

Bansilal Ramnath Agarwal Charitable Trust's Vishwakarma Institute of Information Technology

Department of Artificial Intelligence and Data Science

Name: Siddhesh Dilip Khairnar

Class: SY Division: B Roll No: 272028

Semester: III Academic Year: 2022-2023

Subject Name & Code: Data Structure, ADUA21202

Title of Assignment: Implement polynomial using Doubly Linked List and perform

Addition of Polynomials

Assignment No.- 4

	PAGENO.:		
	DS Assignment: 4		
	Name: siddhesh Dilip Khairnar		
	PRN NO: 22110398 PRN No:		
	Roll no: 272028		
	Problem statement: Implement polynamial using soutly linked list?		
	Problem statement: Implement polynamial using pourly linked list & puyonn Addition / Multiplication of polynamial		
	6.0		
	Theory:		
	Doubly linked list: Doubly linked list is a complex type of linked list		
	in which a nocle coulair a pointer to tru previous as well as the next.		
	node is the signerce. Turefore, is a doubly linked list, a node consist		
	of thru part: mode data, pointer to the next rade in sequence, pointer to		
The previous node. A sample node in a doubly linked list is shown in the			
figure:			
-	node		
	head > Prev Data Next		
A doubly linked list containing these node having number from, to 3			
in their data, is shown in the following image			
	[head]		
	X 1 2 3 X		
	In c, structure of a node in doubly linked list can be given as:		
	start noch		
	& struct wall * Prev;		
	int data;		
	Stuct node * next;		
	3		
1			

	PAGENO.:		
	DATE: / /		
	HILLIAN BURNERS		
Conclusion: Thus we have successfully implemented and a suit			
Conclusion: Thus, we have successfully implemented polynomial using Doubly linked lest and performed Addition / Multiplication of			
poyomials.			
	THE RESERVE OF THE PARTY OF THE		
<u></u>			
Laurin			
	INCOME TO A STATE OF THE STATE		
	THE RESERVE AND ADDRESS OF THE PARTY OF THE		
	La Company Colonia		
	The state of the state of		
	23 3 6 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		

Program:

```
2 using namespace std;
 3 struct Node
4 - {
5 int coeff;
6 int pow;
7 struct Node *next;
8 };
9 void create_node(int x, int y, struct Node **temp)
10 - {
struct Node *r, *z;
12 z = *temp;
13 if(z == NULL)
15 r =(struct Node*)malloc(sizeof(struct Node));
16 r\rightarrow coeff = x;
17 r \rightarrow pow = y;
18 *temp = r;
19 r->next = (struct Node*)malloc(sizeof(struct Node));
20 r = r \rightarrow next;
21 r->next = NULL;
22 }
24 - {
25 r\rightarrow coeff = x;
26 r \rightarrow pow = y;
27 r->next = (struct Node*)malloc(sizeof(struct Node));
28 r = r \rightarrow next;
```

```
29 r->next = NULL;
   void polyadd(struct Node *p1, struct Node *p2, struct Node *result)
33 - {
34 while(p1->next && p2->next)
35 - {
   if(p1->pow > p2->pow)
37 - {
38 result->pow = p1->pow;
    result->coeff = p1->coeff;
40 p1 = p1->next;
41
   else if(p1->pow < p2->pow)
43 - {
   result->pow = p2->pow;
44
   result->coeff = p2->coeff;
46 p2 = p2->next;
48
49 - {
   result->pow = p1->pow;
50
   result->coeff = p1->coeff+p2->coeff;
   p1 = p1->next;
53 p2 = p2->next;
54
    result->next = (struct Node *)malloc(sizeof(struct Node));
56 result = result->next;
```

```
result->next = NULL;
59 while(p1->next || p2->next)
60 - {
   if(p1->next)
62 - {
63 result->pow = p1->pow;
64 result->coeff = p1->coeff;
65 p1 = p1->next;
66 }
67 if(p2->next)
68 - {
69 result->pow = p2->pow;
70 result->coeff = p2->coeff;
71 p2 = p2->next;
72 }
73 result->next = (struct Node *)malloc(sizeof(struct Node));
    result = result->next;
75 result->next = NULL;
76 }
78 void printpoly(struct Node *node)
79 - {
80 while(node->next != NULL)
81 - {
   printf("%dx^%d", node->coeff, node->pow);
    node = node->next;
    if(node->next != NULL)
```

```
printf(" + ");
     }
 86
 88
     int main()
     struct Node *p1 = NULL, *p2 = NULL, *result = NULL;
90
     create_node(41,7,&p1);
 92 create_node(12,5,&p1);
    create_node(65,0,&p1);
94 create_node(21,5,&p2);
    create_node(15,2,&p2);
    printf("polynomial 1: ");
96
    printpoly(p1);
    printf("\npolynomial 2: ");
99
    printpoly(p2);
    result = (struct Node *)malloc(sizeof(struct Node));
100
    polyadd(p1, p2, result);
    printf("\npolynomial after adding p1 and p2 : ");
102
103
     printpoly(result);
    return 0;
```

Output:

```
polynomial 1: 41x^7 + 12x^5 + 65x^0
polynomial 2: 21x^5 + 15x^2
polynomial after adding p1 and p2 : 41x^7 + 33x^5 + 15x^2 + 65x^0
```