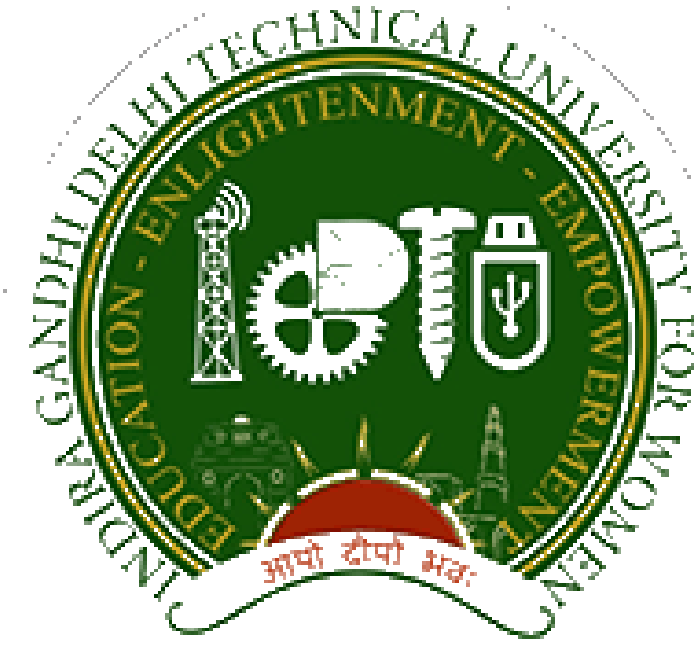
INDIRA GANDHI DELHI TECHNICAL UNIVERSITY FOR WOMEN

KASHMERE GATE,DELHI-110006



(BIT-201) DATA STRUCTURES AND ALGORITHM LAB

Submitted To :-
Dr. Jishu Rawal

Submitted By :-
Hemlata Yadav
04701182024

INDEX

SERIAL NUMBER	PRACTICAL	DATE	SIGNATURE
1.	Write a C++ program to implement Array data structure with following operations: • Traversal • Insertion • Deletion • Sorting • Searching (linear search)	**08-08-2025**	
2	Write a C++ program to perform following operations on Matrices • Addition • Subtraction • Multiplication • Transpose	**22-08-2025**	
3	Write a C++ program to perform following string operations • Concatenate two strings • Reverse a string • Find the no. of occurrences of a word in a string	**03-09-2025**	
4	Write a C++ program to implement Single Linkes List data structure with the following operations • Traversal • Insertion 1. Insertion after a particular node 2. Insertion before a particular node • Deletion • Concatenate two lists	**12-09-2025**	
5	Write a C++ program to add two polynomial equations using Linked List.	**19-09-2025**	
6	Write a C++ program to perform following operations on a Doubly Linked List • Traversal • Insertion • Deletion	**26-09-2025**	
7	Write a C++ program to perform following operations on a Circular Linked List • Traversal • Insertion • Deletion	**03-10-2025**	
8	Write a C++ program to implement Stack using Array.	**24-10-2025**	
9	Write a C++ program to implement Stack using Linked List.	**31-10-2025**	
10	Write a C++ program to implement Queue using Array	**07-11-2025**	

1.Write a C++ program to implement Array data structure with following operations:

Traversal

Insertion

Deletion

Sorting

Searching (linear search)

```
#include <iostream>
using namespace std;

class Array {
    int arr[100];
    int n;
public:
    Array() { n = 0; }

    void input() {
        cout << "Enter number of elements: ";
        cin >> n;
        cout << "Enter elements: ";
        for(int i = 0; i < n; i++) cin >> arr[i];
    }

    void traversal() {
        cout << "Array elements: ";
        for(int i = 0; i < n; i++) cout << arr[i] << " ";
        cout << endl;
    }

    void insertion(int index, int value) {
        if(index < 0 || index > n) { cout << "Invalid index\n"; return; }
        for(int i = n; i > index; i--) arr[i] = arr[i-1];
        arr[index] = value;
        n++;
    }

    void deletion(int index) {
        if(index < 0 || index >= n) { cout << "Invalid index\n"; return; }
        for(int i = index; i < n-1; i++) arr[i] = arr[i+1];
        n--;
    }


}
```

```
 void sorting() {
 for(int i = 0; i < n-1; i++)
 for(int j = i+1; j < n; j++)
 if(arr[i] > arr[j]) swap(arr[i], arr[j]);
 }
 void linearSearch(int key) {
 for(int i = 0; i < n; i++)
 if(arr[i] == key) { cout << "Element found at index " << i << endl; return; }
 cout << "Element not found\n";
 }
};
int main() {
 Array a;
 a.input();
 a.traversal();
 a.insertion(1, 99);
 a.traversal();
 a.deletion(2);
 a.traversal();
 a.sorting();
 a.traversal();
 a.linearSearch(99);
 return 0;
```

```
PS C:\Users\Gaurav Yadav> cd "c:\Users\Gaurav Yadav\Desktop\hema project\" ; if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
Enter number of elements: 6
Enter elements: 3 4 5 6 7 8
Array elements: 3 4 5 6 7 8
Array elements: 3 99 4 5 6 7 8
Array elements: 3 99 5 6 7 8
Array elements: 3 5 6 7 8 99
Element found at index 5
PS C:\Users\Gaurav Yadav\Desktop\hema project>
```

QUESTION 2
Write a C++ program to perform following operations on Matrices:
Addition
Subtraction
Multiplication
Transpose

```
#include <iostream>
using namespace std;

int main() {
   int a[10][10], b[10][10], c[10][10];
   int r, col;

   cout << "Enter rows & columns: ";
   cin >> r >> col;

   cout << "Enter matrix A:\n";
   for (int i = 0; i < r; i++) {
      for (int j = 0; j < col; j++) {
         cin >> a[i][j];
      }
   }

   cout << "Enter matrix B:\n";
   for (int i = 0; i < r; i++) {
      for (int j = 0; j < col; j++) {
         cin >> b[i][j];
      }
   }

   // Addition
   cout << "\nAddition:\n";
   for (int i = 0; i < r; i++) {
      for (int j = 0; j < col; j++) {
         c[i][j] = a[i][j] + b[i][j];
         cout << c[i][j] << " ";
      }
      cout << endl;
   }
```

```
// Subtraction
cout << "\nSubtraction:\n";
for (int i = 0; i < r; i++) {
   for (int j = 0; j < col; j++) {
      c[i][j] = a[i][j] - b[i][j];
      cout << c[i][j] << " ";
   }
   cout << endl;
}

// Multiplication
cout << "\nMultiplication:\n";
for (int i = 0; i < r; i++) {
   for (int j = 0; j < col; j++) {
      c[i][j] = 0;
      for (int k = 0; k < col; k++) {
         c[i][j] += a[i][k] * b[k][j];
      }
      cout << c[i][j] << " ";
   }
   cout << endl;
}

// Transpose of A
cout << "\nTranspose of A:\n";
for (int i = 0; i < col; i++) {
   for (int j = 0; j < r; j++) {
      cout << a[j][i] << " ";
   }
   cout << endl;
}

return 0;
```

```
PS C:\Users\Gaurav Yadav\Desktop\hema project> cd "c:\Users\Gaurav Yadav\Desktop\hema project\" ; if ($?) { g++ main.cpp -o main } ; if (
Enter rows & columns: 2
3
Enter matrix A:
2*2
Enter matrix B:

Addition:
10 216 1273571541
2008831899 1999028612 -412633555

Subtraction:
-6 -216 -1812459957
-2000433627 -1986184596 -126397701

Multiplication:
-1810863600 -552892944 -1521469526
1414542248 1128166000 -1100018624

Transpose of A:
2 4199136
0 6422008
1878039440 1877968020
PS C:\Users\Gaurav Yadav\Desktop\hema project>
```

QUESTION 3
Write a C++ program to perform following string operations:
Concatenate two strings
Reverse a string
Find the number of occurrences of a word in a string

```
#include <iostream>
#include <string>
#include <algorithm>   // Required for reverse()
using namespace std;

int main() {
   string s1, s2, word;

   cout << "Enter first string: ";
   getline(cin, s1);

   cout << "Enter second string: ";
   getline(cin, s2);

   // Concatenation
   cout << "\nConcatenated String: " << s1 + s2 << endl;

   // Reverse
   string rev = s1;
   reverse(rev.begin(), rev.end());
   cout << "Reversed String: " << rev << endl;

   // Occurrence of a word
   cout << "\nEnter word to count occurrences: ";
   cin >> word;

   int count = 0;
   size_t pos = 0;
```

```
    while ((pos = s1.find(word, pos)) != string::npos) {
        count++;
        pos += word.length();
    }

    cout << "Occurrences of '" << word << "': " << count << endl;

    return 0;
}
```

```
PS C:\Users\Gaurav Yadav> cd "c:\Users\Gaurav Yadav\Desktop\hema project\" ; i
Enter first string: hemlata
Enter second string: yadav

Concatenated String: hemlatayadav
Reversed String: atalmeh

Enter word to count occurrences: a
Occurrences of 'a': 2
PS C:\Users\Gaurav Yadav\Desktop\hema project>
```

QUESTION 4

Write a C++ program to implement Single Linked List with:

Traversal

Insertion

Insertion after a particular node

Insertion before a particular node

Deletion

Concatenate two lists

```
#include <iostream>
using namespace std;

struct Node {
    int data;
    Node* next;
};

// Function to traverse list
void traverse(Node* head) {
    Node* temp = head;
    while (temp != NULL) {
        cout << temp->data << " ";
        temp = temp->next;
    }
    cout << endl;
}

// Function to insert at end
void insertEnd(Node*& head, int val) {
    Node* newNode = new Node{val, NULL};

    if (head == NULL) {
        head = newNode;
        return;
    }

    Node* temp = head;
    while (temp->next != NULL)
        temp = temp->next;

    temp->next = newNode;
}
```

```
// Insert after a given value
void insertAfter(Node* head, int key, int val) {
   Node* temp = head;

   while (temp != NULL && temp->data != key)
      temp = temp->next;

   if (temp == NULL) {
      cout << "Node with value " << key << " not found.\n";
      return;
   }

   Node* newNode = new Node{val, temp->next};
   temp->next = newNode;
}

// Insert before a given value
void insertBefore(Node*& head, int key, int val) {
   if (head == NULL) return;

   // If inserting before the head
   if (head->data == key) {
      Node* newNode = new Node{val, head};
      head = newNode;
      return;
   }

   Node* temp = head;

   while (temp->next != NULL && temp->next->data != key)
      temp = temp->next;

   if (temp->next == NULL) {
      cout << "Node with value " << key << " not found.\n";
      return;
   }

   Node* newNode = new Node{val, temp->next};
   temp->next = newNode;
}
```

```
// Delete a node
void deleteNode(Node*& head, int key) {
    if (head == NULL) return;

    // If head is to be deleted
    if (head->data == key) {
        Node* del = head;
        head = head->next;
        delete del;
        return;
    }

    Node* temp = head;

    while (temp->next != NULL && temp->next->data != key)
        temp = temp->next;

    if (temp->next == NULL) {
        cout << "Node not found.\n";
        return;
    }

    Node* del = temp->next;
    temp->next = del->next;
    delete del;
}

// Concatenate two linked lists
Node* concatenate(Node* head1, Node* head2) {
    if (head1 == NULL) return head2;
    if (head2 == NULL) return head1;

    Node* temp = head1;
    while (temp->next != NULL)
        temp = temp->next;

    temp->next = head2;
    return head1;
}
```

```
int main() {
    Node *head = NULL, *head2 = NULL;
    int n, val, key;

    cout << "Enter number of nodes for first list: ";
    if (!(cin >> n)) return 0;

    cout << "Enter elements:\n";
    for (int i = 0; i < n; i++) {
        cin >> val;
        insertEnd(head, val);
    }

    cout << "\nTraversal: ";
    traverse(head);

    // Insert after
    cout << "\nEnter value after which to insert and new value: ";
    cin >> key >> val;
    insertAfter(head, key, val);
    cout << "After insertion after: ";
    traverse(head);

    // Insert before
    cout << "\nEnter value before which to insert and new value: ";
    cin >> key >> val;
    insertBefore(head, key, val);
    cout << "After insertion before: ";
    traverse(head);

    // Delete
    cout << "\nEnter value to delete: ";
    cin >> key;
    deleteNode(head, key);
    cout << "After deletion: ";
    traverse(head);

    // Second list
    cout << "\nEnter number of nodes for second list: ";
    cin >> n;
```

```
    cout << "Enter elements of second list:\n";
    for (int i = 0; i < n; i++) {
        cin >> val;
        insertEnd(head2, val);
    }

    cout << "Second list: ";
    traverse(head2);

    // Concatenate — ensure closing parenthesis and semicolon
    head = concatenate(head, head2);

    cout << "\nConcatenated list: ";
    traverse(head);

    return 0;
}
```

```
Enter number of nodes for first list: 5
Enter elements:
23456
23
234
56
78

Traversal: 23456 23 234 56 78

Enter value after which to insert and new value: 234
890
After insertion after: 23456 23 234 890 56 78

Enter value before which to insert and new value: 890
678
After insertion before: 23456 23 234 678 890 56 78

Enter value to delete: 23456
After deletion: 23 234 678 890 56 78

Enter number of nodes for second list: 3
Enter elements of second list:
2
3
4
Second list: 2 3 4

Concatenated list: 23 234 678 890 56 78 2 3 4
PS C:\Users\Gaurav Yadav\Desktop\hema project>
```

