UNIVERSITÀ
DI TRENTO

Dipartimento di Ingegneria e
Scienza dell'Informazione
DISI - Trento

Programmazione 1

20 - Esercitazione

Stefano Berlato
stefano.berlato-1@unitn.it

Attenzione

La presente esercitazione verrà trasmessa via Zoom. Essa verrà anche registrata e successivamente messa a disposizione degli studenti dell'Università degli Studi di Trento. Per gli utenti connessi attraverso Zoom, in caso non desideriate per qualunque motivo essere registrati, siete pregati di effettuare la disconnessione ora. La lezione sarà comunque visionabile in modo asincrono.

Anno Accademico 2021/2022

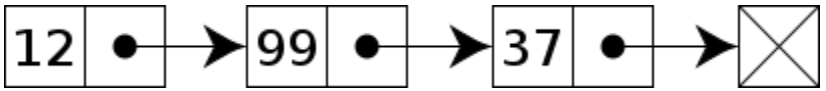
Nelle puntate precedenti

liste concatenate

Una lista concatenata `nodo` di oggetti di tipo `T` è una `struct` contenente:

1. un campo `value` di tipo `T`
2. un campo `next` di tipo `*nodo`

```
struct nodo {
  T value;
  nodo *next;
};
```



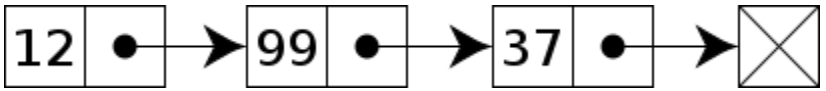
Nelle puntate precedenti

liste concatenate

Una lista concatenata `nodo` di oggetti di tipo `T` è una `struct` contenente:

1. un campo `value` di tipo `T`
2. un campo `next` di tipo `lista` (alias per `*nodo`)

```
struct nodo;
typedef nodo* lista;
struct nodo {
  T value;
  lista next;
};
```



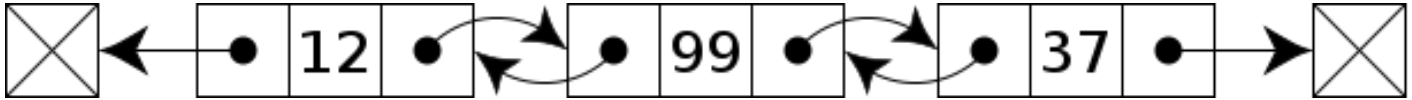
Nelle puntate precedenti

pros&cons

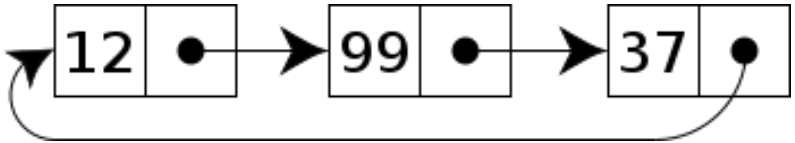
- inserimento e rimozione più efficiente rispetto ad array
- liste concatenate hanno dimensione realmente dinamica
- ricerca meno efficiente rispetto ad array
- usano più memoria rispetto ad array (puntatori)
- accesso solo sequenziale
- allocazione non contigua (efficienza cache CPU ridotta)
- ...

Nelle puntate precedenti

varianti



lista concatenata doppia



lista circolare

Nelle puntate precedenti

utilizzo

Utilizzare per implementare strutture dati astratte quali

- pile
- code
- alberi
- ...

00 - TDA

Definizione

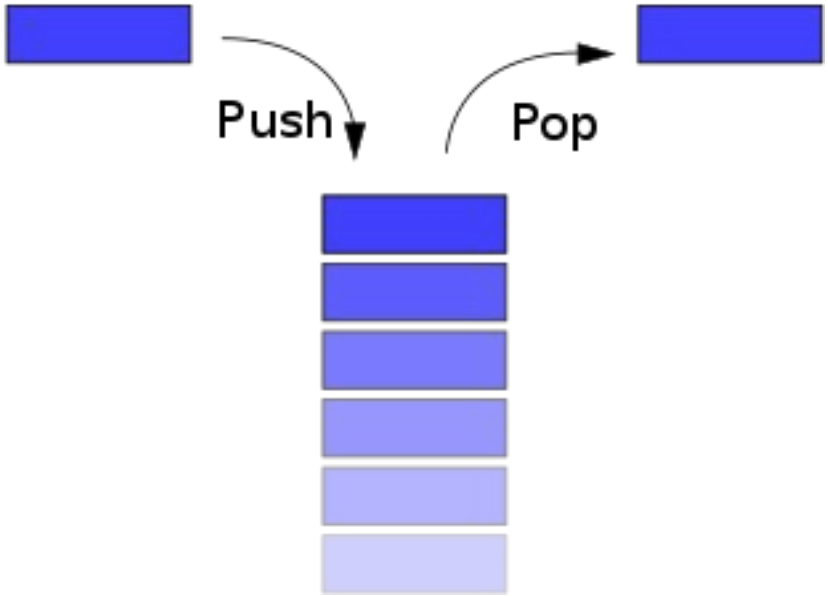
Un insieme di valori (variabili) e di operazioni (funzioni) definite indipendentemente dalla loro implementazione

- Pile (Stack)
- Code (Queue)
- Alberi (Tree)
- ...

