**Programs**

1. Write a program to implement circular linked list.

Ans:

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

struct node

{

int data;

struct node \*next;

};

struct node \*head;

void beg\_insert();

void end\_insert();

void beg\_delete();

void end\_delete();

void display();

void search();

void main()

{

int ch=0;

clrscr();

while(ch!=7)

{

printf("\n---------------------------Main Menu-----------------------------------");

printf("\n1.Beg\_Insert \t2.End\_Insert \t3.Del\_Beg \t4.End\_Delete");

printf("\n5.Display \t6.Search \t7.Exit");

printf("\n---------------------------------------------------------------------");

printf("\nEnter your choice:");

scanf("%d",&ch);

switch(ch)

{

case 1:beg\_insert();

break;

case 2:end\_insert();

break;

case 3:beg\_delete();

break;

case 4:end\_delete();

break;

case 5:display();

break;

case 6:search();

break;

case 7:exit(0);

break;

default:printf("\nPlease enter valid choice...");

}

}

}

void beg\_insert()

{

struct node \*ptr,\*temp;

int item;

ptr=(struct node \*)malloc(sizeof(struct node));

if(ptr==NULL)

{

printf("Circular Linked List is empty");

}

else

{

printf("Enter the node data:");

scanf("%d",&item);

ptr->data=item;

if(head==NULL)

{

head=ptr;

ptr->next=head;

}

else

{

temp=head;

while(temp->next!=head)

temp=temp->next;

ptr->next=head;

temp->next=ptr;

head=ptr;

}

printf("Node inserted Successfully...");

}

}

void end\_insert()

{

struct node \*ptr,\*temp;

int item;

ptr=(struct node \*)malloc(sizeof(struct node));

if(ptr==NULL)

{

printf("Circular Linked List is empty");

}

else

{

printf("Enter data:");

scanf("%d",&item);

ptr->data=item;

if(head==NULL)

{

head=ptr;

ptr->next=head;

}

else

{

temp=head;

while(temp->next!=head)

{

temp=temp->next;

}

temp->next=ptr;

ptr->next=head;

}

printf("Node inserted Successfully...");

}

}

void beg\_delete()

{

struct node \*ptr;

if(head==NULL)

{

printf("Circular Linked List is empty");

}

else if(head->next==head)

{

head=NULL;

free(head);

printf("Node deleted Successfully...");

}

else

{

ptr=head;

while(ptr->next!=head)

ptr=ptr->next;

ptr->next=head->next;

free(head);

head=ptr->next;

printf("Node deleted Successfully...");

}

}

void end\_delete()

{

struct node \*ptr,\*prev;

if(head==NULL)

{

printf("Circular Linked List is empty");

}

else if(head->next==head)

{

head=NULL;

free(head);

printf("Node deleted Successfully...");

}

else

{

ptr=head;

while(ptr->next!=head)

{

prev=ptr;

ptr=ptr->next;

}

prev->next=ptr->next;

free(ptr);

printf("Node deleted Successfully...");

}

}

void display()

{

struct node \*ptr;

ptr=head;

if(ptr==NULL)

{

printf("Circular Linked List is empty");

}

else

{

printf("Elements in list are:\n");

while(ptr->next!=head)

{

printf("%d\t",ptr->data);

ptr=ptr->next;

}

printf("%d",ptr->data);

}

}

void search()

{

struct node \*ptr;

int item,i=0,flag=1;

ptr=head;

if(head==NULL)

{

printf("Circular Linked List is empty");

}

else

{

printf("Enter item which you want to search:");

scanf("%d",&item);

if(head->data==item)

{

printf("Item found at location %d",i+1);

flag=0;

}

else

{

while(ptr->next!=head)

{

if(ptr->data==item)

{

printf("Item found at location %d",i+1);

flag=0;

break;

}

else

{

flag=1;

}

i++;

ptr=ptr->next;

}

}

if(flag!=0)

{

printf("Item not found");

}

}

}

Output:





