WEEK-6

Date:30-07-2025

List of programs:

1. Write a C program to insert a node at the beginning in a Circular single linked list.
2. Write a C program to insert a node at the end in a Circular single linked list.
3. Write a C program to insert a node after a given node(middle case) in a Circular single linked list.
4. Write a C program to delete a node at the beginning in a Circular single linked list.
5. Write a C program to delete a node at the end in a Circular single linked list
6. Write a C program to delete a node after a given node(middle case) in a Circular single linked list.

1. **Aim:** To write a C program to insert a node at the beginning in a Circular single linked list.

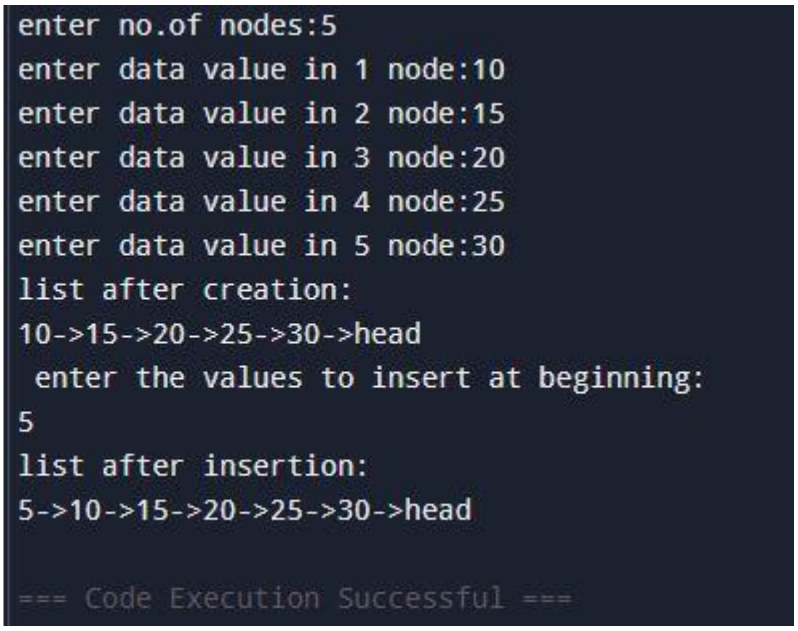
Program:

```
#include<stdio.h>

#include<stdlib.h>

struct node

{

   int data;

   struct node*next;

};

struct node*head=NULL,*new,*temp,*last=NULL;

struct node*getnode(int x)

{

   struct node*temp=(struct node*)malloc(sizeof(struct node));

   temp->data=x;

   temp->next=NULL;
```

```
    return temp;
  }
  void insert_begin()
  {
    if(head==NULL)
    {
      head=new;
      last=new;
      new->next=head;
    }
    else
    {
      new->next=head;
      head=new;
      last->next=head;
    }
  }
  void display()
  {
    if(head==NULL)
    {
      printf("list is empty\n");
    }
    else
    {
      temp=head;
      do
      {
            printf("%d->",temp->data);
            temp=temp->next;
         }
         while(temp!=head);
         printf("head");
      }
   }
   int main()
   {
      int n,x,i;
      printf("enter no.of nodes:");
      scanf("%d",&n);
      for(i=0;i<n;i++)
      {
         printf("enter data value in %d node:",i+1);
         scanf("%d",&x);
         new=getnode(x);
         if(head==NULL)
         {
            head=new;
            last=new;
            new->next=head;
         }
         else
         {
            last->next=new;
            last=new;
            last->next=head;
```

```
        }
    }
    printf("list after creation:\n");
    display();
    printf("\n enter the values to insert at beginning:\n");
    scanf("%d",&x);
    new=getnode(x);
    insert_begin();
    printf("list after insertion:\n");
    display();
    return 0;
}
```

Output:



```
enter no.of nodes:5
enter data value in 1 node:10
enter data value in 2 node:15
enter data value in 3 node:20
enter data value in 4 node:25
enter data value in 5 node:30
list after creation:
10->15->20->25->30->head
 enter the values to insert at beginning:
5
list after insertion:
5->10->15->20->25->30->head


=== Code Execution Successful ===
```

