18/11/2020

LP #5

WAP to implement singly linked list with following operations a) create a linked list b) Insertion of a node at first position, and at end of list c) display the contents of the linked list d) deletion of first element and at last element in the list.

```
#include <stdio.h>
#include <stdlib.h>
struct node {
int info;
struct node *link;
};
```

Scanned with CamScanner

```
typedef struct node *NODE ;
NODE getnode ()
{

NODE x ;
x = (NODE) malloc (sizeof (structnode));
if (x==NULL)
{

printf ("mem full \n");
exit(0);
}

return x;
}
void freenode (NODE x)
{

free (x);
 }


NODE insert_front (NODE first, int item)
{

NODE temp;
temp = getnode();
temp -> info = item;
temp -> link = NULL;
if (first = NULL)
```

Scanned with CamScanner

```
return temp;

temp -> link = first;
first = temp;
return first;
}


NODE delete_front (NODE first )
{

NODE temp;

if ( first == NULL)
{

printf ("list is empty cannot delete \n");

return first;
}

temp = first;

temp = temp -> link;

printf (" item deleted at front end is = %d \n ", first -> info);

free (first);

return temp;
}


NODE insert_rear (NODE first, int item)
{

NODE temp, cur;

temp = getnode ();
```

Scanned with CamScanner

```
temp → info = item ;
temp → link = NULL ;
if (first == NULL)
  return temp;
cur = first ;
while (cur → link ! = NULL)
 {cur = cur → link ; }
 cur → link = temp ;
 return first ;
 }


NODE delete _rear (NODE first)
{
 NODE cur, prev;
 if (first == NULL)
  {
 printf ("list is empty cannot delete \n");
 return first ;
 }
 if (first → link == NULL)
  {
 printf ("item deleted is %d\n", first → info);
 free (first) ;
 return NULL;
}
prev = NULL;
cur = first;
while (cur → link! = NULL)
{
prev = cur;
cur = cur → link;
}
printf (" item deleted at rear-end is %d", cur → info);
free (cur);
prev → link = NULL;
return first;
}


void display (NODE first)
{
NODE temp;
if (first == NULL)
printf(" list empty cannot display items\n");
for (temp = first; temp! = NULL; temp = temp → link)
{
printf ("%d\n", temp → info);
        }
}
```

Scanned with CamScanner

```
int main () {
int item, choice;
NODE first = NULL ;
printf ("\n1: Insert front\n2 : Delete front \n 3: Insert rear \n 4: Delete_
rear \n 5: display_list \n 6: Exit\n");
do
{

printf ("\n enter the choice\n");
scanf ("%d", &choice);
switch (choice)
{
case 1 : printf ("enter the item at front-end \n");
         scanf ("%d.", &item);
         first = insert_front (first, item);
         break;
case 2 : first = delete_front (first);
         break;
case 3 : printf ("enter the item at rear-end\n");
         scanf ("%d", &item);
         first = insert_rear (first, item);
         break;
case 4 : first = delete_rear (first);
         break;
case 5 : display (first); break;
```

Scanned with CamScanner

classmate
Date
Page

```
case 6 : break;
default: break;
}
} while (choice != 6);
return 0;
 }
```

Scanned with CamScanner