WEEK 12 APPLICATION OF LL (ADDITION OF TWO LONG INTEGERS, EVALUATION OF POLYNOMIAL EXPRESSION, ADDITION OF POLYNOMIALS)- EXECUTION

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&&cur2!=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;
```

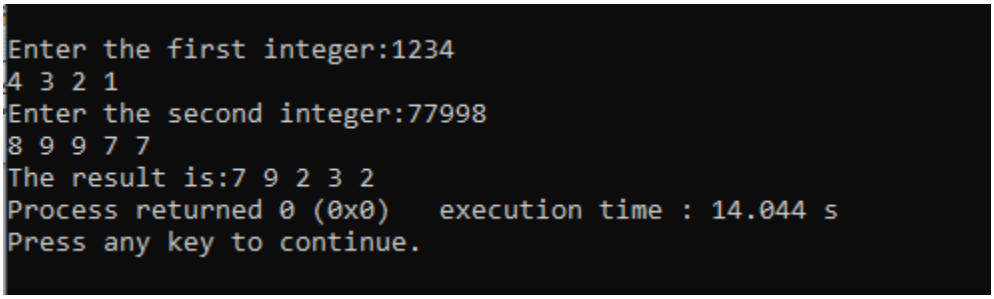
```
}
while(cur1!=NULL)
{
   temp=cur1->info+carry;
   if(temp>9)
   {
      sum=temp%10;
      carry=temp/10;
   }
   else
   {
   sum=temp;
   carry=0;
   }
   head3=insert_front(head3,sum);
   cur1=cur1->link;
}
while(cur2!=NULL)
{
   temp=cur2->info+carry;
   if(temp>9)
   {
      sum=temp%10;
      carry=temp/10;
   }
   else
   {
      sum=temp;
      carry=0;
   }
   head3=insert_front(head3,sum);
   cur2=cur2->link;
}
```

```
 if(cur1==NULL&&cur2==NULL)//once both have reached end of list
   {
        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];
   printf("\nEnter the first integer:");
   scanf("%s",s1);
   head1=extract(s1,head1);
```

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

OUTPUT:



2. EVALUATION OF POLYNOMIALS

```
#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 polynomial\n");
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)
{
    float x,y,sum=0;
    NODE poly;
    printf("Enter the value of x:");
    scanf("%f",&x);
     printf("Enter the value of y:");
    scanf("%f",&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 doesnt exist");
      return;
   }
   NODE temp=head->link;
   while(temp->link!=head)
   {
   printf("(%5.2fx^%3.1fy^%3.1f)+",temp->cf,temp->px,temp->py);
  temp=temp->link;
   }
      printf("(%5.2fx^%3.1fy^%3.1f)",temp->cf,temp->px,temp->py);//to avoid
plus in last term
   printf("\n");
}
int main()
{
   NODE head;
   head=getnode();
```

```
    head->link=head;
    float res;
    printf("Enter the polynomial\n");
    head=read_poly(head);
    res=evaluate(head);
    printf("The given Polynomial is:\n");
    display(head);
    printf("The result is:%.2f\n",res);



    return 0;
}
```

OUTPUT:

```
Enter the polynomial
Enter the coefficient as -999 to end polynomial
Enter the 1 term:coefficient:1
Power of x:2
Power of y:3
Enter the 2 term:coefficient:4
Power of x:3
Power of y:5
Enter the 3 term:coefficient:8
Power of x:3
Power of y:1
Enter the 4 term:coefficient:-999
Enter the value of x:2
Enter the value of y:1
The given Polynomial is:
( 1.00x^2.0y^3.0)+( 4.00x^3.0y^5.0)+( 8.00x^3.0y^1.0)
The result is:100.00

Process returned 0 (0x0)   execution time : 17.718 s
Press any key to continue.
```

3. 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=(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:",&py);
        scanf("%f",&py);
        head=insert_rear(cf,px,py,head);
    }
    return head;
}
void display(NODE head)
{
    NODE temp;
 if(head==NULL)
 {
     printf("Polynomial doesnt 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->cf,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=p2->cf;
    if(x1==x2&&y1==y2)
    {
       break;//it will go out of the loop
    }
    p2=p2->link;
 }
 if(p2!=NULL)//when it comes out of loop when power of x and y are equal and the node p2 is not null
 {
   cf3=cf1+cf2;
   p2->flag=1;
   if(cf3!=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("\nThe second polynomial is:\n");
  display(h2);
  printf("\nThe sum of the two polynomials is:\n");
  display(h3);
   return 0;
}
```

OUTPUT:

```
Enter the first polynomial:
Enter -999 to end the polynomial:Enter the 1 term
Coefficient:1
power of x:2
power of y:3
Enter the 2 term
Coefficient:4
power of x:3
power of y:3
Enter the 3 term
Coefficient:-999
Enter the second polynomial:
Enter -999 to end the polynomial:Enter the 1 term
Coefficient:5
power of x:2
power of y:3
Enter the 2 term
Coefficient:5
power of x:1
power of y:1
Enter the 3 term
Coefficient:-999
The first polynomial is:
( 1.00x^2.00y^3.00)+( 4.00x^3.00y^3.00)
The second polynomial is:
( 5.00x^2.00y^3.00)+( 5.00x^1.00y^1.00)
The sum of the two polynomials is:
( 6.00x^2.00y^3.00)+( 4.00x^3.00y^3.00)+( 5.00x^1.00y^1.00)
Process returned 0 (0x0)   execution time : 27.079 s
Press any key to continue.
```