Week-12 Practice Program

PAGE No.
DATE / /

1) Addition of two long integers.

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct node
{
int info;
struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
{
printf("OUT OF MEMORY");
exit(0);
}
return x;
}
NODE insert_front(NODE first,int data)
{
NODE temp=getnode();
temp->link=first;
temp->info=data;
return temp;
}
```

```
NODE extract(char *S, NODE head)
{
int n;
for(int i=0; i<strlen(S); i++)
  {
   n = S[i]-'0';
   head=insert_front(head,n);
  }
return head;
}
NODE add_long(NODE head1, NODE head2, NODE head3)
{ int temp;
  int carry=0;
  int sum;
NODE cur1=head1;
NODE cur2=head2;
while(cur1!=NULL && cur!=NULL)
   {
   temp=cur1->info+cur2->info+carry;
  if(temp>9)
   {
   sum=temp%10;
    carry=temp/10;
   }
  else
  {
  sum=temp;
   carry=0;
  }
head3=insert_front(head3, sum);
cur1=cur1->link;
cur2=cur2->link;
```

```
}
while(curl!=NULL)
{
temp=curl->info+carry;
if(temp>=9)
  {
    sum=temp%10;
    carry=temp/10;
  }
else
  {
  sum=temp;
  carry=0;
  }
head3=insert_front(head3,sum);
curl=curl->link;
}
while(curr2!=NULL)
  {
temp=curr2->info+carry;
if(temp>9)
  {
  sum=temp%10;
  carry=temp/10
}
else
  {
sum=temp;
carry=0;
}
head3=insert_front(head3,sum);
curr2=curr2->link;
```

```
}
if(cur1==NULL && cur2==NULL)
    {
     if(carry==1)
     head3=insert_front(head3, carry);
     return head3;
    }}

void display(NODE first)
 {
  NODE cur;
  if(first==NULL)
   {
    printf("The list is empty");
      return;
   }
  else
   {
    cur=first;
  while(cur!=NULL)
   {
    printf("%d",cur->info);
    cur=cur->link;
   }
  }}
int main()
 {
  NODE head1 = NULL;
  NODE head2 = NULL;
  NODE head3 = NULL;
  char s1[30],s2[30];
```

DATE / /

```
printf("\n Enter the first integer:");
scanf("%s",s1);
printf(":
head1 = extract(s1, head1);
display(head1);
printf("\n Enter the second integer:");
scanf("%s",s2);
head2 = extract(s2, head2);
display(head2);
head3 = add_long(head1, head2, head3);
printf("\n the long result is:");
display(head3);
return 0;
}
```

PAGE No.
DATE / /

Q. WAP to evaluate a polynomial.

```
CODE: #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(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;
        temp=getnode();
        temp->px=x;
        temp->py=y;
        temp->cf=cf;
       cur=head->link;
```

```
while(cur->link!=head)
{
cur=cur->link;
}
cur->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");
for(int i=0;;i++)
{
printf("Enter the %d term",i+1);
printf("coefficient:");
scanf("%f",&cf);
if(cf==-999)
break;
printf("Power of x:");
scanf("%f",&px);
printf("Power of y:");
scanf("%f",&py);
head=insert_rear(cf,px,py,head);
}
return head;
}
float evaluate(NODE head)
{
```

```
    NODE poly;
printf("Enter value of x:");
  Scanf("%lf", &x);
  printf("Enter the value of y:");
  Scanf("%lf", &y);
  poly = head -> link;
while (poly != head)
  {
  sum = sum + poly -> cf * pow(x, poly -> px) *
                                pow(y, poly -> py);
    poly = poly -> link;
  }
  return sum;
}
void display(NODE head)
  {
  if (head -> link == head)
    {
    printf("Polynomial doesn't exist");
    return;
  }
NODE temp = head -> link;
while (temp -> link != head)
  {
  printf("%.2fx^%.1fy^%.1f + ", temp -> cf, temp ->
                                px, temp -> py);
    temp = temp -> link;
  }
printf("(%5.2fx^%3.1fy^%3.1f)", temp -> cf,
  printf("\n")                  temp -> px, temp -> py);
}
```

```
int main()
{
  node head;
  head = getnode();
  head->link = head;
  float res;
  printf("\nEnter the polynomial\n");
    head = read_poly(head);
  res = evaluate(head);
  printf("\nThe given polynomial is:\n");
    display(head);
  printf("\nThe result is: %.2f\n", res);
  return 0;
}
```

PAGE No.
DATE / /

Q. WAP to add 2 polynomials

```
#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=(NODE)malloc(sizeof(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;
 NODE cur
 temp=getnode();
   temp->cf=cf;
   temp->px=x;
   temp->link=NULL;
   temp->py=y;
```

```
temp->flag=0;
if(head==NULL)
  {
    return temp;
  }
cur=head;
while(cur->link!=NULL)
  {
   cur=cur->link;
  }
cur->link=temp;
return head;
}
NODE read_poly(NODE head)
{float cf,px,py;
printf("Enter -999 to end the polynomial:");
for(int i=1;;i++)
{
printf("Enter the %d term\n",i);
printf("Coefficient:");
scanf("%f",&cf);
if(cf==-999)
  {
   break;
  }
printf("power of x:");
scanf("%f",&px);
printf("power of y");
 scanf("%f",&py);
 head=insertrear(cf,px,py,head);
}
return head;
}
```

```
void display(NODE head)
{
 NODE temp;
 if(head == NULL)
    {
  printf("Polynomial doesn't exist\n");
    }
  else
  {
    temp=head;
    while (temp->link!=NULL)
      {
  printf("(%5.2fx^%3.2fy^%3.2f)+",
     temp->cf, temp->px, temp->py);
    temp=temp->link;
      }
     printf("(%5.2fx^%3.2fy^%3.2f)", temp->
                          temp->px, temp->py);
  }
}
NODE add_poly(NODE h1, NODE h2, NODE h3)
{ NODE p1;
  NODE p2;
  float x1, x2;
  float y1, y2;
  float cf1, cf2, cf3;
    p1=h1;
    while (p1!=NULL)
      {
      x1=p1->px;
      y1=p1->py;
      cf1=p1->cf;
```

```
p2=h2;
while(p2!=NULL)
{
  x2=p2->px;
  y2=p2->py;
  cf2=c->cf;
  if(x1==x2&&y1==y2)
  {
   break;
  }
  p2=p2->link;
}
if(p2!=NULL)
{
  x2=p2->px;
  y2=p2->py;
  cf3=p1->cf2;
  p2->flag=1;
  if(c!=3!=0)
  {
   h3=insert_rear(cf3,x1,y1,h3);
  }
}
else
{
h3=insert_rear(cf1,x1,y1,h3);
}
p1=p1->link;
}
p2=h2;
while(p2!=NULL)
{
```

```
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=NULL;
h2=NULL;
h3=NULL;
printf("Enter the first polynomial\n");
h1=read_poly(h1);
printf("Enter the second polynomial\n");
h2=read_poly(h2);
h3=add_poly(h1,h2,h3);
printf("The first polynomial is:\n");
display(h1);
printf("The second polynomial is:\n");
display(h2);
printf("\nThe sum of two polynomials is:\n");
display(h3);
return 0;
}
```