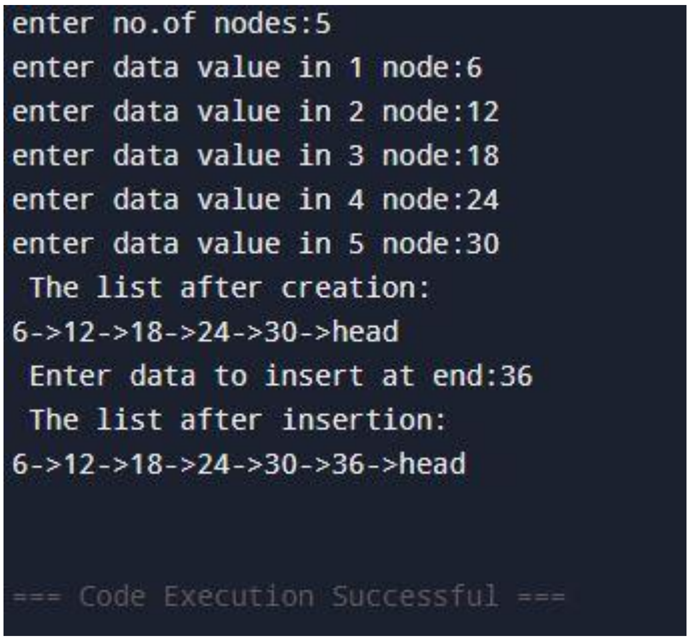
2. **Aim:** Write a C program to insert a node at the end in a Circular single linked list.

Program:

```
#include<stdio.h>

#include<stdlib.h>

struct node

{

   int data;

   struct node*next;

};

struct node*new,*last=NULL,*head=NULL,*temp;

struct node*getnode(int x)

{

   struct node*temp=(struct node*)malloc(sizeof(struct node));

   temp->data=x;

   temp->next=NULL;

   return temp;

}

void insert_end()

{

   if(head==NULL)

   {

      head=last=new;

      last->next=head;

   }

   else
```

```
    {
        last->next=new;
        last=new;
        last->next=head;
    }
}
void display()
{
    if(head==NULL)
    {
        printf("list is empty\n");
    }
    else
    {
        struct node*temp=head;
        do
        {
            printf("%d->",temp->data);
            temp=temp->next;
        }
        while(temp!=head);
        printf("head\n");
    }
}
int main()
{
    int n,x,i;
    printf("enter no.of nodes:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("enter data value in %d node:",i+1);
        scanf("%d",&x);
        new=getnode(x);
        if(head==NULL)
        {
            head=last=new;
            last->next=head;
        }
        else
        {
            last->next=new;
            last=new;
            last->next=head;
        }
    }
    printf(" The list after creation:\n");
    display();
    printf(" Enter data to insert at end:");
    scanf("%d",&x);
    new=getnode(x);
    insert_end();
    printf(" The list after insertion:\n");
    display();
    return 0;
}
```

Output:



```
enter no.of nodes:5
enter data value in 1 node:6
enter data value in 2 node:12
enter data value in 3 node:18
enter data value in 4 node:24
enter data value in 5 node:30
 The list after creation:
6->12->18->24->30->head
 Enter data to insert at end:36
 The list after insertion:
6->12->18->24->30->36->head


=== Code Execution Successful ===
```

3. **Aim:** Write a C program to insert a node after a given node(middle case) in a Circular single linked list.

