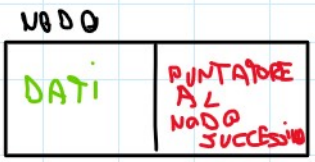
Liste

venerdì 6 ottobre 2023 14:13

È UNA STRUTTURA DATI DINAMICA. VARIA LA SUA DIMENSIONE DURANTE L'ESECUZIONE DEL CODICE. (L'ARRAY NO)

NON È INDICIZZATA (ARRAY[2] È INDICIZZATO)

È SUDDIVISA IN NODI:



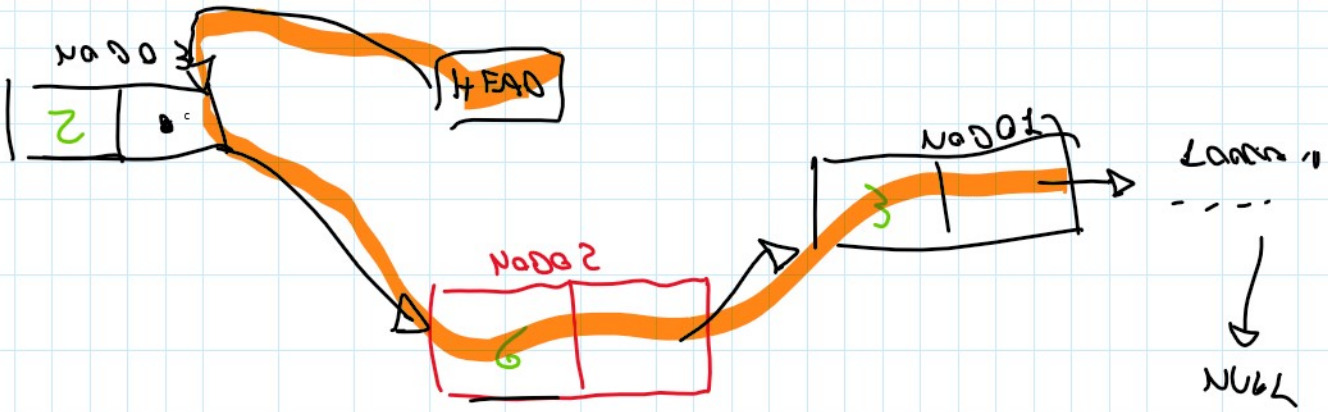
(LISTA CONCATENATA SEMPLICE)

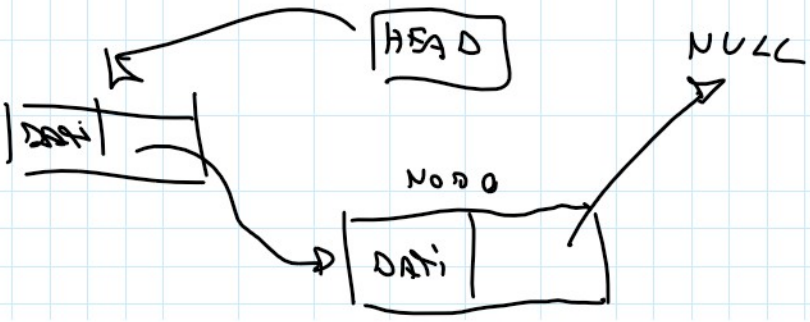
È SEMPRE NECESSARIO UN PUNTATORE ALLA TESTA DELLA LISTA.

