Ex.no:04 Implementation of Circular Lists & Applications of Linked List

Date: 23.07.24

Aim:

Program:

1.) Merge two sorted circular linked list into one sorted circular linked list and display the third linked list.

2.) Given a circular doubly linked list consisting of N nodes, the task is to modify every node of the given Linked List such that each node contains the sum of all nodes except that node.

3.) Develop an employee job scheduling system using a circular linked list. Traverse through all the employee tasks one at a time. And find out which employee completes the allotted task first.

Algorithm:

Code:

1.)

```
#include <bits/stdc++.h>
using namespace std;
struct Node {
    int data;
    Node *next, *prev;
};
void insert(Node** head_ref, int data)
{
Node* new_node = new Node;
new_node->data = data;
if (*head_ref == NULL) {
        new_node->next = new_node;
        new_node->prev = new_node;}
else{
        Node* last = (*head_ref)->prev;
        new_node->next = *head_ref;
        new_node->prev = last;
        last->next = (*head_ref)->prev = new_node;}
*head_ref = new_node;
}

Node* merge(Node* first, Node* second)
{

    if (!first)
        return second;
    if (!second)
```

```
        return first;
    if (first->data < second->data) {
        first->next = merge(first->next, second);
        first->next->prev = first;
        first->prev = NULL;
        return first;
    }
    else {
        second->next = merge(first, second->next);
        second->next->prev = second;
        second->prev = NULL;
        return second;}}

Node* mergeUtil(Node* head1, Node* head2)
{

    if (!head1)
        return head2;

    if (!head2)
        return head1;
    Node* last_node;
    if (head1->prev->data < head2->prev->data)
        last_node = head2->prev;
    else
        last_node = head1->prev;

    head1->prev->next = head2->prev->next = NULL;
```

```
    Node* finalHead = merge(head1, head2);

    finalHead->prev = last_node;
    last_node->next = finalHead;
    return finalHead;}

void printL(Node* head)
{
    Node* temp = head;

    while (temp->next != head) {
        cout << temp->data << " ";
        temp = temp->next;
    }
    cout << temp->data << " ";
}
int main()
{
    Node *head1 = NULL, *head2 = NULL;

    insert(&head1, 8);
    insert(&head1, 5);
    insert(&head1, 3);
    insert(&head1, 1);

    insert(&head2, 11);
    insert(&head2, 9);
    insert(&head2, 7);
    insert(&head2, 2);
```

```
    Node* newHead = mergeUtil(head1, head2);
     cout << "Final Sorted List: ";
    printL(newHead);


    return 0;
}
```

Output:

```
Final Sorted List: 1 2 3 5 7 8 9 11
Process returned 0 (0x0)   execution time : 0.087 s
Press any key to continue.
```

2.)

```
#include<iostream>
using namespace std;
class node
{
    public:
    int data;
    node* next;
    node* prev;
};
void sumofnodes(node* head)
{
    node* temp=head;
    int sum=0;
    if(head!=NULL)
    {
        do{
            sum=sum+temp->data;
            temp=temp->next;
        }while(temp!=head);}
```

```
    temp=head;
    do{
        int currentsum=sum-temp->data;
        temp=temp->next;
        cout<<currentsum<<"->";
    }while(temp!=head);
    cout<<"end";
}
int main()
{
    int d1,d2,d3,d4,d5;
    node* head=NULL;
    node* second=NULL;
    node* third=NULL;
    node* four=NULL;
    node* five=NULL;

    cout<<"Enter data for 1st node:";
    cin>>d1;
    cout<<"Enter data for 2nd node:";
    cin>>d2;
    cout<<"Enter data for 3rd node:";
    cin>>d3;
    cout<<"Enter data for 4th node:";
    cin>>d4;
    cout<<"Enter data for 5th node:";
    cin>>d5;
    head=new node();
    second=new node();
    third=new node();
```

```
    four=new node();
    five=new node();
    head->prev=five;
    head->data=d1;
    head->next=second;
    second->prev=head;
    second->data=d2;
    second->next=third;
    third->prev=second;
    third->data=d3;
    third->next=four;
    four->prev=third;
    four->data=d4;
    four->next=five;
    five->prev=four;
    five->data=d5;
    five->next=head;
    sumofnodes(head);
}
```

Output:

```
Enter data for 1st node:7
Enter data for 2nd node:5
Enter data for 3rd node:3
Enter data for 4th node:8
Enter data for 5th node:1
17->19->21->16->23->end
Process returned 0 (0x0)   execution time : 8.752 s
Press any key to continue.
```

3.)

```
#include<iostream>
#include<string>
using namespace std;
class node
{
   public:
   int data1,data3;
   string data2;
   node* next;};
void completefirst(node* head)
{
   int min=head->data3;
   node* temp=head->next;
   if(head!=NULL)
   {
      do
      {
         if(temp->data3<min)
      {
         min=temp->data3;

      }
      temp=temp->next;
      }while(temp!=head);
   }
   cout<<"number "<<min<<" with task allotment will complete first"<<endl;

}
int main()
```

```
{
    int id,task;
    string name;
   node* head=NULL;
   node* second=NULL;
   node* third=NULL;

   head=new node();
   second=new node();
   third=new node();

 cout<<"Enter the id of the employee:";
   cin>>id;
   cout<<"Enter name of the employee:";
   cin>>name;
   cout<<"enter number of tasks:";
   cin>>task;
   head->data1=id;
   head->data2=name;
   head->data3=task;
   head->next=second;

   cout<<"Enter the id of the employee:";
   cin>>id;
   cout<<"Enter name of the employee:";
   cin>>name;
   cout<<"enter number of tasks:";
   cin>>task;
   second->data1=id;
   second->data2=name;
```

```
    second->data3=task;
    second->next=third;

    cout<<"Enter the id of the employee:";
    cin>>id;
    cout<<"Enter name of the employee:";
    cin>>name;
    cout<<"enter number of tasks:";
    cin>>task;
    third->data1=id;
    third->data2=name;
    third->data3=task;
    third->next=head;

    completefirst(head);
}
```

Output:

```
Enter the id of the employee:8622
Enter name of the employee:arun
enter number of tasks:05
Enter the id of the employee:7622
Enter name of the employee:sriram
enter number of tasks:05
Enter the id of the employee:6622
Enter name of the employee:rv
enter number of tasks:07
number 5 with task allotment will complete first

Process returned 0 (0x0)   execution time : 25.708 s
Press any key to continue.
```

Result:

The above programs are executed successfully.