	PUNE INSTITUTE OF COMPUTER TECHNOLOGY PUNE - 411043	
	Department of Electronics & Telecommunication	
	ASSESMENT YEAR: 2021-2022	CLASS: SE-5
	SUBJECT: DATA STRUCTURES	
EXPT No:	LAB Ref: SE/2021-22/	Starting date: 29/11/2021
	Roll No:22108	Submission date:5/12/2021
Title:	**Polynomial Operations**	
Problem statement	Implement polynomial operations using linked list. Perform addition, subtraction and multiplication on it.	
Prerequisites:	Basics of C programming	
	Decision making and loop controls	
	Choice based program	
	Functions, Linked List	
Objectives:	Learn to create linked list	
	Implement polynomial operation using linked list	
Theory:		

1) **Functions –**

- It is a self-contained block of statements that perform task of some kind.
- C program may have one or more functions
- C program must have at least one function i.e. main()
- There is no limit on number of functions
- Each function is called in a sequence specified by the function calls in main()
- After each function has done its job, control returns to next location from where it has been called..

5) **Linked List –**

- Linked lists are linear data structures, elements here are stored using pointers unlike arrays..
- Dynamic memory allocation is used in linked list, thus making insertion and deletion easier than arrays.
- Linked list has two parts, first is the data and second is the pointer, consider there are 3 nodes (nodes – building blocks of a linked list), in a linked list first node has pointer of second node, second node has pointer of third and third has no pointer data as there is no other node after it.
  The pointer of first node is contained by head variable.
- Initialization: struct Node

```
{
        int data;
        struct Node *ptr;
};
struct Node *head;  (head node initialized)
```

