

Liste dinamiche

Le liste sono strutture dati più potenti degli array, in quanto permettono l'**allocazione dinamica della memoria** (non c'è bisogno di conoscerne la dimensione prima della compilazione).

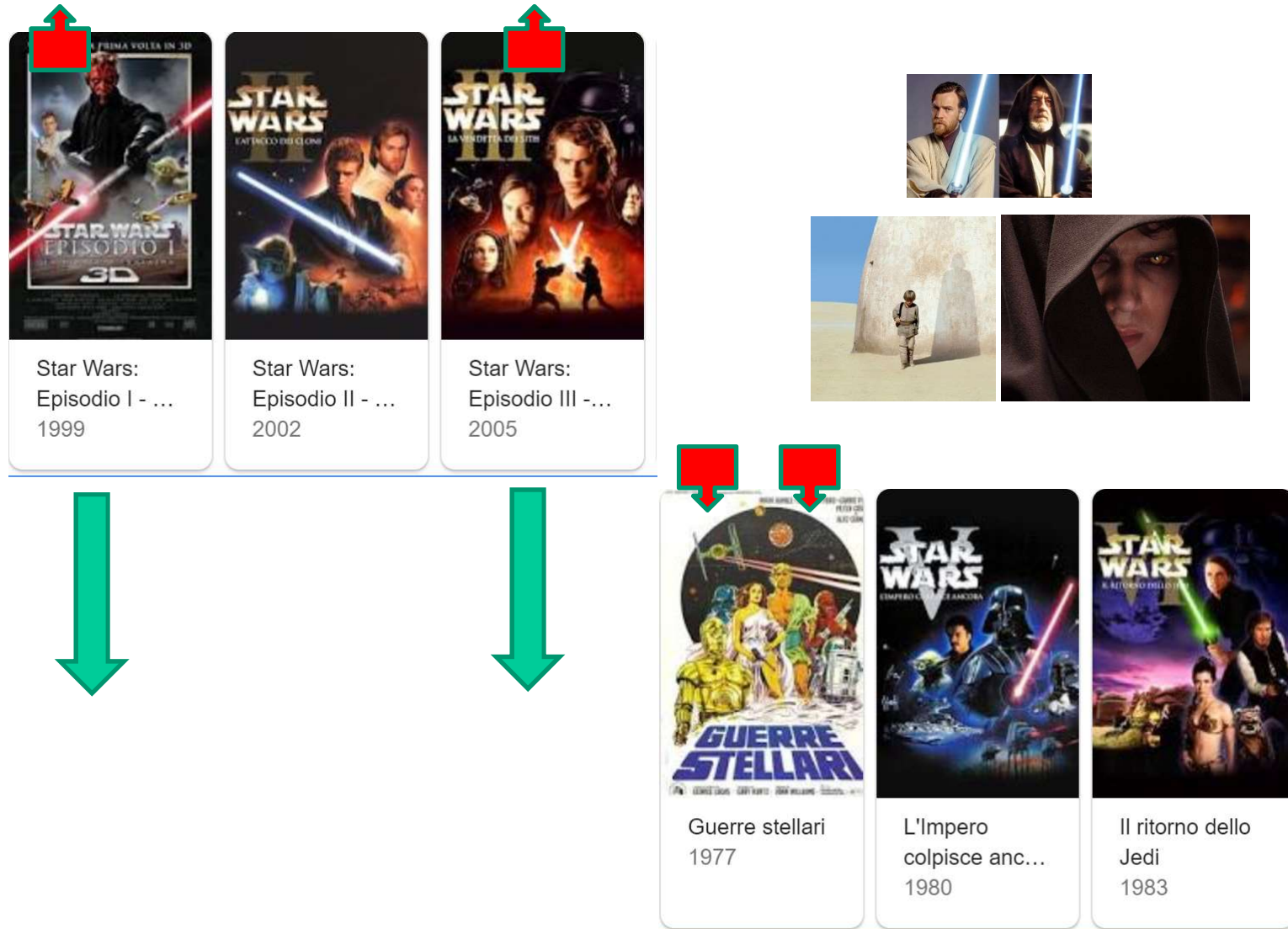
Tratteremo esempi su **liste concatenate (o semplici)**. Si tratta di **sequenze di nodi**, ciascuno contenente sia la vera e propria informazione (**info**) sia il collegamento al nodo successivo (**link**).