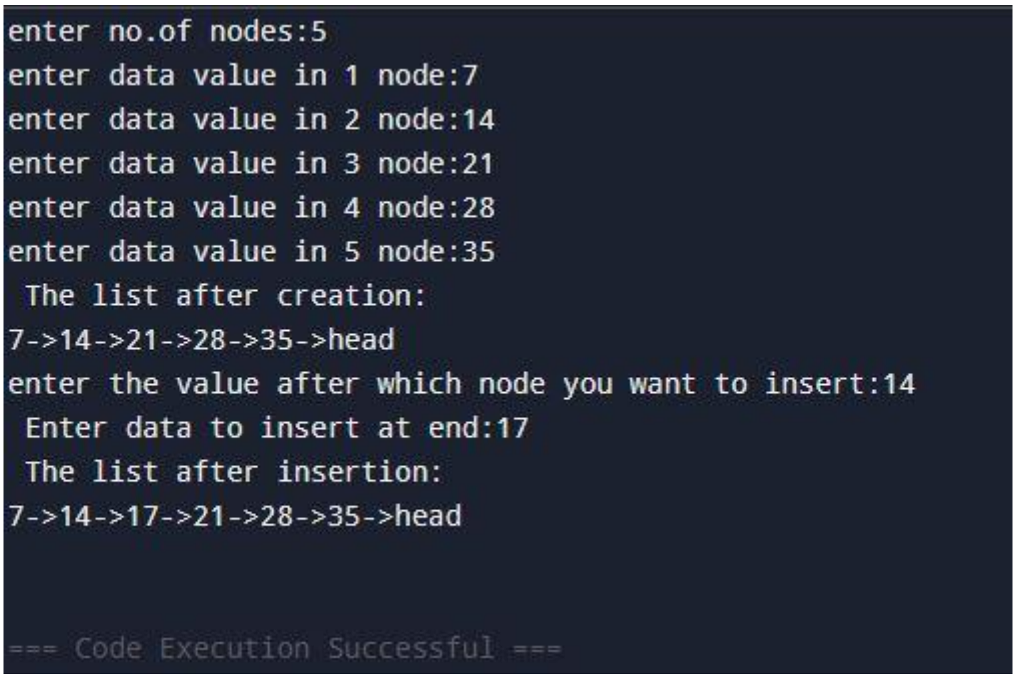
Program:

```
#include<stdio.h>

#include<stdlib.h>

struct node

{

   int data;

   struct node*next;

};

struct node*new,*last,*head=NULL,*temp;

struct node*getnode(int x)

{

   struct node*temp=(struct node*)malloc(sizeof(struct node));

   temp->data=x;

   temp->next=NULL;

   return temp;

}

void insert_middle(int x,int val)

{

   temp=head;

   new=getnode(x);

   if(head==NULL)

   {

      printf("insertion is not possible \n");

   }
```

```
    else
    {
      while(temp->data!=val&&temp->next!=head)
      {
        temp=temp->next;
      }
      if(temp->data==val)
      {
        new->next=temp->next;
        temp->next=new;
      }
    }
}
void display()
{
    if(head==NULL)
    {
      printf("list is empty\n");
    }
    else
    {
      struct node*temp=head;
      do
      {
        printf("%d->",temp->data);
        temp=temp->next;
      }
      while(temp!=head);
```

```
        printf("head\n");
    }
}
int main()
{
    int n,x,i,val;
    printf("enter no.of nodes:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("enter data value in %d node:",i+1);
        scanf("%d",&x);
        new=getnode(x);
        if(head==NULL)
        {
            head=new;
            last=new;
            last->next=head;
        }
        else
        {
            last->next=new;
            last=new;
            last->next=head;
        }
    }
    printf(" The list after creation:\n");
    display();
    printf("enter the value after which node you want to insert:");
    scanf("%d",&val);
    printf(" Enter data to insert at end:");
    scanf("%d",&x);
    insert_middle(x,val);
    printf(" The list after insertion:\n");
    display();
    return 0;
}
```

Output:



```
enter no.of nodes:5
enter data value in 1 node:7
enter data value in 2 node:14
enter data value in 3 node:21
enter data value in 4 node:28
enter data value in 5 node:35
 The list after creation:
7->14->21->28->35->head
enter the value after which node you want to insert:14
 Enter data to insert at end:17
 The list after insertion:
7->14->17->21->28->35->head



=== Code Execution Successful ===
```

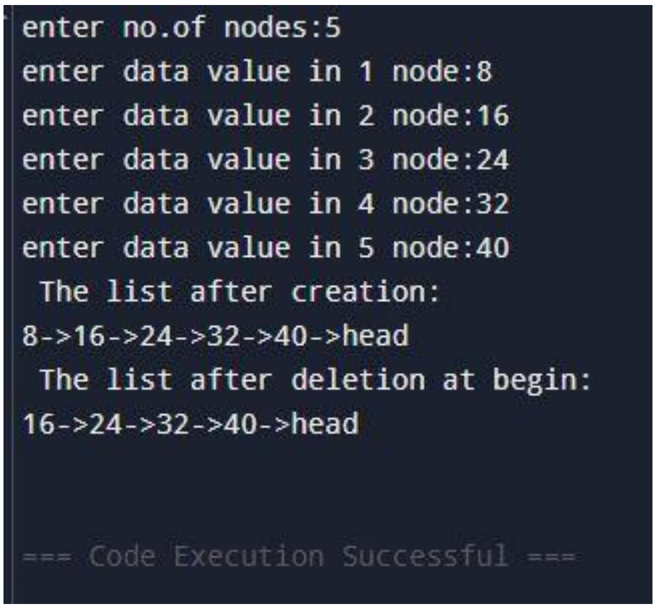
4. **Aim:** To write a C program to delete a node at the beginning in a Circular single linked list.

