Data Link List structure (Circular Linked List)

The linked list (Circular Linked List) is a variant of the Linked List, in which the first element points to the last element and the last element points to the first element.

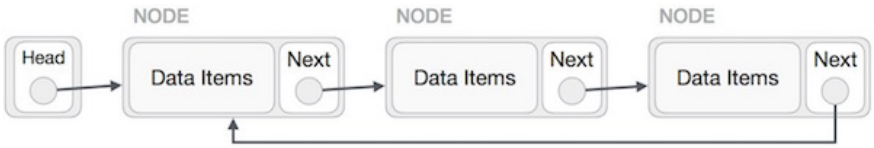
What is a round list (Circular Linked List)?

The linked list (Circular Linked List) is a variant of the Linked List, in which the first element points to the last element and the last element points to the first element.

Both types of Single Link List (Singly Linked List) and Double Listed List (Doubly Linked List) can all be created as a Round Link List. Below we will learn how to create one.

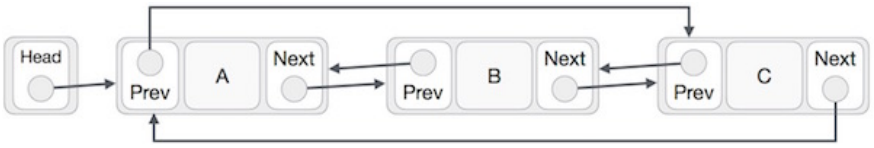
Create a linked Link List from a single linked list

In the Single Linked List, the next point to the end of the last node points to the first node, instead of pointing to NULL.



Create List of linked links from the Double Link List

In the Double-Link List, the next point of the last node points to the first node and the point pointing to the front of the previous node points to the last node. This process will form a ring in both directions.



Looking at the two illustrations above, you need to keep in mind:

Last Link's next pointer points to First Link in both cases with the Single Link List as well as the Double Link List.

First Link's Prev points to the last element of the Linked List in case of Double Link List.

Basic activities on the Round Link List

Here are some basic activities supported by the Round Link List:

Insert operation : insert an element into the starting position of the Round Link List.

Delete operation : delete an element of the Round Link List.

Display : display the entire Round Link List.

Insert operation in the Round Link List

The following is an algorithm that illustrates the insert operation in the Round Link List based on the Single Link List.

```
 //Chèn link t?i v? trí ??
u tiên void  insertFirst ( int  key , int  data ) { //t?o m?
t link struct  node  * link  = ( struct  node *)  malloc ( sizeof ( struct  node
? nó t?i first node c?  link -> next =  head ; //tr? first t?i first node m
?i  head  =  link ; } }
```

To monitor the code deployment section illustrating in detail in C language, go to the chapter: Program List of linked links in C.

Delete operation in the Round Link List

Below is an algorithm that illustrates the delete operation in the Round Link List based on a single linked list.

```
 //Xóa ph?n t? ??u tiên struct  node  *  deleteFirst () { //L?u tham chi?
u t?
i first link struct  node  * tempLink  =  head ; if ( head -> next ==  head ){  
?ánh d?u next t?i first link là first   head  =  head -> next ; //tr? v?
 link ?ã b? xóa return  tempLink ; }
```

To monitor the code deployment section illustrating in detail in C language, go to the chapter: Program List of linked links in C.

Show List of linked links

Here is an algorithm that illustrates the operation of displaying the entire Round Link List.

```
 //Hi?n th? danh sách liên k?
t vòng void  printList () { struct  node  * ptr  =  head ;  printf ( "n[ " ); //
?t ??u t? v? trí ??
u tiên if ( head  !=  NULL ) { while ( ptr -> next !=  ptr ) {  printf ( "(%d,%d
```

To monitor the code deployment section illustrating in detail in C language, go to the chapter: Program List of linked links in C.

According to Tutorialspoint

Previous article: Linked list data structure (Linked List)

Next article: Stack data structure (Stack)

You finished reading the article "**Data Link List structure (Circular Linked List)**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.