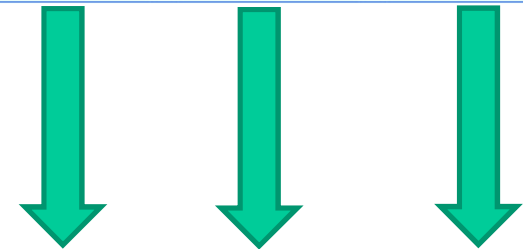
Il primo nodo della lista è la **TESTA**. Noto il riferimento alla testa, da lì posso accedere sequenzialmente a tutti gli altri nodi (pertanto, da un punto di vista realizzativo/sintattico, è possibile identificare la lista con il suo riferimento alla testa). L'ultimo è la **CODA** (il cui link è un puntatore a NULL).

La lista si dice **ordinata** quando l'ordine fisico degli elementi (la posizione occupata dai nodi nella struttura) coincide con l'ordine logico (la relazione di ordine definita sull'elemento info).

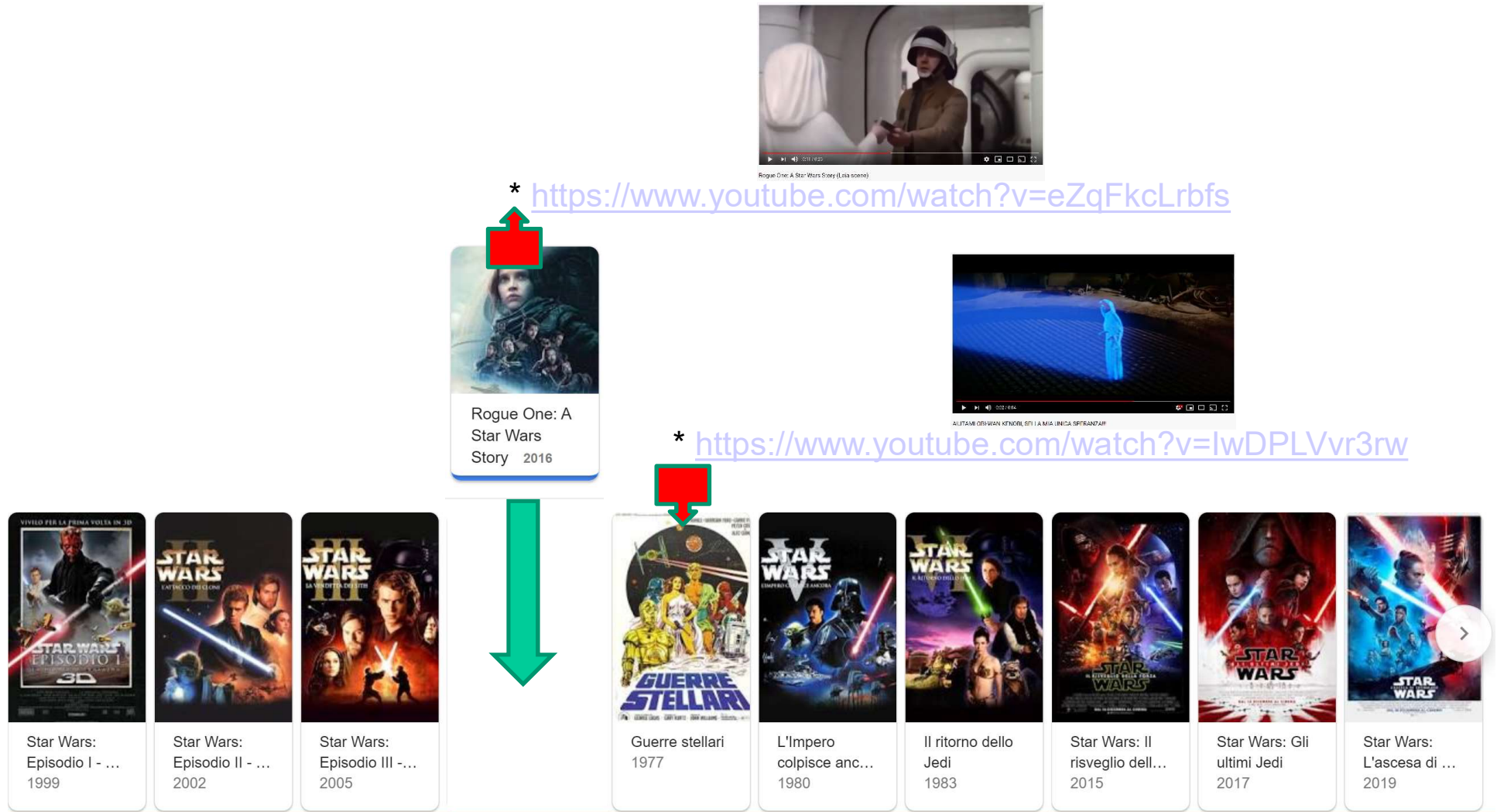
Liste dinamiche – inserimenti in testa, intermedi/in coda e puntatori di memoria (*una metafora stellare*)