Program:

```
#include<stdio.h>

#include<stdlib.h>

struct node

{

   int data;

   struct node*next;

};

struct node*new,*last,*head,*temp;

struct node*getnode(int x)

{

   struct node*temp=(struct node*)malloc(sizeof(struct node));

   temp->data=x;

   temp->next=NULL;

   return temp;

}

void delete_begin()

{

   if(head==NULL)

   {

      printf("deletion is not possible \n");

   }

   else if(head->next==head)
```

```
    {
        temp=head;
        head=NULL;
        free(temp);
    }
    else
    {
      temp=head;
      head=head->next;
      last->next=head;
      free(temp);
    }
}
void display()
{
    if(head==NULL)
    {
        printf("list is empty\n");
    }
    else
    {
        struct node*temp=head;
        do
        {
            printf("%d->",temp->data);
            temp=temp->next;
        }
        while(temp!=head);
        printf("head\n");
    }
}
int main()
{
    int n,x,i;
    printf("enter no.of nodes:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("enter data value in %d node:",i+1);
        scanf("%d",&x);
        new=getnode(x);
        if(head==NULL)
        {
            head=new;
            last=new;
            last->next=head;
        }
        else
        {
            last->next=new;
            last=new;
            last->next=head;
        }
    }
    printf(" The list after creation:\n");
    display();
    delete_begin();

    printf(" The list after deletion at begin:\n");

    display();

    return 0;

}
```

Output:



```
enter no.of nodes:5
enter data value in 1 node:8
enter data value in 2 node:16
enter data value in 3 node:24
enter data value in 4 node:32
enter data value in 5 node:40
 The list after creation:
8->16->24->32->40->head
 The list after deletion at begin:
16->24->32->40->head


=== Code Execution Successful ===
```

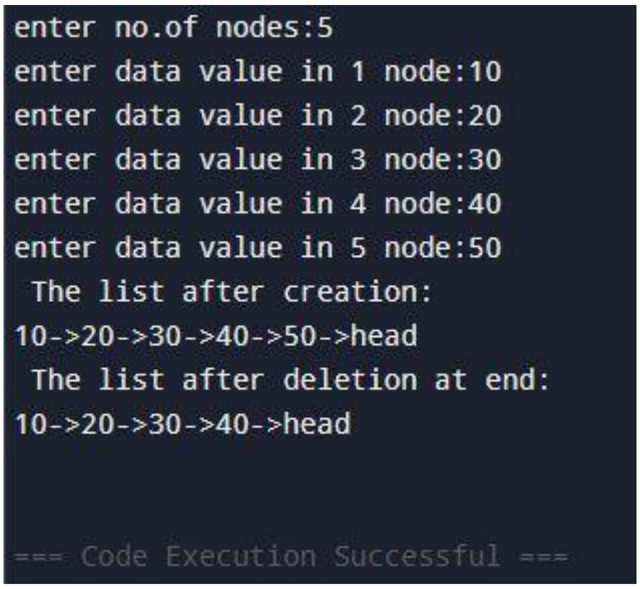
5. **Aim:** Write a C program to delete a node at the end in a Circular single linked list.

Program:

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
   int data;
   struct node*next;
};
struct node*new,*last=NULL,*head,*temp;
struct node*getnode(int x)
{
   struct node*temp=(struct node*)malloc(sizeof(struct node));
   temp->data=x;
   temp->next=NULL;
   return temp;
}
void delete_end()
{
   if(head==NULL)
   {
      printf("deletion is not possible \n");
   }
   else if(head->next==head)
   {
        temp=head;
        head=last=NULL;
        free(temp);
    }
    else
    {
      temp=head;
      struct node*temp2=NULL;
      while(temp->next!=head)
      {
         temp2=temp;
         temp=temp->next;
      }
      last=temp2;
      last->next=head;
      free(temp);
    }
}
void display()
{
    if(head==NULL)
    {
        printf("list is empty\n");
    }
    else
    {
        struct node*temp=head;
        do
        {
           printf("%d->",temp->data);
           temp=temp->next;
        }
        while(temp!=head);
        printf("head\n");
    }
}
int main()
{
    int n,x,i;
    printf("enter no.of nodes:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("enter data value in %d node:",i+1);
        scanf("%d",&x);
        new=getnode(x);
        if(head==NULL)
        {
           head=new;
           last=new;
           last->next=head;
        }
        else
        {
          temp=head;
          while(temp->next!=head)
```

