DSA Lab File 2022-23

Submitted by:

BUSHRA SHAHZAD

21BCS046

3rd Semester

CEN-391

INDEX

Serial No.	Name of the program	Date	Remarks
	LAB WORK QUESTIONS		
1	WAP to print 1-d & 2-d array using pointer.	18-07-22	
2	WAP to insert & delete the element at any index from array.	25-07-22	
3	WAP to implement structure functionality.	01-08-22	
4	WAP to implement stack ADT.	22-08-22	
5	WAP using recursive function: factorial, power, sum of array, Fibonacci term, Fibonacci sum.	29-08-22	
6	WAP to implement singly linked list.	12-09-22	
7	WAP to implement circular queue using array.	26-09-22	
8	WAP to implement CBT using array.	17-10-22	
9	WAP to implement CBT using linked list.	14-11-22	
	ASSINMENT QUESTIONS		
10	Conversion of number system to all.	25-11-22	
11	WAP to implement doubly linked list.	25-11-22	
12	WAP to implement XOR linked list.	25-11-22	

Question 1: WAP to implement 1-d and 2-d array using pointers.

Code:

```
// to print 1d & 2d array using pointers
#include <iostream>
using namespace std;

int main()
{
    int n1, *array, m, n, **array2;
    cout << "Enter the size of  1-D array : ";
    cin >> n1;
    array = new int[n1];
    cout << "Enter the elements : \n";
    for (int i = 0; i < n1; i++)
    {
        cin >> *(array + i);
    }
    cout << "Printing 1-d array:\n";
    cout<<"\nBushra Shahzad : 21BCS046\n";
    for (int i = 0; i < n1; i++)
    {
        cout << *(array + i) << " ";
    }
    cout << "\nEnter the rows and columns of  2-D array respectuvely : \n";
    cin >> m >> n;
    array2 = new int *[m];

    for (int i = 0; i < m; i++)
    {
        array2[i] = new int[n];
    }
    cout << "Enter the elements : \n";
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            cin >> *(*(array2 + i) + j);
        }
    }
    cout << "Printing 2-d array:\n";
    cout<<"\nBushra Shahzad : 21BCS046\n";
    for (int i = 0; i < m; i++)
    {
```

```
        for (int j = 0; j < n; j++)
        {
            cout << *(*(array2 + i) + j) << "\t";
        }
        cout << "\n";
    }
    delete array;
    delete array2;

    return 0;
}
```

Output 1:

```
Enter the size of  1-D array : 5
Enter the elements :
11
12
13
14
15
Printing 1-d array:

Bushra Shahzad : 21BCS046
11 12 13 14 15
Enter the rows and columns of  2-D array respectuvely :
2
3
Enter the elements :
1
2
3
4
5
6
Printing 2-d array:

Bushra Shahzad : 21BCS046
1       2       3
4       5       6
PS D:\Whioo\Sem III\DS Lab\Final Programs>
```

Question 2: WAP to insert & delete the element at any index from array.

Code:

```
// to insert and delete from particluar position
#include <iostream>
using namespace std;
void display(int *arr, int n)
{
    cout << "Displaying Array:\n";
    for (int i = 0; i < n; i++)
    {
        cout << arr[i] << " ";
    }
    cout << "\n";
}
void insertAtIndex(int *arr, int &n, int index, int value)
{
    if (index >= 0 && index <= n)
    {
        int i;
        for (i = n; i >= index; i--)
        {
            arr[i + 1] = arr[i];
        }
        n++;
        arr[index] = value;
    }
    else
    {
        cout << "Index exceeds the no of elements.\n";
    }
}
void deleteElement(int *arr, int &n, int value)
{
    int flag = 0, newIndex;
    for (int i = 0; i < n; i++)
    {
        if (arr[i] == value)
        {
            newIndex = i;
        }
    }

    for (int i = newIndex; i < n; i++)
    {
        arr[i] = arr[i + 1];
    }
    n--;
```

```
}
int main()
{
    int n, choice, index, value;
    cout << "Enter the size of array : ";
    cin >> n;
    int array[n];
    for (int i = 0; i < n; i++)
    {
        cin >> array[i];
    }
    display(array, n);

    while (1)
    {
        cout << "\nEnter '1' to insert element at at any index.";
        cout << "\nEnter '2' to delete particular element :";
         cout << "\nEnter '3' to exit! :";
        cin >> choice;
        switch (choice)
        {
        case 1:
            cout << "Enter the index where you want to insert in an array : ";
            cin >> index;
            cout << "Enter the value : ";
            cin >> value;
            insertAtIndex(array, n, index, value);
            display(array, n);
            break;
        case 2:
            cout << "Enter the element to be deleted : ";
            cin >> value;
            deleteElement(array, n, value);
            display(array, n);
            break;
        case 3:
            exit(1);
            break;

        default: cout<<"Invalid input!\n";
            break;
        }
    }

    return 0;
}
```