INSERIMENTO IN TESTA: L'ELEMENTO VA INSERITO COME PRIMO ELEMENTO DELLA LISTA

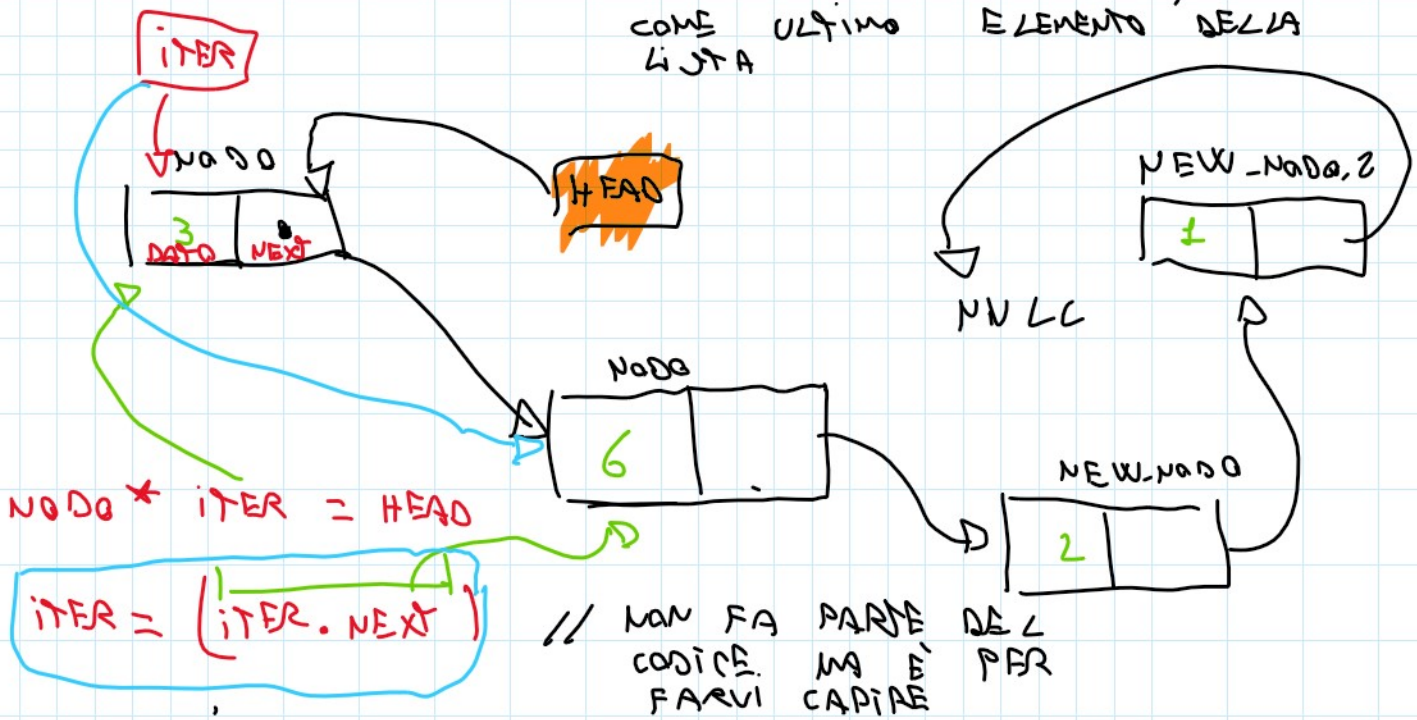
LISTA VUOTA: NODO * HEAD = NULL

```
NODO * NEW_NODO = MALLOC (NEW
NEW_NODO.DATI = DATI;
NEW_NODO.NEXT = HEAD;
HEAD = NEW_NODO;
```





INSERIMENTO IN CODA : INSERIRE UN NUOVO ELEMENTO COME ULTIMO ELEMENTO DELLA LISTA



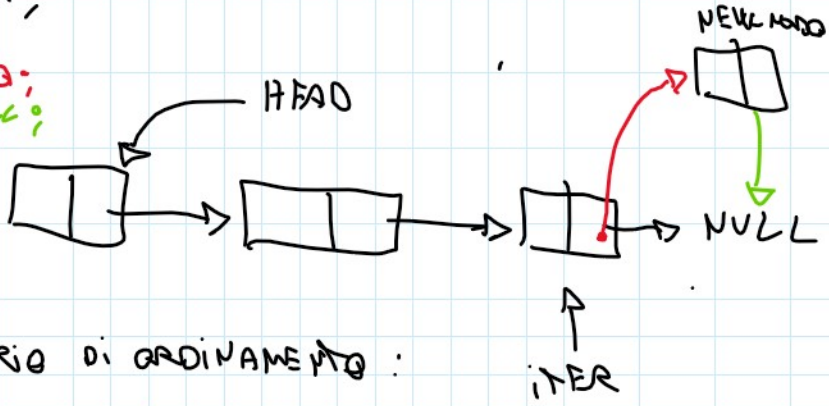
NODO * ITER = HEAD

ITER = (ITER.NEXT)

