Model Lab-Data Structures

Q-1.Remove duplicates from circular linked list

Implimentation-->:

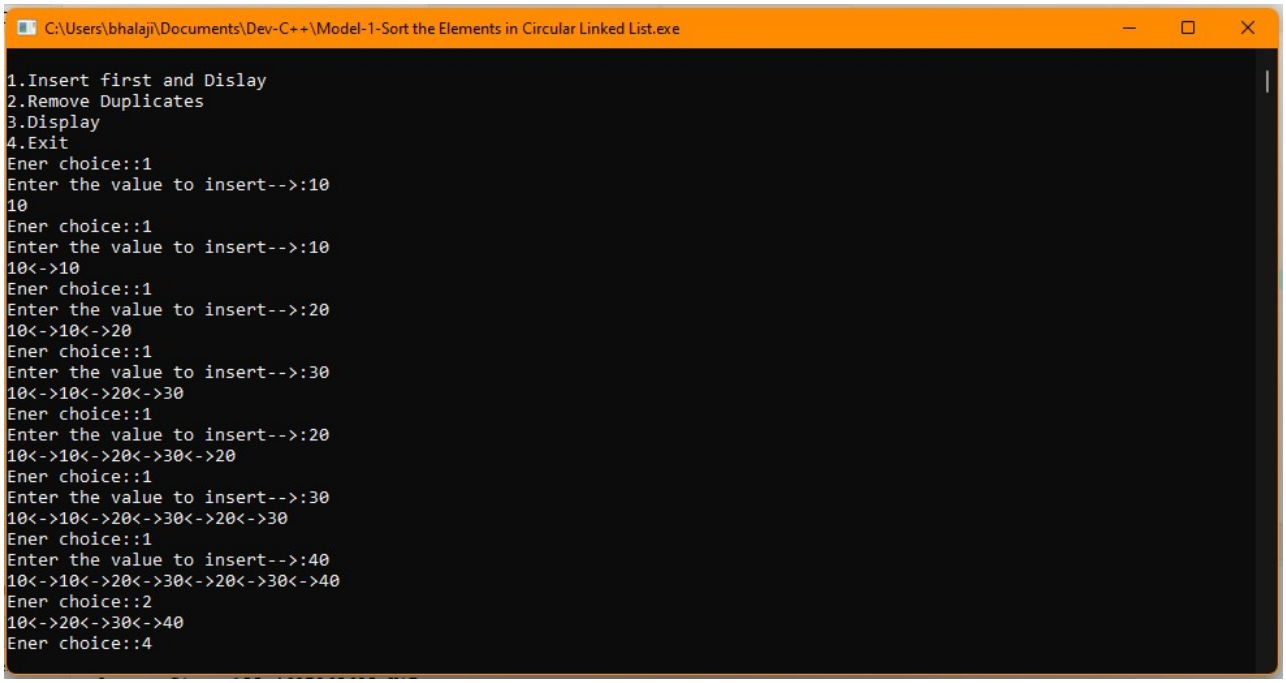
```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct node
{
        struct node *prev;
        int value;
        struct node *next;
}*head=NULL;
void insert()
{
        int data;
        struct node *newnode,*p;
        newnode=(struct node*)malloc(sizeof(struct node));
        printf("Enter the value to insert-->:");
        scanf("%d",&data);
        newnode->value=data;
        newnode->next=NULL;
        newnode->prev=NULL;
        if(head==NULL)
        {
           head=newnode;
           newnode->next=head;
           newnode->prev=head;
```

```
        }
        else
        {
                p=head;
                while(p->next!=head)
                  p=p->next;
                newnode->next=p->next;
                newnode->prev=p;
                p->next=newnode;
        }
}
void display()
{
        struct node *p;
        p=head;
        while(p->next!=head)
        {
                printf("%d<->",p->value);
                p=p->next;
        }
    printf("%d",p->value);
}
void remove_dup()
{
        struct node *p,*q,*temp=NULL,*dummy=NULL;
        p=head;
        while(p->next!=head)
```

```
        {
                q=p->next;
                while(q->next!=head)
                {
                        if(p->value==q->value)
                        {
                                temp=q;
                                dummy=q->next;
                                q->prev->next=q->next;
                                q->next->prev=q->prev;
                                free(temp);
                                q=dummy;
                        }
                        else
                        {
                                q=q->next;
                        }
                }
                p=p->next;
        }
}
int main()
{
        int ch;
        printf("\n1.Insert first and Dislay\n2.Remove Duplicates\n3.Display\n4.Exit");
        while(1)
        {
```

```
            printf("\nEner choice::");
            scanf("%d",&ch);
            switch(ch)
            {
                case (1):
                    insert();
                    display();
                    break;
                case (2):
                    remove_dup();
                    display();
                    break;
                case (3):
                    display();
                    break;
                case (4):
                    exit(0);
            }
        }
        return 0;
}
```

Output-->:



Q-2.Polynomial Addition.

