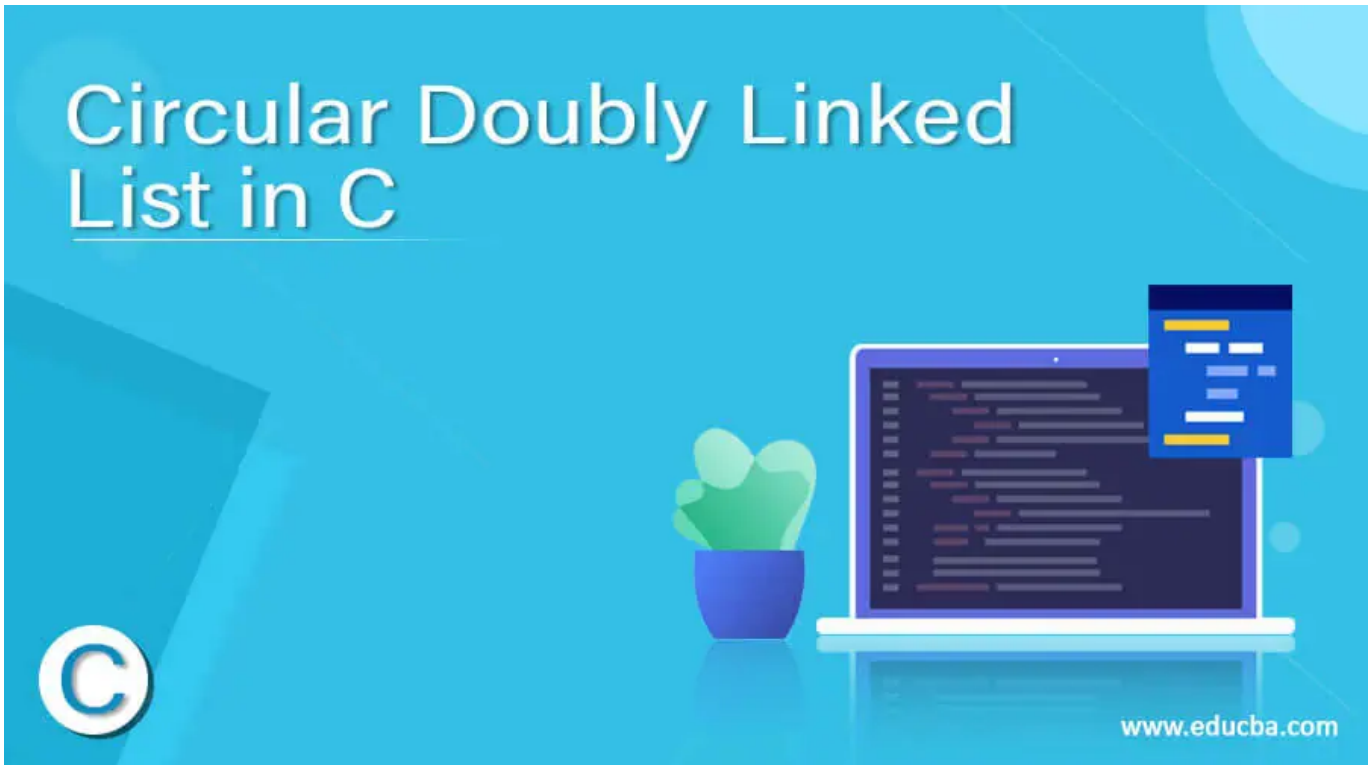
EDUCBA
(https://www.educba.com/software-development/)

← (https://www.educba.com/binary-search-in-c/)

→ (https://www.educba.com/circular-linked-lists-in-c/)




# Definition of Circular Doubly Linked List in C

Circular doubly linked list in C or in any programming language is a very useful data stru

Circular double linked list is a type of linked list that consists of node having a pointer pointing to the previous node and the next node points to the previous node in the defined array.

There is no particular syntax for the Circular doubly linked list but still needs to perform some of the initial steps of the creation of data structure and once created many operations can be performed on that linked list accordingly which is represented below :

```
#include <stdio.h>
Struct node_1
{
Struct node *prvs;
int some_data;
Struct node *nxt;
}
```

Follow scenarios:

– Insertion at the beginning

– Insertion at the end

– Removal from the beginning

– Removal from the end

Close the data structure and perform the further operation.


