Program 1

Develop a Program in C for the following:

a. Declare a calendar as an array of 7 elements (A dynamically Created array) to represent 7 days of a week. Each Element of the array is a structure having three fields. The first field is the name of the Day (A dynamically allocated String), The second field is the date of the Day (A integer), the third field is the description of the activity for a particular day (A dynamically allocated String).

b. Write functions create(), read() and display(); to create the calendar, to read the data from the keyboard and to print weeks activity details report on screen.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
struct Day
{
   char *name;
   int date;
   char *activity;
};
struct Day create()
{
   struct Day day;
   day.name = (char *)malloc(20 * sizeof(char));
   day.activity = (char *)malloc(100 * sizeof(char));
   printf("Enter the day name: ");
   scanf("%s", day.name);
   printf("Enter the date: ");
   scanf("%d", &day.date);
   printf("Enter the activity for the day: ");
   scanf(" %s", day.activity);
   return day;
}
void read(struct Day calendar[], int size)
{
   for (int i = 0; i < size; i++)
   {
      calendar[i] = create();
   }
}



void display(struct Day calendar[], int size)
{
   printf("\nWeekly Activity Details:\n");
     for (int i = 0; i < size; i++)
    {
      printf("Day %d: %s\n", i + 1, calendar[i].name);
      printf("Date: %d\n", calendar[i].date);
      printf("Activity: %s\n", calendar[i].activity);
```

```
        printf("\n");
    }
}
int main()

{
    int weekSize = 7;
    struct Day calendar[weekSize];
    read(calendar, weekSize);
    display(calendar, weekSize);
    for (int i = 0; i < weekSize; i++)
    {
        free(calendar[i].name);
        free(calendar[i].activity);
    }
    return 0;
}
```

Sample Output

```
Enter the day name: Monday
Enter the date: 161023
Enter the activity for the day: Reading
Enter the day name: Tuesday
Enter the date: 171023
Enter the activity for the day: Coding
Enter the day name: Wednesday
Enter the date: 181023
Enter the activity for the day: Painting
Enter the day name: Thursday
Enter the date: 191023
Enter the activity for the day: playing
Enter the day name: Friday
Enter the date: 201023
Enter the activity for the day: shopping
Enter the day name: Saturday
Enter the date: 211023
Enter the activity for the day: Watching Movie
Enter the day name: Sunday
Enter the date: 221023
Enter the activity for the day: Completing Assignments

Weekly Activity Details:
Day 1: Monday
Date: 161023Activity: Reading

Day 2: Tuesday
Date: 171023
Activity: Coding

Day 3: Wednesday
Date: 181023
```

Activity: Painting

Day 4: Thursday
Date: 191023
Activity: playing

Day 5: Friday
Date: 201023
Activity: shopping

Day 6: Saturday
Date: 211023
Activity: Watching Movie

Day 7: Sunday
Date: 221023
Activity: Completing Assignments

Program 2

Design, Develop and Implement a program in C for the following operations on Strings

a. Read a Main String (STR), a Pattern String (PAT) and a Replace String (REP).
b. Perform Pattern Matching Operation: Find and Replace all occurrences of PAT in STR with REP if PATexists in STR. Repost suitable messages in case PAT does not exist in STR.

```
#include<stdlib.h>
#include<stdio.h>
char str[100], pat[50], rep[50], ans[100];
int i, j, c, m, k, flag=0;
void stringmatch()
    {
       i = m = c = j = 0;
        while(str[c]!='\0')
          {
                if(str[m]==pat[i])       //       matching
                  {
                     i++; m++;
                      if(pat[i] == '\0')  //       found occurrences.
                          {
                              flag = 1;
                               for(k = 0; rep[k] != '\0'; k++, j++)
                               ans[j] = rep[k];
                               i = 0;
                               c = m;
                           }
                   }
                 else
                    {
                         ans[j] = str[c];
                          j++; c++;
                           m = c; i = 0;
                     }
           }
    }
```