```
            {
               temp=temp->next;
            }
            temp->next=new;
            last=new;
            last->next=head;
         }
      }
      printf(" The list after creation:\n");
      display();
      delete_end();
      printf(" The list after deletion at end:\n");
      display();
      return 0;
   }
```

Output:



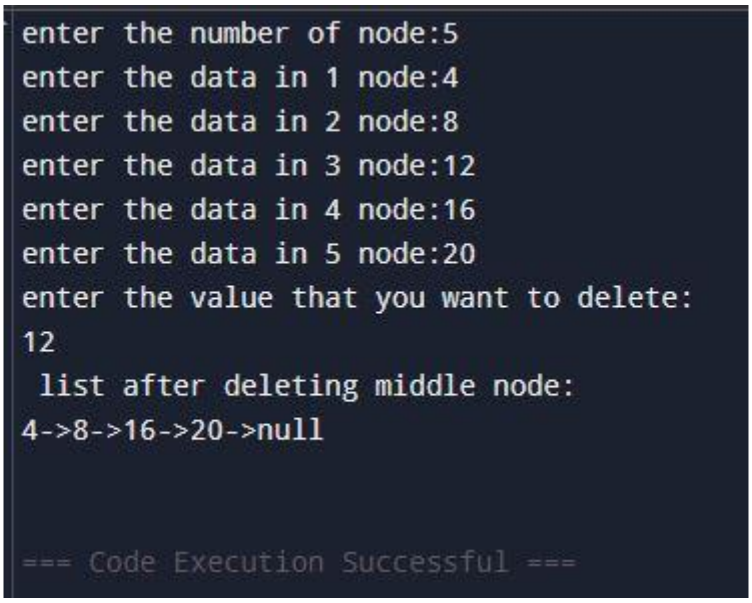
```
enter no.of nodes:5
enter data value in 1 node:10
enter data value in 2 node:20
enter data value in 3 node:30
enter data value in 4 node:40
enter data value in 5 node:50
 The list after creation:
10->20->30->40->50->head
 The list after deletion at end:
10->20->30->40->head


=== Code Execution Successful ===
```

6. **Aim:** Write a C program to delete a node after a given node(middle case) in a Circular single linked list.

Program:

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
   int data;
   struct node*next;
};
struct node*head,*new,*temp,*last=NULL;
struct node*getnode(int x)
{
   struct node*temp=(struct node*)malloc(sizeof(struct node));
   temp->data=x;
   temp->next=NULL;
   return temp;
}
void display()
{
   if(head==NULL)
   {
    printf("list is empty\n");
   }
   else
   {
        temp=head;
        do
        {
            printf("%d->",temp->data);
            temp=temp->next;
        }
        while(temp!=head);
        printf("head");
    }
}
void delete_middle(int val)
{
    if(head==NULL)
    {
        printf("list is empty\n");
    }
    temp=head;
    struct node *temp1=NULL;
    while(temp->data!=val)
    {
        if(temp->next==head)
        {
            printf("node%not found\n",val);
        }
        temp1=temp;
        temp=temp->next;
    }
    if(temp==head||temp->next==head)
    {
        printf("node value%d is not a middle node value",val);
    }
    temp1->next=temp->next;
    free(temp);
}
int main()
{
    int n,x,i,val;
    printf("enter no.of nodes:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("enter data value in %d node:",i+1);
        scanf("%d",&x);
        new=getnode(x);
        if(head==NULL)
        {
            head=new;
            last->next=head;
        }
        else
        {
            temp=head;
            while(temp->next!=head)
            {
                temp=temp->next;
            }
            temp->next=new;
            last=new;
            last->next=head;
        }
    }
    printf("Enter the value to delete at middle:\n");
    scanf("%d",&val);
    printf("The list after deletion:");
    delete_middle(val);
    display();
    return 0;
}
```

Output:



```
enter the number of node:5
enter the data in 1 node:4
enter the data in 2 node:8
enter the data in 3 node:12
enter the data in 4 node:16
enter the data in 5 node:20
enter the value that you want to delete:
12
 list after deleting middle node:
4->8->16->20->null


=== Code Execution Successful ===
```

Inferences:

- In circular linked list, the **last node does not point to NULL** – it points back to the **head node**.
- Traversal can be done starting from **any node** since the list is circular.
- There is **no natural end** (need a condition to stop traversal, e.g., when pointer reaches head again).
- **Head pointer is still important**, but even if head is lost, list can still be traversed from any known node.
- **Efficient for round-robin scheduling** (CPU scheduling, buffering, etc.).
- Insertion at the **beginning or end** can be done in **O(1)** time if a tail pointer is maintained.
- Same as singly linked list, requires **extra memory for next pointer**.
- Searching is still **O(n)** since traversal may need to cover the whole list.
- More flexible than linear singly linked list in **repeated traversals** (no need to restart at head).