// NON FA PARTE DEL CODICE MA È PER FARVI CAPIRE

```
WHILE (ITER!=NULL&& ITER.NEXT != NULL )
        ITER = ITER.NEXT;      // OTTENGO IL PUNTATORE ALL'ULTIMO ELEMENTO
}
IF ( ITER == NULL ) {
    NODO * NEW_NODO = MALLOC() / NEW;
    NEW_NODO.DATI = DATI;
    NEW_NODO.NEXT = HEAD;
    HEAD = NEW_NODO;

ELSE {
    NODO  NEW_NODO = MALLOC() / NEW;
    NEW_NODO.DATI = DATI;
    ITER.NEXT = NEW_NODO;
    NEW_NODO.NEXT = NULL;
}
```



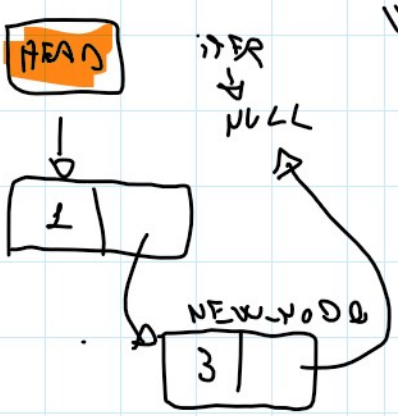
INSERIMENTO SECONDO UN CRITERIO DI ORDINAMENTO :

INSERIRE (3, 1, 2, 5, 4)

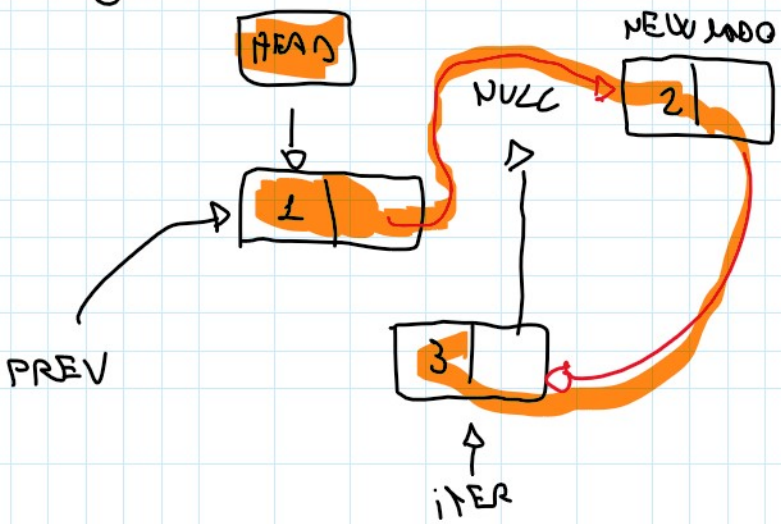
1ª


```
NODO* ITER = HEAD;
WHILE ( ITER!=NULL &&
        ITER.DATA < NEW_NODO.DATA ){
    ITER = ITER.NEXT;
}
IF(ITER == NULL){
INSERISCO NEW_NODO IN CODA;
}
```