```
int main()
{
        printf("\nEnter a main string \n");
        fgets(str ,sizeof(str),stdin);
        printf("\nEnter a pattern string \n");
        fgets(pat,sizeof(pat),stdin);
        printf("\nEnter a replace string \n");
        fgets(rep,sizeof(rep),stdin);
        stringmatch();
        if(flag == 1)
        printf("\nThe resultant string is\n %s" , ans);
        else
        printf("\nPattern string NOT found\n");
  }
```

Output:

Enter a main string

data structures

Enter a pattern string

data structures

Enter a replace string

data structures with c

The resultant string is

data structures with c

Program 3

Design, Develop and Implement a menu driven program in C for the following operations on **STACK** ofintegers (Array implementation of stack with maximum size **MAX**)

a. Push an element on to stack
b. Pop an element from stack.
c. Demonstrate how stack can be used to check palindrome.
d. Demonstrate Overflow and Underflow situations on stack.
e. Display the status of stack.
f. Exit.

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#define max 3
int st[max], top=-1;
void push(int item)
   {
        if(top==max-1)
        {
             printf("Stack overflow\n");
             return ;
        }
         st[++top]=item;
    }
int pop()
  {
     if(top==-1)
        {
           printf("Stack underflow\n");
           return 0;
        }
         return(st[top--]);
   }
  void palin()
  {
        int i, len, count=0;
        char p[100];
        top=-1;
        printf("Enter a string\n");
        scanf("%s",p);
        len=strlen(p);
         for(i=0;i<len;i++)
            {
```

```
                push(p[i]);
            }
           for(i=0;i<len;i++)
             {
                 if(p[i]==pop())
                   {
                       count++;
                    }
               }
            if(len==count)
               {
                  printf("the string is palindrome\n");
               }
            else
               {
                  printf("the string is nor palindrome\n");
               }
         }
void disp()
{
    int i;
    if(top==-1)
      {
           printf("Stack Empty\n");
          return;
      }
        printf("the stack contents are");
        for(i=top;i>=0;i--)
        printf("|%d|\n",st[i]);
   }
void main()
{
    int ch,k,item;
     while(1)
          {
              printf("MAIN MENU\n");
              printf(" 1:Push\n 2:Pop\n 3:Display\n 4:Palindrome\n 5:Exit\n");
              printf("Enter your choice\n");
              scanf("%d",&ch);
              switch(ch)
                   {
                       case 1:printf("Enter an item to push\n");
                              scanf("%d",&item);
                              push(item);
                               break;
                       case 2: k=pop();
```

```
                    if(k)
                    printf("popped element is %d\n",k);
                     break;
                case 3: disp();
                     break;
                  case 4:palin();
                     break;
                 case 5:exit(0);
               }
          }
    }
```

Output

```
MAIN MENU
 1:Push
 2:Pop
 3:Display
 4:Palindrome
 5:Exit
Enter your choice
1
Enter an item to push
1
MAIN MENU
 1:Push
 2:Pop
 3:Display
 4:Palindrome
 5:Exit
Enter your choice
1
Enter an item to push
2
MAIN MENU
 1:Push
 2:Pop
 3:Display
 4:Palindrome
 5:Exit
Enter your choice
1
Enter an item to push
3
MAIN MENU
 1:Push
 2:Pop
 3:Display
```

```
 4:Palindrome
 5:Exit
Enter your choice
1
Enter an item to push
4
Stack overflow
MAIN MENU
 1:Push
 2:Pop
 3:Display
 4:Palindrome
 5:Exit
Enter your choice
3
the stack contents are
|3|
|2|
|1|
MAIN MENU
 1:Push
 2:Pop
 3:Display
 4:Palindrome
 5:Exit
Enter your choice
2
popped element is 3
MAIN MENU
 1:Push
 2:Pop
 3:Display
 4:Palindrome
 5:Exit
Enter your choice
2
popped element is 2
MAIN MENU
 1:Push
 2:Pop
 3:Display
 4:Palindrome
 5:Exit
Enter your choice
2
popped element is 1
```

```
MAIN MENU
 1:Push
 2:Pop
 3:Display
 4:Palindrome
 5:Exit
Enter your choice
2
Stack underflow
MAIN MENU
 1:Push
 2:Pop
 3:Display
 4:Palindrome
 5:Exit
Enter your choice
```