Liste dinamiche – inserimenti in testa, intermedi/in coda e puntatori di memoria (*una metafora stellare*)



Liste dinamiche – inserimenti in testa, intermedi/in coda e puntatori di memoria (*una metafora stellare*)



Liste dinamiche – inserimenti in testa, intermedi/in coda e puntatori di memoria (*una metafora stellare*)



Star Wars:
Episodio I - ...
1999



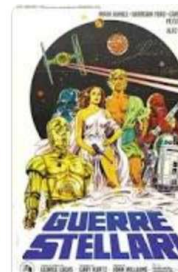
Star Wars:
Episodio II - ...
2002



Star Wars:
Episodio III - ...
2005



Rogue One: A
Star Wars
Story 2016



Guerre stellari
1977



L'Impero
colpisce anc...
1980



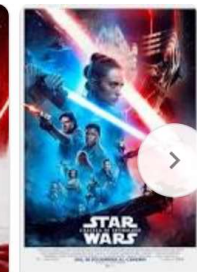
Il ritorno dello
Jedi
1983



Star Wars: Il
risveglio dell...
2015

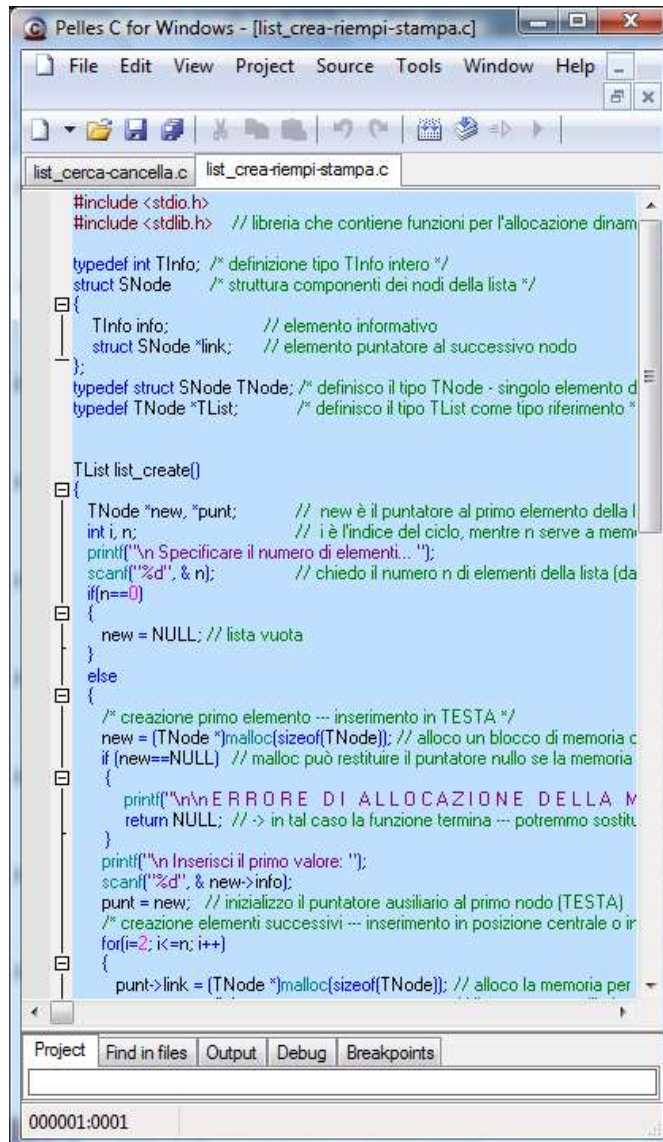


Star Wars: Gli
ultimi Jedi
2017



Star Wars:
L'ascesa di ...
2019

Liste dinamiche



```
#include <stdio.h>
#include <stdlib.h> // libreria che contiene funzioni per l'allocazione dinamica

typedef int TInfo; /* definizione tipo TInfo intero */
struct SNode /* struttura componenti dei nodi della lista */
{
    TInfo info; // elemento informativo
    struct SNode *link; // elemento puntatore al successivo nodo
};
typedef struct SNode TNode; /* definisco il tipo TNode - singolo elemento della lista */
typedef TNode *TList; /* definisco il tipo TList come tipo riferimento */

TList list_create()
{
    TNode *new, *punt; // new è il puntatore al primo elemento della lista
    int i, n; // i è l'indice del ciclo, mentre n serve a memorizzare il numero di elementi
    printf("\n Specificare il numero di elementi... ");
    scanf("%d", &n); // chiedo il numero n di elementi della lista (da 1 a 10)
