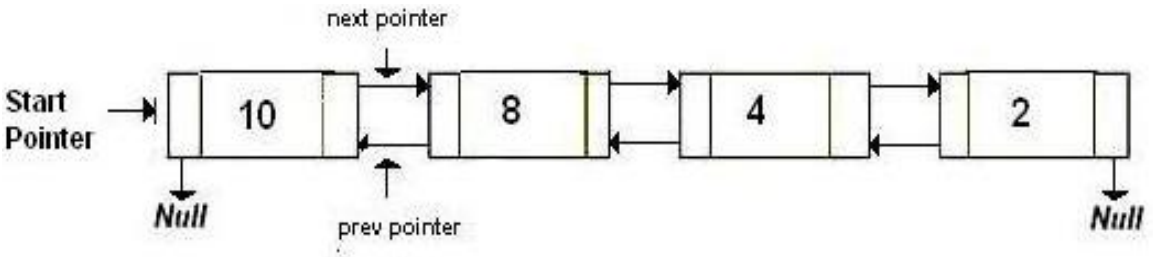
Reverse a Doubly Linked List

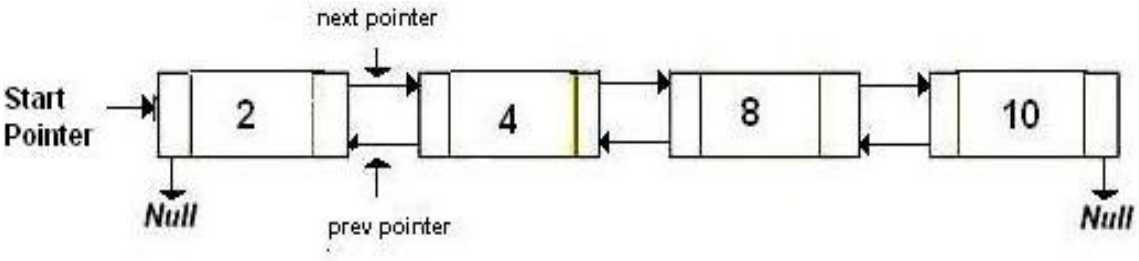
Write a C function to reverse a given Doubly Linked List

See below diagrams for example.

(a) Original Doubly Linked List



(b) Reversed Doubly Linked List



Here is a simple method for reversing a Doubly Linked List. All we need to do is swap prev and next pointers for all nodes, change prev of the head (or start) and change the head pointer in the

end.

```
/* Program to reverse a doubly linked list */
#include <stdio.h>
#include <stdlib.h>

/* a node of the doubly linked list */
struct node
{
  int data;
  struct node *next;
  struct node *prev;
};

/* Function to reverse a Doubly Linked List */
void reverse(struct node **head_ref)
{
      struct node *temp = NULL;
      struct node *current = *head_ref;

      /* swap next and prev for all nodes of
        doubly linked list */
      while (current !=  NULL)
      {
        temp = current->prev;
        current->prev = current->next;
        current->next = temp;
        current = current->prev;
      }

      /* Before changing head, check for the cases like empty
         list and list with only one node */
      if(temp != NULL )
         *head_ref = temp->prev;
}



/* UTILITY FUNCTIONS */
/* Function to insert a node at the beginging of the Doubly Linked Lis
void push(struct node** head_ref, int new_data)
{
    /* allocate node */
    struct node* new_node =
            (struct node*) malloc(sizeof(struct node));

    /* put in the data  */
```


Popular Posts

```
    new_node->data  = new_data;

    /* since we are adding at the begining,
      prev is always NULL */
    new_node->prev = NULL;

    /* link the old list off the new node */
    new_node->next = (*head_ref);

    /* change prev of head node to new node */
    if((*head_ref) !=  NULL)
      (*head_ref)->prev = new_node ;

    /* move the head to point to the new node */
    (*head_ref)    = new_node;
}

/* Function to print nodes in a given doubly linked list
   This function is same as printList() of singly linked lsit */
void printList(struct node *node)
{
  while(node!=NULL)
  {
   printf("%d ", node->data);
   node = node->next;
  }
}

/* Drier program to test above functions*/
int main()
{
  /* Start with the empty list */
  struct node* head = NULL;

  /* Let us create a sorted linked list to test the functions
   Created linked list will be 10->8->4->2 */
  push(&head, 2);
  push(&head, 4);
  push(&head, 8);
  push(&head, 10);

  printf("\n Original Linked list ");
  printList(head);

  /* Reverse doubly linked list */
  reverse(&head);
```

```
  printf("\n Reversed Linked list ");
  printList(head);

  getchar();
}
```

Time Complexity: O(n)

We can also swap data instead of pointers to reverse the Doubly Linked List. Method used for reversing array can be used to swap data. Swapping data can be costly compared to pointers if size of data item(s) is more.

Please write comments if you find any of the above codes/algorithms incorrect, or find better ways to solve the same problem.

Related Tpoics:

- Given a linked list, reverse alternate nodes and append at the end
- Pairwise swap elements of a given linked list by changing links
- Self Organizing List | Set 1 (Introduction)
- Merge a linked list into another linked list at alternate positions

Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 49 minutes ago

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 1 hour ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 1 hour ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 1 hour ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 1 hour ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 2 hours ago

- QuickSort on Singly Linked List
- Delete N nodes after M nodes of a linked list
- Design a stack with operations on middle element
- Swap Kth node from beginning with Kth node from end in a Linked List

Writing code in comment? Please use **ideone.com** and share the link here.