Program 4

Develop a Program in C for converting an Infix Expression to Postfix Expression. Program should support for both parenthesized and free parenthesized expressions with the operators: +, -, *, /, % (Remainder), ^ (Power) and alphanumeric operands

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
int F(char symbol)
{
        switch(symbol)
        {
                case '+':
                case '-':   return 2;
                case '*':
                case '/':   return 4;
                case '^':
                case '$':   return 5;
                case '(':   return 0;
                case '#':     return -1;
                default: return 8;
       }
}
int G(char symbol)
   {
        switch(symbol)
        {
                case '+':
                case '-':    return 1;
                case '*':
                case '/':    return 3;
                case '^':
                case '$':   return 6;
                case '(':    return 9;
                case ')':    return 0;
                default:     return 7;
               }
        }
void infix_postfix(char infix[], char postfix[])
{
       int top, j, i;
       char s[30], symbol; top = -1;
       s[++top] = '#';
        j = 0;
```

```
    for(i=0; i < strlen(infix); i++)
        {
            symbol = infix[i];
            while(F(s[top]) > G(symbol))
                {
                    postfix[j] = s[top--]; j++;
                }
                if(F(s[top]) != G(symbol))
                        s[++top] = symbol;
                 else
                        top--;
            }
      while(s[top] != '#')
        {
            postfix[j++] = s[top--];
        }
        postfix[j] = '\0';
    }

void main()
{
    char infix[20], postfix[20];
    printf("\nEnter a valid infix expression\n");
    scanf("%s",infix);
    infix_postfix(infix,postfix);
    printf("\nThe infix expression is:\n");
    printf ("%s",infix);
    printf("\nThe postfix expression is:\n");
    printf ("%s",postfix);
}
```

```
Enter a valid infix expression
(a+b)*c
The infix expression is:
(a+b)*c
The postfix expression is:
ab+c*
```

Program 5

Design, Develop and Implement a Program in C for the following Stack Applications

a. Evaluation of **Suffix expression** with single digit operands and operators: **+, -, *, /, %, ^**

b. Solving **Tower of Hanoi** problem with **n** disks.

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<string.h>
double compute(char symbol, double op1, double op2)
{
        switch(symbol)
        {
              case '+':  return op1 + op2;
             case '-':  return op1 - op2;
              case '*':  return op1 * op2;
             case '/':  return op1 / op2;
             case '$':
             case '^':  return pow(op1,op2);
            default:   return 0;
        }
}
void main()
{
         double s[20], res, op1, op2;
         int top, i;
         char postfix[20], symbol;
         printf("\nEnter the postfix expression:\n");
         scanf("%s",postfix);
         top=-1;
         for(i=0; i<strlen(postfix); i++)
               {
                     symbol = postfix[i];
                     if(isdigit(symbol))
                           s[++top] = symbol-'0';
                    else
                      {
                           op2 = s[top--];
                           op1 = s[top--];
                           res = compute(symbol, op1, op2);
                            s[++top] = res;
                      }
                }
```

```
                res = s[top--];
                printf("\nThe result is : %f\n", res);
    }
```