Code

```
#include<stdio.h>
#include<stdlib.h>

struct Node
{     float
coeff;     int
power;
    struct Node *next;
};

void end()
{
    printf("This is the end of execution!");
}

struct Node* insert(struct Node* head, float coeff, int power)
{
    struct Node *temp;
    struct Node *p = (struct Node *)malloc(sizeof(struct Node));

    p -> coeff = coeff;
p -> power = power;
    p -> next = NULL;

    if (head == NULL && power>head->power)
    {
        p -> next = head;
head = p;
    }
else
    {
        temp = head;
        while (temp->next != NULL && temp->next->power >= power)
        {
            temp = temp->next;
        }
        p -> next = temp -> next;
temp -> next = p;
    }
    return head;
}

struct Node* create(struct Node *temp)
{
    int n, coeff, power;
    temp = (struct Node *)malloc(sizeof(temp));
    printf("\nEnter the Coefficient and Power of terms in decending order of power
of x.");
    printf("\nEnter the number of terms in the polynomial: ");
scanf("%d", &n);     for (int i = 0; i < n; i++)
    {
        printf("\nEnter the coefficient of the %d term: ", i+1);
scanf("%f", &coeff);
        printf("\nEnter the power of x of %d term: ", i+1);
scanf("%d", &power);

        temp = insert(temp, coeff, power);
    }
    return temp;
}

void display_data(struct Node *disp)
{
    if (disp == NULL)
    {
        printf("There is no
polynomial!");        end();        return;
    }

    struct Node* temp = disp;

    while (disp!=NULL)
    {
        printf("%dx^%d", disp->coeff, disp->power);
disp = disp->next;
        if (temp !=  NULL)
        {
```

```
        printf("+");
    }
```

```
        else
{
            printf("\n");
        }
    }
}

void add(struct Node* head1, struct Node* head2)
{     struct Node* ptr1 =
head1;     struct Node* ptr2 =
head2;
    struct Node* head3 = NULL;

    while (ptr1 != NULL, ptr2 != NULL)
    {
        if (ptr1->power == ptr2->power)
        {
            head3 = insert(head3, ptr1->coeff+ptr2->coeff, ptr1-
>power);             ptr1 = ptr1->next;             ptr2 = ptr2->next;
        }
        else if (ptr1->power>ptr2->power)
        {
            head3 = insert(head3, ptr1->coeff, ptr1->power);
ptr1 = ptr1->next;
        }
        else if (ptr2->power>ptr1->power)
        {
            head3 = insert(head3, ptr2->coeff, ptr2->power);
ptr2 = ptr2->next;
        }
    }
    while (ptr1!=NULL)
    {
        head3 = insert(head3, ptr1->coeff, ptr1->power);
    }
    while (ptr2!=NULL)
    {
        head3 = insert(head3, ptr2->coeff, ptr2->power);
    }
    printf("The added polynomial is: ");
display_data(head3);
}

void subtract(struct Node* head1, struct Node* head2)
```

```
{     struct Node* ptr1 = head1;
struct Node* ptr2 = head2;
struct Node* head3 = NULL;
```

```
    while (ptr1 != NULL, ptr2 != NULL)
    {
        if (ptr1->power == ptr2->power)
        {
            head3 = insert(head3, ptr1->coeff-ptr2->coeff, ptr1-
>power);             ptr1 = ptr1->next;             ptr2 = ptr2->next;
        }
        else if (ptr1->power>ptr2->power)
        {
            head3 = insert(head3, ptr1->coeff, ptr1->power);
ptr1 = ptr1->next;
        }
        else if (ptr2->power>ptr1->power)
        {
            head3 = insert(head3, (ptr2->coeff)*(-1), ptr2->power);
ptr2 = ptr2->next;
        }
    }
    while (ptr1!=NULL)
    {
        head3 = insert(head3, ptr1->coeff, ptr1->power);
    }
    while (ptr2!=NULL)
    {
        head3 = insert(head3, (ptr2->coeff)*(-1), ptr2->power);
    }
    printf("The subtracted polynomial is: ");
display_data(head3);
}

void multiply(struct Node *head1, struct Node *head2)
{    struct Node* ptr1 =
head1;     struct Node* ptr2 =
head2;
    struct Node* head3 = NULL;

    if (head1 == NULL && head2 == NULL)
    {
        printf("\nZero Polynomial.");
return;
    }

    while (ptr1 != NULL)
    {
```

```
while (ptr2 != NULL)
{
```

```
            head3 = insert(head3, ptr1->coeff*ptr2->coeff, ptr1->power+ptr2-
>power);
            ptr2 = ptr2->next;
        }
        ptr1 = ptr1->next;
ptr2 = head2;
    }
    display_data(head3);
}

int main()
{
    int checker, i, choice;
struct Node *head1 = NULL;
struct Node *head2 = NULL;

    head1 = (struct Node *)malloc(sizeof(struct Node));
head2 = (struct Node *)malloc(sizeof(struct Node));

    printf("Do you want to perform polynomial operation? (y(1)/ n(0)): ");
scanf("%d", &checker);

    if (checker == 0)
    {
end();
return 0;
    }

    printf("\nFirst arrange your polynomials in decending order according to the powers of x, then start entering following data of first polynomial and then second polynomial.\nThen choose the operation to be executed on those two polynomials, mainly Addition, Subraction and Multiplication of polynomials.");

    printf("\nThe first Polynomial:-");
head1 = create(head1);

    printf("\nThe second Polynomial:-");
head2 = create(head2);

    printf("1) Addition\n2) Subtraction\n3) Multiplication\nEnter your choice: ");
scanf("%d", &choice);

    switch (choice)
    {
case 1:
    {
```

```
    add(head1, head2);
    break;
}
```

```
    case 2:
    {
        subtract(head1, head2);
        break;
    }
case 3:
    {
        multiply(head1, head2);
        break;
    }
default:
    {
end();
break;
    }
    }
return 0;
}
```

CONCLUSION:

In this experiment, we are able to implement polynomial operations using SLL.
We applied addition, subtraction and multiplication of polynomials using SLL.

REFERENCES:

Seymour Lipschutz, Data Structure with C, Schaum's Outlines, Tata McGrawHill
E Balgurusamy - Programming in ANSI C, Tata McGraw-Hill (Third Edition)
Yashavant Kanetkar- Let Us C, BPB Publication, 8th Edition.

Continuous Assessment for DS AY 2021-22			
RPP (5)	**SPO (5)**	**Total (10)**	**Signature:**
			Assessed By: Mr. V. B. Vaijapurkar
Start date	**Submission date**		**Date:**
29/11/2021	**5/12/2021**		**Roll. No.22108**
***Regularity, Punctuality, performance** ***Submission, Presentation, orals**			