



Laboratorio di Algoritmi e Strutture Dati

Aniello Murano
<http://people.na.infn.it/~murano/>

Murano Aniello - Lab. di ASD
Nona Lezione

1



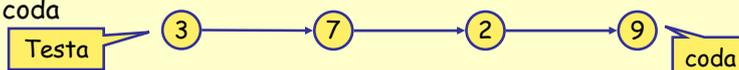
Operazioni su Liste Doppie e Circolari

Murano Aniello - Lab. di ASD
Nona Lezione

2

Indice

- **Liste puntate semplici:** Gli elementi sono logicamente organizzati in modo sequenziale e si possono scorrere in un unico verso. La lista ha un primo elemento chiamato testa e un ultimo elemento chiamato coda



- **Liste doppiamente puntate:** Sono simili alle liste puntate semplici, ma permettono di scorrere gli elementi in entrambi i versi



- **Liste puntate semplici circolari:** Sono liste puntate semplici senza testa ne coda.



- **Liste doppiamente puntate circolari:** Liste doppiamente puntate senza testa ne coda.

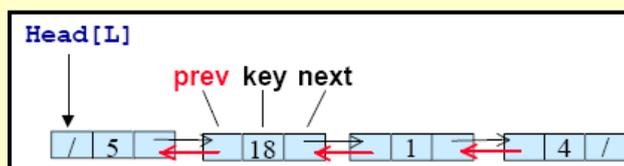


Murano Aniello - Lab. di ASD
Nona Lezione

3

Liste puntate doppie

- Una Lista Doppia Puntata è un insieme dinamico in cui ogni elemento ha uno o più campi contenenti informazioni e due riferimenti, uno all'elemento successivo (next) della lista ed uno all'elemento precedente (prev) della lista.



Murano Aniello - Lab. di ASD
Nona Lezione

4

Implementazione in C

- Per definire la struttura di un elemento di una lista doppiamente puntata bisogna utilizzare due puntatori alla stessa struttura. Il primo punterà all'elemento precedente mentre il secondo punterà all'elemento successivo:

```
struct elemento {
    struct elemento *prev;
    int inf;
    struct elemento *next;};
```

- Per l'inizializzazione di una lista doppiamente puntata si può considerare il codice visto per l'inizializzazione delle liste singolarmente puntate, introducendo opportunamente il codice per il puntatore all'elemento precedente.



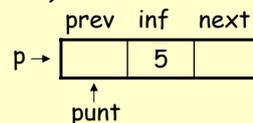
Murano Aniello - Lab. di ASD
Nona Lezione

5

Funzione crea_lista() 1/2

- La funzione `crea_lista()` crea due puntatori ad elemento, uno di nome `p` (puntatore al primo elemento della lista) e l'altro di nome `punt` (puntatore che permette di scorrere la lista);

```
struct el *crea_lista() {
    struct el *p, *punt;
    int i, n;
    printf("\n Specificare il numero di elementi... ");
    scanf("%d", &n);
    if(n==0)
        p = NULL;
    else {
        /* creazione primo elemento */
        p = (struct el *)malloc(sizeof(struct el));
        printf("\nInserisci il primo valore: ");
        scanf("%d", &p->inf);
        punt = p; p->prev=NULL;
```



Murano Aniello - Lab. di ASD
Nona Lezione

6

Funzione crea_lista() 2/2

```
for(i=2; i<=n; i++)
{
    punt->next = (struct el *)malloc(sizeof(struct el));
    → punt->next->prev=punt;
    punt = punt->next;
    printf("\nInserisci il %d elemento: ", i);
    scanf("%d", &punt->inf);
} // chiudo il for
punt->next = NULL; // marcatore fine lista
} // chiudo l'if-else
return(p);
} // chiudo la funzione
```



Murano Aniello - Lab. di ASD
Nona Lezione

7

Inserimento in coda: main()

```
#include <stdio.h>
#include <malloc.h>
struct el {struct el *prev; int inf; struct el *next;};
struct el *crealista();
int inserisci_in_coda(struct el*,int);

int main() {
    struct el *lista;
    int valore;
    lista=crealista(); stampalista();
    printf("\nInserisci elemento da inserire: ");
    scanf("%d", &valore);
    inserisci_in_coda(lista,valore);
}
```



Murano Aniello - Lab. di ASD
Nona Lezione

8

Inserimento in coda: funzione

```
void inserisci(struct el *p, int valore)
{ struct el *nuovo;
  if (p==NULL) {
    p=(struct el *)malloc(sizeof(struct el));
    p->inf=valore; p->prev=NULL; }
  else {
    while (p->next!= NULL) p=p->next;
    nuovo=(struct el *)malloc(sizeof(struct el));
    nuovo->prev=p;
    nuovo->inf=valore;
    nuovo->next = NULL; }
  p->next=nuovo;
  return;
}
```

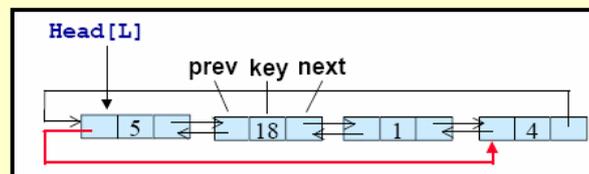


Murano Aniello - Lab. di ASD
Nona Lezione

9

Liste puntate circolari

- Una Lista Circolare puntata è un una lista puntata in cui il puntatore next dell'ultimo elemento della lista punta all'elemento in testa alla lista.
- Infine, se la lista è doppiamente puntata, il puntatore prev della testa della lista punta all'elemento in coda alla lista



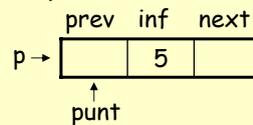
Murano Aniello - Lab. di ASD
Nona Lezione

10

Funzione crea_lista() 1/2

- La funzione `crea_lista()` crea due puntatori ad elemento, uno di nome `p` (puntatore al primo elemento della lista) e l'altro di nome `punt` (puntatore che permette di scorrere la lista);

```
struct el *crea_lista() {
    struct el *p, *punt;
    int i, n;
    printf("\n Specificare il numero di elementi... ");
    scanf("%d", &n);
    if(n==0)
        p = NULL;
    else {
        /* creazione primo elemento */
        p = (struct el *)malloc(sizeof(struct el));
        printf("\nInserisci il primo valore: ");
        scanf("%d", &p->inf);
        punt = p; p->prev=NULL;
    }
}
```



Murano Aniello - Lab. di ASD
Nona Lezione

11

Funzione crea_lista() 2/2

```
for(i=2; i<=n; i++)
{
    punt->next = (struct el *)malloc(sizeof(struct el));
    punt->next->prev=punt;
    punt = punt->next;
    printf("\nInserisci il %d elemento: ", i);
    scanf("%d", &punt->inf);
} // chiudo il for
punt->next = p; p->prev = punt; // circolarit  della lista
} // chiudo l'if-else
return(p);
} // chiudo la funzione
```



Murano Aniello - Lab. di ASD
Nona Lezione

12

Elimina elemento dalla lista

```
struct elemento *togli(struct elemento *p, int valore, int *posizione)
{ struct elemento *inizio=NULL;
  int i=1;
  if (p!=NULL) {
    inizio=p;
    do {
      if (p->inf==valore) {
        *posizione=i;
        if (p->next==p)
          inizio=NULL;
        else {
          p->prev->next=p->next;
          p->next->prev=p->prev; }
        free(p); p=NULL; }
      else {p=p->next; i++;} }
    while (p!=inizio && p!=NULL); }
  return inizio;
}
```

