1/12/2020

Addition of two polynomials :-

```
#include <stdio.h>
#include <stdlib.h>
#include < math.h>
struct node {
        float cf;
        float px;
        float py;
        int flag;
        struct node* link;
      };
      typedef struct node * NODE;
      NODE getnode ()
      {
        NODE x;
        x= (NODE) malloc (size of (struct node));
        if (x==NULL)
        {
          printf ("out of memory");
           exit (0);
        }
        return x;
       }
       NODE insert_rear (float cf, float x, float y, NODE head)
       {
         NODE temp, cur;
         int flag;
         temp=getnode();
         temp->cf=cf;
         temp->py = y;
         temp->px= x;
```

```
tenp->flag=0;
cue=head->link;
while (cue->link != head)
 cue = cue->link;
  cue-> link=temp;
  temp->link = head;
  return head;
}

NODE read-poly(NODE head){
  int i;
  float cf,px,py;
  printf(" Enter the coefficient as -999 to end the polynomial\n");
  for (i= 1; ;i++){
   printf(" enter the %d term\n", i);
   printf(" coeff :\n");
   if (cf== -999)
    break;
    printf(" pow x:\n");
    scanf ("%f", &px);
    printf(" pow y:\n");
    scanf(" %f", &py);
    head = insert_rear(cf,px,py,head);
  }
  return head;
 }
  void display (NODE head) {
   NODE temp;
   if (head->link==head)
   {
     printf(" polynomial does not exist \n");
     return;
   }
```

```
temp= head ->link;
while ( temp!= head)

{
 printf ("%5.2fx^%3.1fy ^%3.1f\t", temp->cf, temp->px, temp->py);
   temp=temp->link;
}
printf ("\n");
}
NODE add- poly(NODE h1, NODE h2, NODE h3) {
 NODE p1,p2;
 int x1, x2, y2, cf1, cf2, cf;
 p1= h1->link;
 while (p1!= h1) {
   x1= p1->px;
   y1= p1->py;
   cf1= p1->cf;
   p2= h2->link;
   while (p2!= h2)
   {
   x2=p2->px;
   y2= p2->py;
   cf2 =p2->cf;
   if (x1== x2 &&py1==y2)
    break;
    p2= p2->link;
   }
   if(p2!= h2){
    cf =cf1+cf2;
     p2 -> flag = 1;
     if (cf!=0)
    h3= insert_rear(cf, x1,y1, h3);
    }
    else
   h3= insert_rear (cf1, x1,y1, h3);
   p1= p1 ->link;
}
p2 = h2->link;
while (p2 != h2)
{
  if (p2->flag == 0)
  {
   h3 = insert_rear(p2->cf, p2->px, p2->py, h3);
  }
   p2 = p2->link;
  }
  return h3;
 }
 int main()
 {
 NODE h1,h2,h3;
  h1 = getnode();
  h2 = getnode();
  h3 = getnode();
   h1->link = h1;
   h2->link = h2;
   h3->link = h3;
   printf("Enter the first polynomial\n");
   h1 = read_poly(h1);
   printf("Enter the second polynomial\n");
    h2 = read_poy(h2);
    h3 = add_poly(h1,h2,h3);
    printf("the first polynomial\n");
    display(h1);
    printf(" the second polynomial \n");
    display(h2);
    printf("the sum of the polynomials\n");
    display(h3);
    return 0;
  }
```

9) Evaluation of polynomial :

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
struct node
{
  float cf;
  float px;
  float py;
  struct node * link;
};
typedef struct node * NODE;
NODE getnode()
{
  NODE x;
  x = (NODE) malloc (sizeof (structnode));
  if (x==NULL)
  {
    printf(" Memory full\n");
    exit(0);
  }
  return x;
}
NODE insert_rear (float cf, float x, float y, NODE first)
{
  NODE temp, cur;
  temp= getnode();
  temp->cf = cf;
  temp->px = x;
  temp->py = y;
  temp->link = NULL;
  if (first==NULL)
  {
    return temp;
  }
  cur first;
  while (cur->link!=NULL)
  {
    cur = cur->link;
  }
  cur->link = temp;
  return first;
}
```

Scanned with CamScanner

```
NODE read-poly(NODE first)
{
  int i;
  float cf,px,py;
  printf("Enter -999 to end the polynomial :\n");
  for(i=1;;i++)
  {
    printf("Enter %d term :\n",i);
    printf("Coefficient :\n");
    scanf("%f" &cf);
    if (cf == -999)
    {
      break;
    }
    printf("Power of x :\n");
    scanf("%f" &px);
    printf("Power of y :\n");
    scanf("%f",&py);
    first=insert-rear(cf,px,py,first);
  }
  return first;
}
float evaluate-polynomial(NODE first)
{
  float x,y, sum=0;
  NODE polynomial;
  printf("Enter the value of x and y :\n");
  scanf("%f%f",&x,&y);
  polynomial = first;
  while(polynomial != NULL)
  {
    sum = sum+ polynomial->cf *pow(x,polynomial->px)
        * pow(y,polynomial->py);
    polynomial = polynomial ->link;
  }
```