allocazione

Garbage production

La memoria allocata per una variabile puntata risulta inaccessibile

Dangling references

Crea riferimenti ad aree di memoria non più esistenti