# How Circular doubly linked list works in C?

address of the previous or the first node of the entire list.

The first node present in the list contains address of the last node for the pointer in its previous node. Since a circular double-linked list demands three structures, therefore, it is required to have more space and more expensive operations especially on the basics part of it. Searching in the case of a doubly linked list becomes quite easy and efficient as manipulation with the pointers is easy. But sometimes developers don't prefer such data structure due to costly basic operation applied on the entire linked list.

(https://www.educba.com/software-development/)

The head as a variable contains the address of the first element of list. This first element is then the starting node of the list, the next node contains the second element, and so on till the last pointer which points back to the first node again proves the fact that the node is the last node that is pointing to the first since it does not contain any null element concept. There are various operations that are performed as part of the circular double linked list like insertion at the beginning, insertion at the end, deletion from the beginning, deletion at the end.

# Examples

Let us discuss examples of Circular Doubly Linked List in C.

## Example #1

This example represents an implementation of circular double-linked list with the operations of insertion at the beginning, insertion at the last, deletion at the beginning, and deletion at last which further displays the operation.

**Code:**

```
#include<stdio.h>
#include<stdlib.h>
struct nd_0
{
struct nd_0 *prv_1;
struct nd_0 *nxt_1;

int dta;
};
struct nd_0 *head;
```

(https://www.educba.com/software-development/)

```
void show();
void srch();
void main ()
{
int choce =0;
while(choce != 8)
{
printf("\n*********Main_Menu_for_Display*********\n");
printf("\nChoose_any_one_option_from_list ...\n");
printf("\n-------------------------------------------------
\n");
printf("\n1.Insertion_At_start\n2.Insertion_At_last\n3.Delet_at_Be
ginning\n4.Deletion_frm_end\n5.find\n6.display_val\n7.stop\n");
printf("\nSelect_the_desired_choice?\n");
scanf("\n%d",&choce);
switch(choce)
{
case 1:
insrtion_begnng();
break;
case 2:
insrtion_lst();
break;
case 3:
delnt_begnng();
break;
```

(https://www.educba.com/software-development/)

```
break;
case 6:
show();
break;
case 7:
exit(0);
break;
default:
printf("Select_entry_of_your_choicce..");
}
}
}
void insrtion_begnng()
{
struct nd_0 *ptr_0,*temp_1;
int item_0;
ptr_0 = (struct nd_0 *)malloc(sizeof(struct nd_0));
if(ptr_0 == NULL)
{
printf("\nList_Overflow");
}
else
{
printf("\nEnter desired_element");
scanf("%d",&item_0);
```

(https://www.educba.com/software-development/)

```
ptr_0 -> nxt_1 = head;
ptr_0 -> prv_1 = head;
}
else
{
temp_1 = head;
while(temp_1 -> nxt_1 != head)
{
temp_1 = temp_1 -> nxt_1;
}
temp_1 -> nxt_1 = ptr_0;
ptr_0 -> prv_1 = temp_1;
head -> prv_1 = ptr_0;
ptr_0 -> nxt_1 = head;
head = ptr_0;
}
printf("\nInserted_Node..\n");
}
}
void insrtion_lst()
{
struct nd_0 *ptr_0,*temp_1;
int itm_0;

ptr_0 = (struct nd_0 *) malloc(sizeof(struct nd_0));
if(ptr_0 == NULL)
{
```

(https://www.educba.com/software-development/)

```
printf(" \nEnter desired_val ");
scanf("%d",&itm_0);
ptr_0->dta=itm_0;
if(head == NULL)
{
head = ptr_0;
ptr_0 -> nxt_1 = head;
ptr_0 -> prv_1 = head;
}
else
{
temp_1 = head;
while(temp_1->nxt_1 !=head)
{
temp_1 = temp_1->nxt_1;
}
temp_1->nxt_1 = ptr_0;
ptr_0 ->prv_1=temp_1;
head -> prv_1 = ptr_0;
ptr_0 -> nxt_1 = head;
}
}
printf("\nnode_insertd_at_lst\n");
}
void delnt_begnng()
{
```