00 - TDA

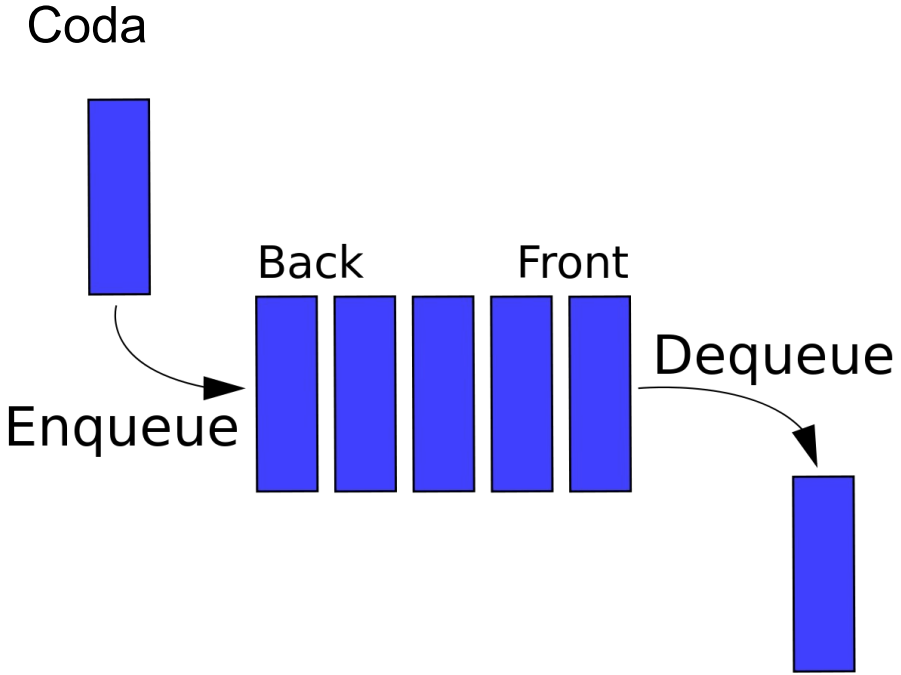
Pila

- `init()`
- `deinit()`
- `push(T)`
- `pop()`
- `top(&T)`



00 - TDA

- `init()`
- `deinit()`
- `enqueue(T)`
- `dequeue()`
- `first(&T)`

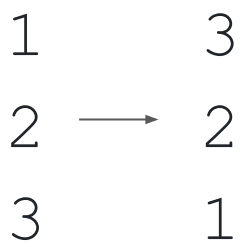


01 - Inverti Righe

Partendo dall'implementazione di una Pila tramite liste concatenate
https://pastebin.com/sCmY8xYG (.cc), https://pastebin.com/Tgtvik1C (.h)

scrivere un programma che prenda in ingresso un file con un intero per riga e lo copi su un altro file con le righe in ordine inverso

Potete usare la libreria `<fstream>`



02 - Controllo Formula

Partendo dall'implementazione di una Pila tramite liste concatenate
https://pastebin.com/sCmY8xYG (.cc), https://pastebin.com/Tgtvik1C (.h)

scrivere un programma che legge una sequenza di caratteri in input dall'utente (max 100 caratteri) e determina se le parentesi sono "bilanciate":

```
( ( ( ) ( ) ) )   =>   SI

( ) ( ) ( ( )     =>   NO
```

03 - Alle Poste

Partendo dall'implementazione di una Coda tramite liste concatenate
https://pastebin.com/yCpJTCeT (.cc), https://pastebin.com/D7XRZMw2 (.h)

Scrivere un programma che simuli l'arrivo e lo smaltimento di una coda all'ufficio postale. Gli elementi della coda sono i nomi delle persone (max 30 caratteri). Il main sarà un menù con tre opzioni: (1) aggiungi persona, (2) smaltisci persona, (3) stampa situazione corrente

Potete usare la funzione `strcmp` della libreria `<cstring>`

Per maggiori informazioni:
http://www.cplusplus.com/reference/cstring/strcmp/