**Assignment No. 8**

**AIM**:- Impliment polynomial using circular linked list and perform

i: Addition/multiplication of polynomials

Ii:Evaluation of polynomials

**OBJECTIVE**:-

Circular lists are the required data structure when we want a list to be accessed in a circle or loop. Address reference to previous node can easily be found which is not posiible in single linked list.from this polynomial arithmatic using circular linked lists we can understand accessing and handling the data of circular linked lists and perform different operations.

**Theory** :-

1. Circular linked list is a linked list where all nodes are connected to form a circle. There is no NULL at the end. A circular linked list can be a singly circular linked list or doubly circular linked list. ... We can maintain a pointer to the last inserted node and front can always be obtained as next of last.

2. Following are the important operations supported by a circular list.

insert − Inserts an element at the start of the list.

delete − Deletes an element from the start of the list.

display − Displays the list.

3. A polynomial p(x) is the expression in variable x which is in the form (axn + bxn-1 + …. + jx+ k), where a, b, c …., k fall in the category of real numbers and 'n' is non negative integer, which is called the degree of polynomial. An essential characteristic of the polynomial is that each term in the polynomial expression consists of two parts:

* One is the coefficient
* other is the exponent

4. A polynomial can be thought of as an ordered list of non zero terms. Each non zero term is a two-tuple which holds two pieces of information:

* The exponent part
* The coefficient part

**Algorithm :-**

1. **Create circular linked list by including coef,exp,and pointer to next varialble in structure definition.**
2. **For each node ask user to input coef and exp and save them in particular node.**
3. **For displaying polynomial traverse through each node of circular linked list and display in form of (coef)x^(exp)**
4. **For addition traverse through the cll and check the exp first if the exp match then add the coef and**

**Display the resulting polynomial else skip to next node.**

1. **For subtraction traverse through the cll and check the exp first if the exp match then subtract the coef and**

**Display the resulting polynomial skip to next node.**

1. **For multiplication for each node multiply their coef and add their exp.**

**Sourcecode :-**

**#include<iostream>**

**#include<math.h>**

**using namespace std;**

**struct node**

**{**

**int data,data1,data2,data3,data4;**

**struct node \*next;**

**}\*head=NULL,\*perm=head;**

**void create()**

**{**

**struct node \*temp,\*nn;**

**nn=new node;**

**cout<<endl;**

**cout<<"Enter Value of base X and index Y :";**

**cin>>nn->data;**

**cin>>nn->data1;**

**nn->next=head;**

**temp=head;**

**if(head==NULL)**

**{**

**head=nn;**

**nn->next=head;**

**}**

**else**

**{**

**while(temp->next!=head)**

**{**

**temp=temp->next;**

**}**

**temp->next=nn;**

**}**

**}**

**void disp()**

**{**

**struct node \*temp;**

**temp=head;**

**cout<<"Values of X and Y"<<endl;**

**while(temp->next!=head)**

**{**

**cout<<temp->data<<" "<<temp->data1<<endl;**

**temp=temp->next;**

**}**

**cout<<temp->data<<" "<<temp->data1<<endl;**

**}**

**void disp1()**

**{**

**cout<<"Addition of Polynomial"<<endl;**

**cout<<perm->data3<<endl;**

**}**

**void disp2()**

**{**

**cout<<"Multiplication of Polynomial"<<endl;**

**cout<<perm->data4<<endl;**

**}**

**void insert()**

**{**

**struct node \*nn,\*temp;**

**nn=new node;**

**cout<<"Enter Value of X and Y:";**

**cin>>nn->data;**

**cin>>nn->data1;**

**nn->next=head;**

**nn->data1;**

**temp=head;**

**while(temp->next!=head)**

**{**

**temp=temp->next;**

**}**

**temp->next=nn;**

**cout<<endl;**

**}**

**void add()**

**{**

**struct node \*temp,\*nn;**

**temp=head;**

**perm=head;**

**perm->data3=0;**

**while(temp->next!=head)**

**{**

**temp->data2=pow(temp->data,temp->data1);**

**temp=temp->next;**

**}**

**temp->data2=pow(temp->data,temp->data1);**

**temp=head;**

**perm=head;**

**while(temp->next!=head)**

**{**

**perm->data3=perm->data3+temp->data2;**

**temp=temp->next;**

**}**

**perm->data3=perm->data3+temp->data2;**

**}**

**void mul()**

**{**

**struct node \*temp;**

**temp=head;**

**perm=head;**

**perm->data4=1;**

**while(temp->next!=head)**

**{**

**temp->data2=pow(temp->data,temp->data1);**

**temp=temp->next;**

**}**

**temp->data2=pow(temp->data,temp->data1);**

**temp=head;**

**while(temp->next!=head)**

**{**

**perm->data4=perm->data4\*temp->data2;**

**temp=temp->next;**

**}**

**perm->data4=perm->data4\*temp->data2;**

**}**

**int main()**

**{**

**int i,n,a,b=1;**

**cout<<"Enter number of nodes: ";**

**cin>>n;**

**for(i=0;i<n;i++)**

**{**

**create();**

**}**

**disp();**

**while(b==1)**

**{**

**cout<<"Which operation you want"<<endl;**

**cout<<endl<<"1.Insert"<<endl<<"2.Addition"<<endl<<"3.Multiplication"<<endl;**

**cin>>a;**

**switch(a)**

**{**

**case 1:**

**insert();**

**disp();**

**break;**

**case 2:**

**add();**

**disp1();**

**break;**

**case 3:**

**mul();**

**disp2();**

**break;**

**default:**

**cout<<"Invalid option"<<endl;**

**}**

**cout<<endl<<"You want to repeat again press 1"<<endl<<"Otherwise press 0"<<endl;**

**cin>>b;**

**}**

**return 0;**

**}**

**Output:-**

**nput:**

**No of terms: 3**

**Enter coefficient and exponent: 3 3**

**Enter coefficient and exponent: 2 2**

**Enter coefficient and exponent: 1 1**

**Output:**

**Enter number of nodes: 2**

**Enter Value of base X and index Y :2**

**2**

**Enter Value of base X and index Y :3**

**3**

**Values of X and Y**

**2 2**

**3 3**

**Which operation you want**

**1.Insert**

**2.Addition**

**3.Multiplication**

**2**

**Addition of Polynomial**

**31**

**You want to repeat again press 1**

**Otherwise press 0**

**Conclusion:-CLL is used to implement addition,subtraction and multiplication along with display of polynomials.**