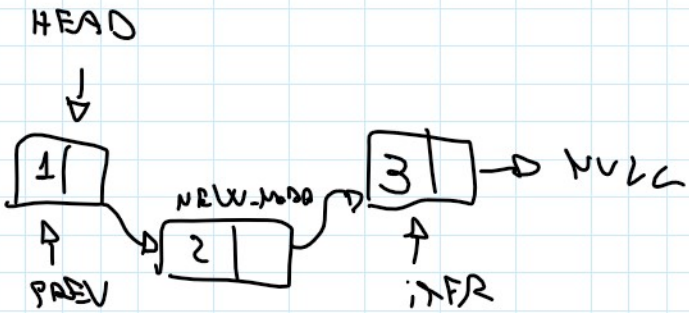
2ª



3ª

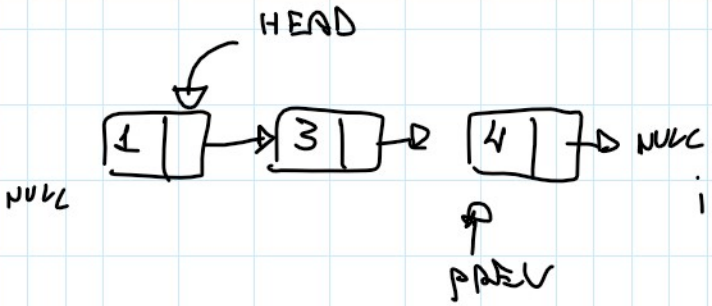



```
NODO*  PREV =NULL;
NODO*  ITER = HEAD;
WHILE( ITER!= NULL &&
       ITER.DATA < NEW_NODO.DATA
    {
       PREV = ITER;
       ITER = ITER.NEXT
    }
PREV.NEXT = NEW_NODO;
NEW_NODO.NEXT = ITER;
```



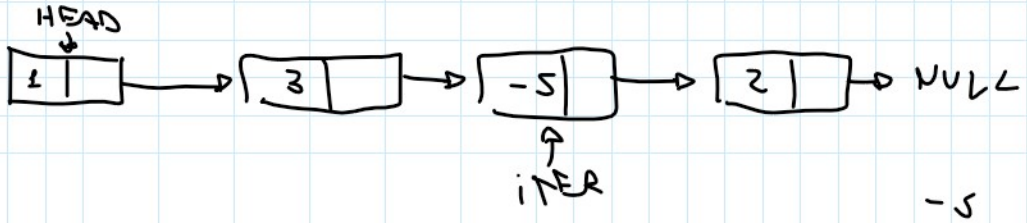

```
NODO*  PREV =NULL;
NODO*  ITER = HEAD;
```

2




```
WHILE( ITER != NULL &&
       ITER.DATA < NEW_NODO.DATA
{
    PREV = ITER;
    ITER = ITER.NEXT
}
IF( PREV == NULL)
    NEW_NODO.NEXT = HEAD;
    HEAD = NEW_NODO;
ELSE IF( ITER == NULL )
    PREV.NEXT = NEW_NODO;
    NEW_NODO.NEXT = NULL;
ELSE {
    PREV.NEXT = NEW_NODO;
    NEW_NODO.NEXT = ITER;
}
```

RICERCA IN UNA LISTA

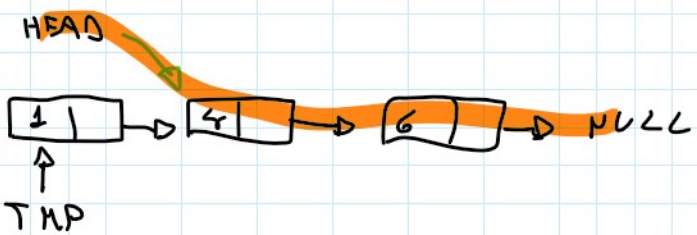



```
WHILE (ITER != NULL && ITER.DATA != VALUE){
    ITER = ITER.NEXT;
}
RETURN ITER
```

(VALUE = -5)