Sample Output

```
Enter the postfix expression:
22^3*
The result is : 12.000000
```

```
Enter the postfix expression:
123+*321-+*
The result is : 20.000000
```

b.

```
#include<stdio.h>
void tower(int n, int source, int temp, int destination)
{
        if(n == 0)
        return;
        tower(n-1, source, destination, temp);
        printf("\nMove disc %d from %c to %c", n, source, destination);
        tower(n-1, temp, source, destination);
}
void main()
{
       int n;
        printf("\nEnter the number of discs: \n");
        scanf("%d", &n);
        tower(n, 'A', 'B', 'C');
        printf("\n\nTotal Number of moves are: %d", (int)pow(2,n)-1);
}
```

Sample Output

```
Enter the number of discs:
3
Move disc 1 from A to C
Move disc 2 from A to B
Move disc 1 from C to B
Move disc 3 from A to C
Move disc 1 from B to A
Move disc 2 from B to C
Move disc 1 from A to C

Total Number of moves are: 7
```

Program 6

Design, Develop and Implement a menu driven Program in C for the following operations on **CircularQUEUE** of Characters (Array Implementation of Queue with maximum size **MAX**)

a. Insert an Element on to Circular QUEUE
b. Delete an Element from Circular QUEUE
c. Demonstrate ***Overflow*** and ***Underflow*** situations on Circular QUEUE
d. Display the status of Circular QUEUE
e. Exit

```
#include<stdio.h>
#include<stdlib.h>
#define MAX 4
int ch, front = 0, rear = -1, count=0;
char q[MAX], c;
void insert(char c)
{
    if(count == MAX)
        {
            printf("\nQueue is Full");
            return;
        }
            rear = (rear + 1) % MAX;
            q[rear]=c;
            count++;
}
void del()
{
        if(count == 0)
        {
            printf("\nQueue is Empty");
            return;
        }
           c=q[front];
           printf("Deleted item is: %c", c);
             front = (front + 1) % MAX;
            count--;
      }
 void display()
{
```

```
        int i ;
        if(count == 0)
        {
           printf("\nQueue is Empty");
           return;
    }
                        printf("\nContents of Queue is:\n");
                        for(i=1; i<=count; i++)
                        {
                                printf("%c\t", q[front]);
                                      front = (front + 1) % MAX;
                        }
        }

void main()
{
 while(1)
{
        int ch;
        printf("\n1.Insert\n 2.Delete\n 3.Display\n 4.Exit\n");
          printf("Enter the choice\n");
        scanf("%d",&ch);
        switch(ch)
{
   case 1: printf("Enter the character");
              scanf("%s",&c);
              insert(c);
             break;
   case 2: del();
              break;
   case 3: display();
             break;
   case 4: exit(0);
}
}
}
```

Output:

1.Insert

2.Delete

3.Display
4.Exit
Enter the choice
1
Enter the character
A
1.Insert
2.Delete
3.Display
4.Exit
Enter the choice
1
Enter the character
B
1.Insert
2.Delete
3.Display
4.Exit
Enter the choice
1
Enter the character
C
1.Insert
2.Delete
3.Display
4.Exit
Enter the choice
1
Enter the character
D
1.Insert
2.Delete
3.Display
4.Exit
Enter the choice
1
Enter the character
E
Queue is Full

1.Insert
2.Delete
3.Display
4.Exit
Enter the choice
3
Contents of Queue is:
A B C D
1.Insert
2.Delete
3.Display
4.Exit
Enter the choice
2
Deleted item is: A
1.Insert
2.Delete
3.Display
4.Exit
Enter the choice
2
Deleted item is: B
1.Insert
2.Delete
3.Display
4.Exit
Enter the choice
2
Deleted item is: C
1.Insert
2.Delete
3.Display
4.Exit
Enter the choice
2
Deleted item is: D
1.Insert
2.Delete
3.Display
4.Exit

Enter the choice
2
Queue is Empty
1.Insert
2.Delete
3.Display
4.Exit
Enter the choice