19 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

sam • 7 days ago
why the time required would be more in case of swapping data items
as there will be n/2 swaps only as compared to swapping pointers with n swa

Reply • Share ›

Amit • 4 months ago
I don't understand what is the need of all this code. Just store the pointer of the linked list is reversed. :) Done :P

2 Reply • Share ›

Castle Age Amit • 4 months ago
What is the last nodes's next pointer points to? NULL

Reply • Share ›

Nishant M Gandhi Amit • 4 months ago
revise your concepts via figure.
It wont work.

Reply • Share ›

▶ Java Reverse

Nikolche Kolev • 5 months ago

void reverse(Node * &head){

Node * prev = NULL;

Node * next;

Node * temp;

while(head){

temp = head;

//remember next

next = head->next;

// set next

head->next = prev;

// set prev

see more

1 ^ | v • Reply • Share ›

Arunx • 10 months ago

```
 node* reversedouble(node* &L){

node* ahead = L;
node* curr  = NULL;

if(!ahead){
```

```
        cout << " empty list";
        return L;
}

while(ahead){

 curr = ahead;
 ahead = ahead->next;
 curr-> next = curr -> prev;
 curr-> prev = ahead;

}
return curr;
}
```

• Reply • Share ›

Arunx • 10 months ago

[sourcecodnode* reversedouble(node* &L){

node* ahead = L;
node* curr = NULL;

if(!ahead){

cout << " empty list";
return L;
}

while(ahead){

curr = ahead;
ahead = ahead->next;

```
curr-> next = curr -> prev;
curr-> prev = ahead;

}

return curr;

}
```


abhishek08aug • a year ago

```
 #include<stdio.h>
#include<stdlib.h>

struct node {
  int data;
  struct node * prev;
  struct node * next;
};

void insert_node(struct node ** head_ref, struct node *prev_node, int
  struct node * head=*head_ref;
  struct node * new_node=NULL;
  if(head==NULL) {
    new_node=(struct node *)malloc(sizeof(struct node));
    new_node->data=value;
    new_node->prev=prev_node;
    new_node->next=NULL;
    *head_ref=new_node;
```

prakash • a year ago

by just swaping the data of elements itself will satisfy the requirement.since th
and tail ptrs
front_ptr=head;
back_ptr=tail;
while(front_ptr!=back_ptr && back_ptr->next!=frotn_ptr)
{
temp_var=back_ptr->data;
back_ptr->data=front_ptr->data;
front_ptr->data=temp_var
}

4 ▲ | ▼ • Reply • Share ›

Ashish Rai • a year ago

Yes we need to preserve the meaning of prev and next pointers.
so why can&#039t we swap the values of the prev and next pointers inside the
swapping the nodes?
Please do give your opinion.

▲ | ▼ • Reply • Share ›

whizkid08 • a year ago

Wrote the same function using Recursion:

```
 #include<stdio.h>
#include<stdlib.h>
#include<conio.h>

struct Node
{
int data;
struct Node* next;
```

```
struct Node* prev;
};

void push(struct Node** head, int val)
{
        struct Node* temp = (struct Node*)malloc(sizeof(struct Node));

        if((*head)==0)
```

see more

• Reply • Share ›

pinkyponky • a year ago

```
 /* Paste your code here (You may delete these lines if not writing c
```

```
#include<iostream>
#include<stdio.h>
using namespace std;
struct list
{
int data;
struct list *next;
struct list *prev;
}*head=NULL,*head1;
int ins(int n)
{
struct list *temp,*temp1;
if(!head)
{
```

see more

1 ^ | v • Reply • Share ›

Nitin gupta iitian • 2 years ago

```
template
void Doubly :: ReverseLL( )
{
Current = Head ;
Tail = Head ;

while ( Current)
{
Head = Current->prev ;
Current->prev = Current->next ;
Current->next = Head ;
Current = Current->prev ;

}
if ( Head )
Head = Head->prev ;

}
```

```
/* Paste your code here (You may delete these lines if not writing co
```

^ | v • Reply • Share ›

Ankur • 3 years ago

A more simpler one

```
void reverse(struct node **head_ref)
{
struct node *temp = NULL ;
```

```
struct node *temp = NULL;
struct node *current = *head_ref;

/* swap next and prev for all nodes of
doubly linked list */
while (current != NULL)
{
// temp = current->prev;
current->prev = current->next;
current->next = temp;
temp = current;
current = current->prev;
}
*head_ref = temp;
}
```

1 ▲ | ▼ • Reply • Share ›

jagdish • 3 years ago

Can't we just set the head pointer to the last node

that will reverse the list what say?

▲ | ▼ • Reply • Share ›

Sandeep ➦ jagdish • 3 years ago

@jagdish: We need to change prev and next pointers of all nodes so th
remains same.

▲ | ▼ • Reply • Share ›

vinodh ➦ Sandeep • 2 years ago

Yes we need to preserve the meaning of prev and next pointers
so why can't we swap the values of the prev and next pointers i
the swapping the nodes?
Please do give your opinion

```
/* Paste your code here (You may delete these lines if r
```


vinodh ➦ Sandeep • 2 years ago

Yes we need to preserve the meaning of prev and next pointers so why can't we swap the values of the prev and next pointers i the swapping the nodes?
Please do give your opinion.

```
/* Paste your code here (You may delete these lines if r
```


Bragaadeesh • 4 years ago

Here is my two cents,

Program to reverse a singly list ITERATIVELY

Program to reverse a linked list RECURSIVELY