    if(n==0)
    {
        new = NULL; // lista vuota
    }
    else
    {
        /* creazione primo elemento --- inserimento in TESTA */
        new = (TNode *)malloc(sizeof(TNode)); // alloco un blocco di memoria
        if (new==NULL) // malloc può restituire il puntatore nullo se la memoria è esaurita
        {
            printf("\n\nERRORE DI ALLOCAZIONE DELLA MEMORIA\n");
            return NULL; // -> in tal caso la funzione termina --- potremmo sostituire con un valore di errore
        }
        printf("\n Inserisci il primo valore: ");
        scanf("%d", &new->info);
        punt = new; // inizializzo il puntatore ausiliario al primo nodo (TESTA)
        /* creazione elementi successivi --- inserimento in posizione centrale o in coda */
        for(i=2; i<=n; i++)
        {
            punt->link = (TNode *)malloc(sizeof(TNode)); // alloco la memoria per il nuovo nodo
            if (punt->link==NULL)
            {
                printf("\n\nERRORE DI ALLOCAZIONE DELLA MEMORIA\n");
                return NULL;
            }
            scanf("%d", &punt->link->info);
            punt = punt->link;
        }
        punt->link = NULL;
    }
    return new;
}
```

LIST_CREARIEMPISTAMPA.c dimostra come creare una lista, riempirla di numeri interi (con input da tastiera), e stamparla a video (utilizzando una funzione iterativa o una ricorsiva).

LIST_CERCA-CANCELLA.c lavora su una lista ordinata di interi (definita nel codice) ed implementa gli algoritmi, sia iterativi che ricorsivi (ottimali), per cercare un nodo nella lista e per cancellare un nodo nella lista.