PROGRAM 7

Design, Develop and Implement a menu driven Program in C for the following operations on **Singly Linked List (SLL)** of Student Data with the fields: ***USN, Name, Branch, Sem, PhNo***

a. Create a **SLL** of **N** Students Data by using ***front insertion***.
b. Display the status of **SLL** and count the number of nodes in it
c. Perform Insertion and Deletion at End of **SLL**
d. Perform Insertion and Deletion at Front of **SLL**
e. Demonstrate how this **SLL** can be used as **STACK** and **QUEUE**
f. Exit

```
#include<stdio.h>
#include<string.h>
#define null 0
struct student
  {
     char usn[15],name[20],branch[10];
     int sem;
     char phno[20];
     struct student *link;
   };
      typedef struct student node;
      node *first;
  void main()
   {
     void create(),insert_end(),del_front(),disp();
     int ch;
     while(1)
     {
          printf("Main Menu\n");
          printf("1:Create\n2:Display\n3:Insert Endt\n4:Delete Front\n5:Exit\n");
          printf("Enter your choice\n");
          scanf("%d",&ch);
          switch(ch)
            {

                case 1:create();
                        break;
                case 2:disp();
                       break;
                case 3:insert_end();
                        break;
               case 4:del_front();
                       break;
               case 5:exit(0);
            }
```

```
        }
  }

 void create()
  {
      int i,n; node *p;
      printf("Enter the number of students\n");
      scanf("%d",&n);
      for(i=0;i<n;i++)
         {
            p=(node*)malloc(sizeof(node));
            printf("Enter the student USN , NAME ,BRANCH,SEM,PHNO\n");
    scanf("%s%s%s%d%s", p->usn,p->name,p->branch,&p->sem,p->phno);
            p->link=first;
            first=p;
         }
  }
void disp()
{
   int cnt=0;
   node *t;
   t=first;
  while(t)
   {
       cnt++;
     printf("%s\t%s\t%s\t%d\t%s->\n",t->usn,t->name,t->branch,t->sem,t->phno);
       t=t->link;
    }
            printf("Total number of nodes=%d\n",cnt);
     }
void insert_end()
{
        node *p,*r;
        p=(node*)malloc(sizeof(node));
        printf("Enter the student USN , NAME ,BRANCH,SEM,PHNO\n");
        scanf("%s%s%s%d%s",p->usn,p->name,p->branch,&p->sem,p->phno);
         r=first;
         while(r->link!=null)
          {
             r=r->link;
          }
          r->link=p;
          p->link=null;
}
void del_front()
 {
```

```
    node *q;
     if(first==null)
          {
              printf("List empty\n");
              return;
          }
         q=first;
         printf("Deleted node is %s",q->usn);
         first=first->link;
         free(q);
}
```

Output:

Main Menu
1:Create
2:Display
3:Insert Endt
4:Delete Front
5:Exit
Enter your choice
1
Enter the number of students
2
Enter the student USN , NAME ,BRANCH,SEM,PHNO
1EP22CS001 ravi CSE 3 7845362789
Enter the student USN , NAME ,BRANCH,SEM,PHNO
1EP22CS002 Raju CSE 3 954735678
Main Menu
1:Create
2:Display
3:Insert Endt
4:Delete Front
5:Exit
Enter your choice
3
Enter the student USN , NAME ,BRANCH,SEM,PHNO
1Ep22CS003 Ranga EC 3 9856784934
Main Menu
1:Create
2:Display
3:Insert Endt

4:Delete Front
5:Exit
Enter your choice
2
1EP22CS002 Raju CSE 3 954735678->
1EP22CS001 ravi CSE 3 7845362789->
1Ep22CS003 RangaEC 3 9856784934->
Total number of nodes=3
Main Menu
1:Create
2:Display
3:Insert Endt
4:Delete Front
5:Exit
Enter your choice