Implimentation-->:

```
#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

struct Node

{

  int coeff;

  int pow;

  struct Node *next;

};

void create_node(int x, int y, struct Node **temp)

{

  struct Node *r, *z;

  z = *temp;
```

```
  if(z == NULL)
  {
     r =(struct Node*)malloc(sizeof(struct Node));
     r->coeff = x;
     r->pow = y;
     r->next = NULL;
     *temp = r;
  }
  else
  {
     r=z;
     while (r->next!=NULL)
     {
       r=r->next;
     }
     r->next = (struct Node*)malloc(sizeof(struct Node));
     r = r->next;
     r->coeff = x;
     r->pow = y;
     r->next = NULL;
  }
}
void polyadd(struct Node *p1, struct Node *p2, struct Node *result)
{
  while(p1!=NULL && p2!=NULL)
  {
   if(p1->pow > p2->pow)
```

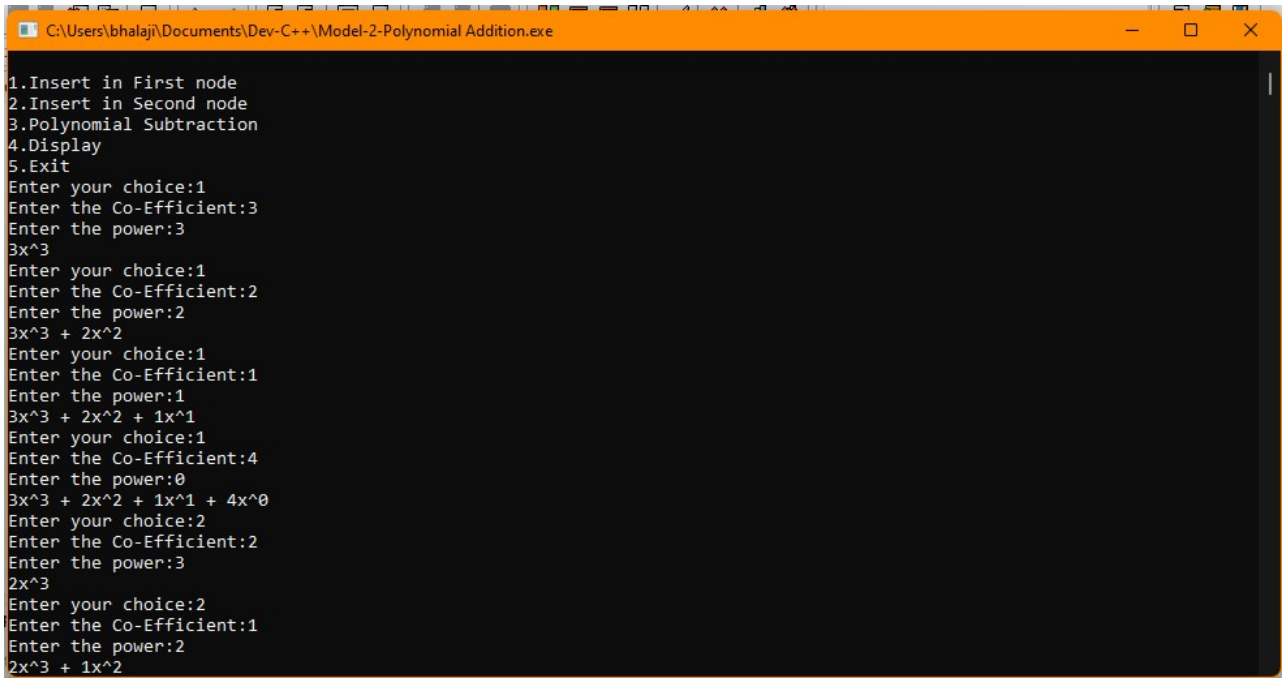
```
        {
    result->pow = p1->pow;
    result->coeff = p1->coeff;
    p1 = p1->next;
  }
  else if(p1->pow < p2->pow)
        {
    result->pow = p2->pow;
    result->coeff = p2->coeff;
    p2 = p2->next;
  }
      else
        {
    result->pow = p1->pow;
    result->coeff = p1->coeff-p2->coeff;
    p1 = p1->next;
    p2 = p2->next;
  }
  result->next = NULL;
  if(p1!=NULL && p2!=NULL)
        {
    result->next = (struct Node *)malloc(sizeof(struct Node));
    result = result->next;
  }
 }
}
void printpoly(struct Node *node)
```