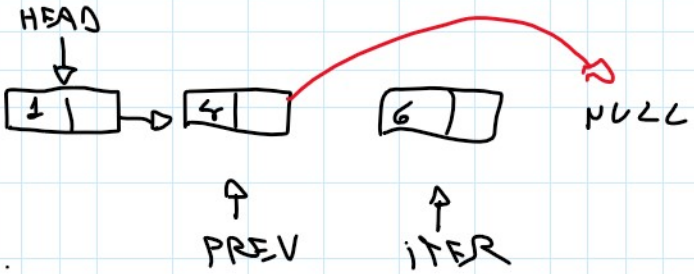
RIMOZIONE DI UN ELEMENTO

RIMOZIONE IN TESTA



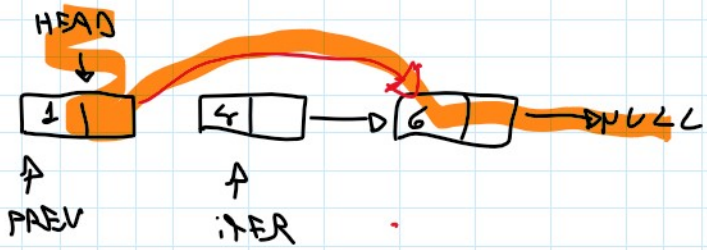

```
NODO * TMP = HEAD
HEAD = HEAD.NEXT
RETURN TMP; /DEL TMP;
FREE(TMP)
```

RIMOZIONE IN CODA




```
NODO* ITER = HEAD
NODO* PREV = NULL
WHILE( ITER!=NULL &&
     ITER.NEXT != NULL) {
     PREV = ITER;
     ITER = ITER.NEXT;
}
PREV.NEXT = NULL
RETURN ITER;
```

RIMOZIONE DEL NODO CERCATO

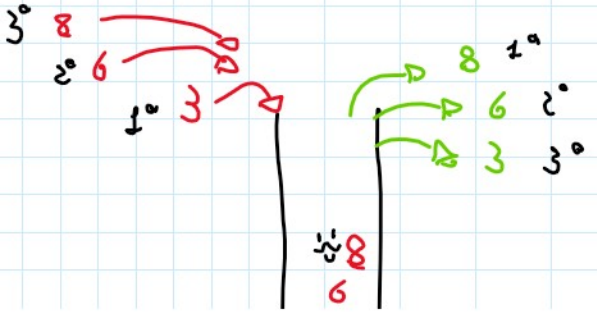


6

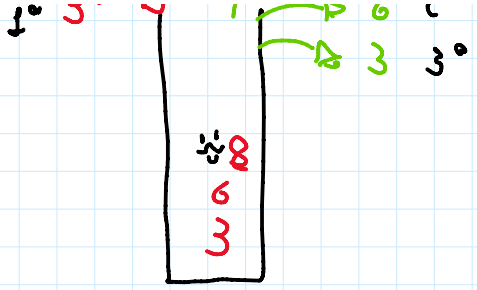
```
NODO* ITER = HEAD
NODO* PREV = NULL
WHILE( ITER!=NULL &&
     ITER.VALUE != VALUE) {
     PREV = ITER;
     ITER = ITER.NEXT;
}
IF( PREV == NULL )
   HEAD = HEAD.NEXT;
ELSE if( ITER!=NULL) {
   PREV.NEXT = ITER.NEX
RETURN ITER;
```

PILA/STACK: TIPO LIFO (LAST IN FIRST OUT)

FUNZIONA COME UN TUBETTO DI PRINGLES



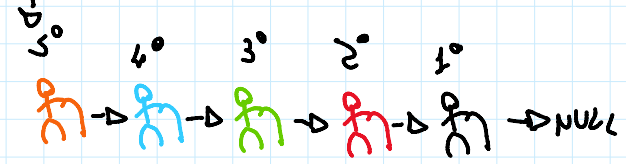
LA PILA PUÓ ESSERE IMPLEMENTATA TRAMITE L'INSERIMENTO E LA RIMOZIONE IN TESTA IN UNA LISTA CONCATENATA.
(ANCHE IN CODA MA È PIÚ BRUTTO)



RIMOZIONE IN TESTA IN UNA LISTA CONCATENATA.
(ANCHE IN CODA MA È PIÚ BRUTTO)

CODA: È TIPO FIFO (FIRST IN FIRST OUT)

FUNZIONA COME LA FILA ALLE POSTE (SENZA PRELAZIONE)



ORDINE D'INGRESSO UGUALE ALL'ORDINE D'USCITA

INSERISCO IN CODA E RIMUOVO IN TESTA

OPPURE

INSERISCO IN TESTA E RIMUOVO IN CODA