Program 8

Design, Develop and Implement a menu driven Program in C for the following operations on **Doubly LinkedList (DLL)** of Employee Data with the fields: ***SSN, Name, Dept, Designation, Sal, PhNo.***

```
#include<stdio.h>
#include<stdlib.h>
#define null 0
struct emp
{
        char name[40],dept[40],desig[40];
        int ssn;
        long int sal;
        char phno[20];
        struct emp *llink;
        struct emp *rlink;
};
typedef struct emp node; node *start;
void create(),insert_front(),del_front(),disp();
void main()
{
         int ch;
         while(1)
         {
            printf("\nMain Menu\n");
            printf("1:Create\n2:Display\n3:Insert_Front\n4:Del_Front\n5:Exit\n");
            printf("Enter your choice\n");
            scanf("%d",&ch);
            switch(ch)
                 {
                        case 1:create();
                                 break;
                        case 2:disp();
                                  break;
                        case 3:insert_front();
                                  break;
                        case 4:del_front();
                                  break;
                        case 5:exit(0);
                  }
```

```
        }
    }
void create()
{
        node *p, *t;
        int i, n;
        printf("Enter the number of employees \n");
        scanf("%d", &n);
        printf("Enter the employee details[SSN,NAME,DEPT,DESIG,SAL AND
        PH.NO.]\n");
        for(i=0; i<n; i++)
        {
                printf("enter the Employee %d details\n",i+1);
                p=(node*)malloc(sizeof(node));
                p->rlink=null;
                 scanf("%d%s%s%s%ld%s",&p->ssn,p->name,p->dept,p-
                 >desig,&p->sal,p->phno);
                if(start==null)
                 {
                    start=p;
                    start->llink=null;
                 }
                 else
                {
                     t=start;
                    while(t->rlink!=null)
                     t=t->rlink;
                    t->rlink=p;
                    p->llink=t;
                 }
            }
      }

void disp()
{
        node *r;
        r=start;
        while(r)
        {
```

```
        printf("|%d|%s|%s|%s|%ld|%s|\n<->",r->ssn,r->name,r->dept,r->desig,r->sal,r->phno);
        r=r->rlink;
        }
}
void insert_front()
{
        node *p; p=(node*)malloc(sizeof(node));
        printf("Enter emp details\n");
        scanf("%d%s%s%s%ld%%s", &p->ssn, p->name,p->dept, p->desig, &p->sal, p->phno);
        p->rlink=start;
        start=p;
        start->llink=null;
}
void del_front()
{
        node *q;
        if(start==null)
        {
                printf("list empty\n");
                return;
        }
        q=start;
        printf("Deleted node is %d",q->ssn);
        start=start->rlink;
        free(q);
}
```

Output

```
Main Menu
1:Create
2:Display
3:Insert_Front
4:Del_Front
5:Exit
Enter your choice
1
Enter the number of employees
```

```
2
Enter the employee details[SSN,NAME,DEPT,DESIG,SAL AND PH.NO.]
enter the Employee 1 details
123 Ravi CSE Assistprof 43000 9845242456
enter the Employee 2 details
345 Ragu EC Assistprof 45000  9845145632
Main Menu
1:Create
2:Display
3:Insert_Front
4:Del_Front
5:Exit
Enter your choice
2
|123|Ravi|CSE|Assistprof|43000|9845242456|
<->|345|Ragu|EC|Assistprof|45000|9845145632|
<->
Main Menu
1:Create
2:Display
3:Insert_Front
4:Del_Front
5:Exit
Enter your choice
3
Enter emp details
456 Ranga ME AssistProf 42000 7865456123
Main Menu
1:Create
2:Display
3:Insert_Front
4:Del_Front
5:Exit
Enter your choice

Main Menu
1:Create
2:Display
3:Insert_Front
4:Del_Front
```