QUESTION 5

Write a C++ program to add two polynomial equations using Linked List.

```
#include <iostream>
using namespace std;

struct Node {
   int coeff;
   int pow;
   Node* next;
};

// Insert term at end
void insert(Node*& head, int c, int p) {
   Node* newNode = new Node{c, p, NULL};

   if (head == NULL) {
      head = newNode;
      return;
   }

   Node* temp = head;
   while (temp->next != NULL)
      temp = temp->next;

   temp->next = newNode;
}

// Display polynomial
void display(Node* head) {
   if (head == NULL) {
      cout << "0\n";
      return;
   }

   Node* temp = head;
   while (temp != NULL) {
      cout << temp->coeff << "x^" << temp->pow;
      if (temp->next != NULL)
         cout << " + ";
      temp = temp->next;
   }
   cout << endl;
}
```

```
// Add two polynomials
Node* addPoly(Node* poly1, Node* poly2) {
   Node* result = NULL;

   while (poly1 != NULL && poly2 != NULL) {
      if (poly1->pow == poly2->pow) {
         insert(result, poly1->coeff + poly2->coeff, poly1->pow);
         poly1 = poly1->next;
         poly2 = poly2->next;
      }
      else if (poly1->pow > poly2->pow) {
         insert(result, poly1->coeff, poly1->pow);
         poly1 = poly1->next;
      }
      else {
         insert(result, poly2->coeff, poly2->pow);
         poly2 = poly2->next;
      }
   }

   // Add remaining terms
   while (poly1 != NULL) {
      insert(result, poly1->coeff, poly1->pow);
      poly1 = poly1->next;
   }

   while (poly2 != NULL) {
      insert(result, poly2->coeff, poly2->pow);
      poly2 = poly2->next;
   }

   return result;
}

int main() {
   Node *poly1 = NULL, *poly2 = NULL, *result = NULL;
   int n, coeff, power;

   cout << "Enter number of terms for first polynomial: ";
   if (!(cin >> n)) return 0;
```

```
    cout << "Enter terms as: coefficient power (one per line)\n";
    for (int i = 0; i < n; ++i) {
        cin >> coeff >> power;
        insert(poly1, coeff, power);
    }

    cout << "\nFirst Polynomial: ";
    display(poly1);

    cout << "\nEnter number of terms for second polynomial: ";
    cin >> n;
    cout << "Enter terms as: coefficient power (one per line)\n";
    for (int i = 0; i < n; ++i) {
        cin >> coeff >> power;
        insert(poly2, coeff, power);
    }

    cout << "\nSecond Polynomial: ";
    display(poly2);

    result = addPoly(poly1, poly2);

    cout << "\nResultant Polynomial: ";
    display(result);

    return 0;
}
```

```
PS C:\Users\Gaurav Yadav> cd "c:\Users\Gaurav Yadav\Desktop\hema project\"
Enter number of terms for first polynomial: 3
Enter terms as: coefficient power (one per line)
3
2
1

4
5
6

First Polynomial: 3x^2 + 1x^4 + 5x^6

Enter number of terms for second polynomial: 3
Enter terms as: coefficient power (one per line)
2
3
4
5
6
7

Second Polynomial: 2x^3 + 4x^5 + 6x^7

Resultant Polynomial: 2x^3 + 4x^5 + 6x^7 + 3x^2 + 1x^4 + 5x^6
PS C:\Users\Gaurav Yadav\Desktop\hema project>
```