**17° lezione prog1**

**Le liste gestite dinamicamente (il numero degli elementi non è stato stabilito)**

La lista è una sequenza di elementi che  vengono allocati in punti differenti della memoria(RAM). Si deduce che gli elementi non sono condivise memoria. la lista inizialmente è vuota. In questa struttura si ha la testa o head  che è un puntatore al primo elemento della lista. Inizialmente la head punterà a NULL perché  non si ha nessun elemento nella lista. Se si inserisce un elemento questo viene creato con la funzione di  allocazione dinamica. Per il corretto inserimento di un elemento bisogna  far puntare  la testa al nuovo elemento. Ogni elemento della lista si chiama  nodo. I vari nodi saranno sparsi in memoria pertanto per legare un nodo all'altro Bisogna usare  un puntatore al successivo elemento della lista(next). Se in questa struttura viene inserito un elemento il puntatore next  punterà a NULL.” Si deduce pertanto che il nodo è formato dal  dato è un puntatore al prossimo elemento. se si inserisce un altro elemento la testa punterà all'elemento appena inserito e avrà  il puntatore Next che sarà NULL. Tutti questi nodi sono istanze di strutture. Ci potrebbero essere anche delle strutture annidate.  ogni nodo è formato da due  parti:

1. informazioni
2. metadati, ci permettono di mantenere integra la struttura (PUNTATORI next  che punta al prossimo elemento. Sì deduce pertanto che l'ultimo elemento della lista punterà a null)

Si preferisce mantenere le liste ordinate In base a un criterio di ordinamento chiamato chiave Pertanto ogni qualvolta si procede ad un inserimento di un nuovo elemento non si può intervenire in testa  né in coda Si procede con un algoritmo di ordinamento che calcola la posizione corretta in cui il nuovo elemento deve stare.

Immagine che contiene testo, linea, Carattere, schermata

Descrizione generata automaticamente**Struttura della lista**

**typedef struct dato{**

**struct dato \*value;**

**}dato;**

**typedef struct nodo{**

**struct dato data;**

**struct nodo \* next;**

**}nodo;**

**Inserimento di un nodo in testa**

La funzione di inserimento deve svolgere delle operazioni:

1. **creazione del nuovo nodo con la malloc**
2. **il puntatore del nuovo nodo punta alla testa**
3. **la testa punta al nuovo nodo**

**new\_node->data=\*ptrDato**

**new\_node->next=\*head;**

**\*head=new\_node;**

Immagine che contiene testo, diagramma, schermata, linea

Descrizione generata automaticamente

**inserimento di un nodo**

void insertNodo\_midle(struct dato\* ptrDato,  struct nodo \*\*head){

   nodo \*new\_node= malloc(sizeof(nodo));

   new\_node->data.value=20;

   new\_node->next=**NULL**;

   nodo \*prev=**NULL**;

   nodo \*curr=**NULL**;

*//2 puntatori che vanno messi a null*

while(curr!=**NULL** && ptrDato->value > curr->data.value ){

       prev=curr;   *//faccio puntare prev a curr*

curr=curr->next; *// curr punta al prossimo nodo*

}

*//in questo modo avrò prev nel nodo prima e curr nel nodo corrente*

prev->next=new\_node;  *//faccio puntare prev al nodo nodo*

new\_node->next=curr; *//il puntatore del nuovo nodo sarà curr*

}

1. creo il nuovo nodo
2. creo due puntatori prev e curr che all’inizio sono NULL
3. ciclo while() finché  curr sia diverso da null e il valore dato dalla funzione sia maggiore del dato di curr
4. dentro il ciclo assegno a prev in valore di curr
5. e faccio puntare curr al prossimo nodo
6. dopo che è finito il ciclo while()  il punatore di prev va al nuovo nodo
7. e il puntatore del nuovo nodo va a quello di curr