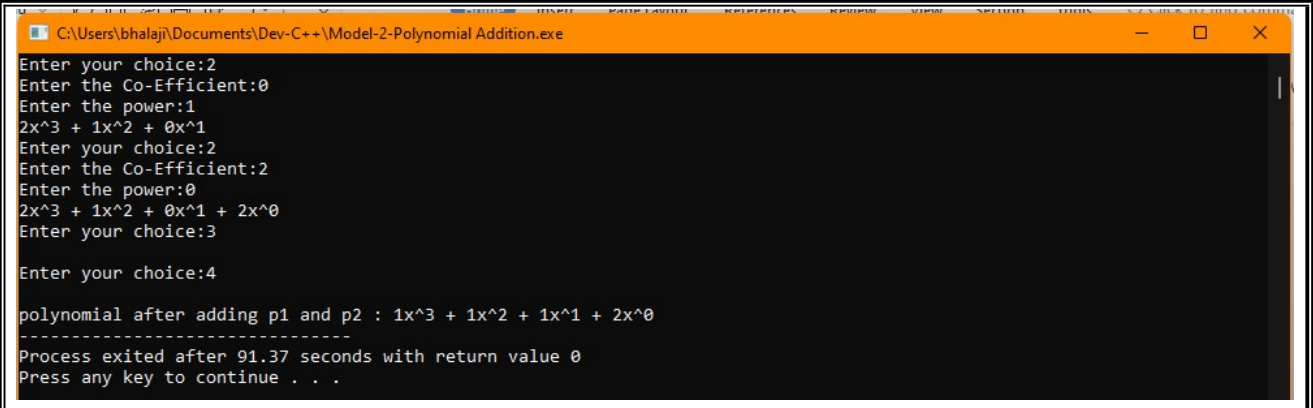
```
{
  if(node==NULL)
      {
      printf("No List");
  }
       else
  {
  while(node->next != NULL)
  {
      printf("%dx^%d", node->coeff, node->pow);
      if(node->next != NULL)
         printf(" + ");
      node = node->next;
  }
   printf("%dx^%d", node->coeff, node->pow);
  }
}
int main()
{
  struct Node *p1 = NULL, *p2 = NULL, *result = NULL;
  int c,coe,po,choice;
  printf("\n1.Insert in First node\n2.Insert in Second node\n3.Polynomial
Subtraction\n4.Display\n5.Exit");
  while(1)
  {
      printf("\nEnter your choice:");
      scanf("%d",&c);
```

```
switch (c)
{
   case 1:
   {
      printf("Enter the Co-Efficient:");
      scanf("%d",&coe);
      printf("Enter the power:");
      scanf("%d",&po);
      create_node(coe,po,&p1);
      printpoly(p1);
      break;
   }
   case 2:
   {
      printf("Enter the Co-Efficient:");
      scanf("%d",&coe);
      printf("Enter the power:");
      scanf("%d",&po);
      create_node(coe,po,&p2);
      printpoly(p2);
      break;
   }
   case 3:
   {
      result = (struct Node *)malloc(sizeof(struct Node));
      polyadd(p1, p2, result);
      break;
```

```
        }
        case 4:
        {
            printf("\npolynomial after adding p1 and p2 : ");
            printpoly(result);
        }
        default:
            exit(0);
    }
  }
  return 0;
}
```

Output-->:




C:\Users\bhalaji\Documents\Dev-C++\Model-2-Polynomial Addition.exe
Enter your choice:2
Enter the Co-Efficient:0
Enter the power:1
2x^3 + 1x^2 + 0x^1
Enter your choice:2
Enter the Co-Efficient:2
Enter the power:0
2x^3 + 1x^2 + 0x^1 + 2x^0
Enter your choice:3
Enter your choice:4
polynomial after adding p1 and p2 : 1x^3 + 1x^2 + 1x^1 + 2x^0
Process exited after 91.37 seconds with return value 0
Press any key to continue . . .