5:Exit
Enter your choice
2
|456|Ranga|ME|AssistProf|42000||
<->|123|Ravi|CSE|Assistprof|43000|9845242456|
<->|345|Ragu|EC|Assistprof|45000|9845145632|
<->
Main Menu
1:Create
2:Display
3:Insert_Front
4:Del_Front
5:Exit
Enter your choice
4
Deleted node is 456
Main Menu
1:Create
2:Display
3:Insert_Front
4:Del_Front
5:Exit
Enter your choice

Program-9

Using circular representation for a polynomial, design, develop, and execute a program in C to accept two polynomials, add them, and then print the resulting polynomial.

```
#include<stdio.h>
#include<stdlib.h>
#define COMPARE(x,y)(((x)==(y))?0:((x)>(y))?1:-1)
struct node
{
        int coeff;
        int expon;
        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 attach(int coeff,int expon,NODE head)
{
        NODE temp, cur;
        temp=getnode();
        temp->coeff=coeff;
        temp->expon=expon;
        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=1;
        int coeff;
        int expon;
```

```
        printf("enter the coefficient as -999 to the end of the polynomial");
        while(1)
        {
                printf("enter the %d term\n",i++);
                printf("Coeff=");
                scanf("%d",&coeff);
                if(coeff==-999) break;
                printf("pow x=");
                scanf("%d",&expon);
                head=attach(coeff,expon,head);
        }
        return head;
}
NODE poly_add(NODE head1,NODE head2,NODE head3)
{
        NODE a,b;
        int coeff;
        a=head1->link;
        b=head2->link;
        while(a!=head1 && b!=head2)
        {
                switch(COMPARE(a->expon,b->expon))
                {
                case 0:  coeff=a->coeff+b->coeff;
                        if(coeff!=0)head3=attach(coeff,a->expon,head3);
                        a=a->link;
                        b=b->link;
                        break;
                case 1:head3=attach(a->coeff,a->expon,head3);
                        a=a->link;
                        break;
                default: head3=attach(b->coeff,b->expon,head3);
                          b=b->link;
                 }
         }
        while(a!=head1)
         {
                head3=attach(a->coeff,a->expon,head3);
                a=a->link;
         }
         while(b!=head2)
        {
                head3=attach(b->coeff,b->expon,head3);
                 b=b->link;
          }
          return head3;
```

```
}
void display(NODE head)
{
    NODE temp;
     if(head->link==head)
            {
                    printf("polynomial doesnot exist");
                    return;
             }
      temp=head->link;
      while(temp!=head)
       {
          if(temp->coeff<0)
          printf("%2dx^%2d",temp->coeff,temp->expon);
          else
          printf("+%2dx^%2d",temp->coeff,temp->expon);
           temp=temp->link;
        }

  }
void main()
{
     NODE head1,head2,head3;
     head1=getnode();
     head2=getnode();
     head3=getnode();
     head1->link=head1;
     head2->link=head2;
     head3->link=head3;
     printf("enter the first polynamial");
     head1=read_poly(head1);
     printf("enter the second polynamial");
     head2=read_poly(head2);
     head3=poly_add(head1,head2,head3);
     printf("polynomial1\n");
     display(head1);
     printf("\npolynomial2\n");
     display(head2);
     printf("\npolynomial3\n");
     display(head3);
    }
```

Output:

```
enter the first polynamialenter the coefficient as -999 to the end of the
polynomialenter the 1 term
Coeff=2
pow x=2
enter the 2 term
Coeff=5
pow x=1
enter the 3 term
Coeff=6
pow x=0
enter the 4 term
Coeff=-999
enter the second polynamialenter the coefficient as -999 to the end of the
polynomialenter the 1 term
Coeff=3
pow x=2
enter the 2 term
Coeff=3
pow x=1
enter the 3 term
Coeff=5
pow x=0
enter the 4 term
Coeff=-999
polynomial1
+ 2x^ 2+ 5x^ 1+ 6x^ 0
polynomial2
+ 3x^ 2+ 3x^ 1+ 5x^ 0
polynomial3
+ 5x^ 2+ 8x^ 1+11x^ 0
```