(https://www.educba.com/software-development/)

```
}
else if(head->nxt_1 == head)
{
head = NULL;
free(head);
printf("\ndelete_node_at_beginning\n");
}
else
{
temp_1 = head;
while(temp_1 -> nxt_1 != head)
{
temp_1 = temp_1 -> nxt_1;
}
temp_1 -> nxt_1 = head -> nxt_1;
head -> nxt_1 -> prv_1 = temp_1;
free(head);
head = temp_1 -> nxt_1;
}
}
void deln_lst()
{
struct nd_0 *ptr_1;

if(head == NULL)
{
printf("\n List_Underflow");
```

(https://www.educba.com/software-development/)

```
free(head);
printf("\nDeleted_Node\n");
}
else
{
ptr_1 = head;
if(ptr_1->nxt_1 != head)
{
ptr_1 = ptr_1 -> nxt_1;
}
ptr_1 -> prv_1 -> nxt_1 = head;
head -> prv_1 = ptr_1 -> prv_1;
free(ptr_1);
printf("\nDeleted_Node\n");
}
}
void show()
{
struct nd_0 *ptr_0;
ptr_0=head;
if(head == NULL)
{
printf("\nnot_to_print_anything;;");
}
else
{
```

(https://www.educba.com/software-development/)

```
ptr_0 = ptr_0 -> nxt_1;
}
printf("%d\n", ptr_0 -> dta);
}
}
void srch()
{
struct nd_0 *ptr_0;
int itm,i_0=0,flag=1;
ptr_0 = head;
if(ptr_0 == NULL)
{
printf("\nBlank_all_elements.\n");
}
else
{
printf("\nSearch_for_items?\n");
scanf("%d",&itm);
if(head ->dta == itm)
{
printf("found_location_item %d",i_0+1);
flag=0;
}

else
{
while (ptr_0->nxt_1 != head)
```

(https://www.educba.com/software-development/)

```
flag=0;
break;
}
else
{
flag=1;
}
i_0++;
ptr_0 = ptr_0 -> nxt_1;
}
}
if(flag != 0)
{
printf("Element_Not_found\n");
}
}
}
```

**Output:**

(https://www.educba.com/software-development/)

(https://www.educba.com/software-development/)

## Conclusion

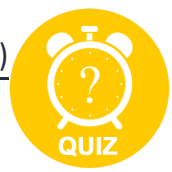
Circular Doubly linked list is a type of linked list and is part of data structure which has lot of advantages when it comes to memory management. It supports complex pointer concepts with ease. Lot of manipulations and operations can be performed on this data structure containing elements in a row.

## Recommended Articles

This is a guide to Circular Doubly Linked List in C. Here we discuss the definition, syntax, and parameters, How Circular doubly linked list works in C? examples with code implementation. You may also have a look at the following articles to learn more –

1. C# LinkedList (https://www.educba.com/c-sharp-linkedlist/)
2. Linked List in C (https://www.educba.com/linked-list-in-c/)
3. LinkedList in Java (https://www.educba.com/linkedlist-in-java/)
4. Reverse Linked List in Java (https://www.educba.com/reverse-linked-list-in-java/)





(https://www.educba.com/software-development/)

### Apps

iPhone & iPad (https://itunes.apple.com/in/app/educba-learning-app/id1341654580?mt=8)

Android (https://play.google.com/store/apps/details?id=com.educba.www)

### Resources

Free Courses (https://www.educba.com/software-development/free-courses/?source=footer)

Java Tutorials (https://www.educba.com/software-development/software-development-tutorials/java-tutorial/?source=footer)

Python Tutorials (https://www.educba.com/software-development/software-development-tutorials/python-tutorial/?source=footer)

All Tutorials (https://www.educba.com/software-development/software-development-tutorials/?source=footer)

### Certification Courses

All Courses (https://www.educba.com/software-development/courses/?source=footer)

Software Development Course - All in One Bundle (https://www.educba.com/software-development/courses/software-development-course/?source=footer)

Become a Python Developer (https://www.educba.com/software-development/courses/python-certification-course/?source=footer)

Java Course (https://www.educba.com/software-development/courses/java-course/?source=footer)

Become a Selenium Automation Tester (https://www.educba.com/software-development/courses/selenium-training-certification/?source=footer)


Become an IoT Developer (https://www.educba.com/software-development/courses/iot-course/?source=footer)

(https://www.educba.com/software-development/)