Output 2:

```
PS D:\Whioo\Sem III\DS Lab\Final Programs> cd "d:\Whioo\Sem III\DS Lab\Final P
Enter the size of array : 4
11
13
15
17
Displaying Array:
11 13 15 17

Bushra Shahzad : 21BCS046

Enter '1' to insert element at at any index.
Enter '2' to delete particular element :
Enter '3' to exit! :1
Enter the index where you want to insert in an array : 2
Enter the value : -30
Displaying Array:
11 13 -30 15 17

Bushra Shahzad : 21BCS046

Enter '1' to insert element at at any index.
Enter '2' to delete particular element :
Enter '3' to exit! :2
Enter the element to be deleted : -30
Displaying Array:
11 13 15 17

Bushra Shahzad : 21BCS046

Enter '1' to insert element at at any index.
Enter '2' to delete particular element :
Enter '3' to exit! :
```

Question 3: WAP to implement structure functionality.

Code:

```
#include <iostream>
#include <bits/stdc++.h>
using namespace std;
struct student
{
    char name[30];
    int roll_no;
    int sem;
    int marks;
};
student s[30];
void enterData(int n)
{
    int i = 0;
    while (i < n)

    {
        cout << "Enter data of student " << i + 1 << "\n";
        cout << "Name  : ";
        getchar();
        gets(s[i].name);

        cout << "Roll No : ";
        cin >> s[i].roll_no;
        cout << "Semester : ";
        cin >> s[i].sem;
        cout << "Marks : ";
        cin >> s[i].marks;
        i++;
    }
}
void display(int n)
{
    cout << "Roll No\t\tName\t\t\tSemester\t\tMarks\n";
    int i = 0;
    while (i < n)
    {
        cout << s[i].roll_no << "\t\t" << s[i].name << "\t\t\t" << s[i].sem <<
"\t\t\t" << s[i].marks << "\n";
        i++;
    }
}
void search_name(int n)
{
    int flag = 0;
```

```
    char searchName[50];
    cout << "Enter the name you want to search : ";
    cin >> searchName;
    for (int i = 0; i < n; i++)
    {
        if (strcmp(s[i].name, searchName) == 0)
        {
            flag = 1;
            cout << "Following are the records of " << searchName << " : " <<
endl;
            cout << "Roll No\t\tName\t\t\tSemester\t\tMarks\n";
            cout << s[i].roll_no << "\t\t" << s[i].name << "\t\t\t" <<
s[i].sem << "\t\t\t" << s[i].marks << "\n";
            break;
        }
    }
    if (flag == 0)
        cout << "\nNo student by such name exists!\n";
}
void search_roll(int n)
{
    int flag = 0;
    int x;
    cout << "Enter the roll no. you want to search : ";
    cin >> x;
    for (int i = 0; i < n; i++)
    {
        if (s[i].roll_no == x)
        {
            flag = 1;
            cout << "Following are the records of roll no." << x << " : " <<
endl;
            cout << "Roll No\t\tName\t\t\tSemester\t\tMarks\n";
            cout << s[i].roll_no << "\t\t" << s[i].name << "\t\t\t" <<
s[i].sem << "\t\t\t" << s[i].marks << "\n";
            break;
        }
    }
    if (flag == 0)
        cout << "\nNo student of such roll no exists!\n";
}
void compare(int n)
{
    int max = 0, x;
    for (int i = 0; i < n; i++)
    {
        if (s[i].marks > max)
            max = s[i].marks;
```

```
        x = i;
    }
    cout << "Following are the records of topper"
         << " : " << endl;
    cout << "Roll No\t\tName\t\t\tSemester\t\tMarks\n";
    cout << s[x].roll_no << "\t\t" << s[x].name << "\t\t\t" << s[x].sem <<
"\t\t\t" << s[x].marks << "\n";
}
int main()
{
    int choice, n, flag;
    char searchName[50];
    while (1)
    {
        cout << "\nBushra Shahzad : 21BCS046\n";
        cout << "\nPress '1' to enter student data.\n";
        cout << "Press '2' to display student records.\n";
        cout << "Press '3' to search student by Roll No.\n";
        cout << "Press '4' to search student by name.\n";
        cout << "Press '5' to display topper student's details.\n";
        cout << "Press '6' to Exit!\n";
        cin >> choice;
        switch (choice)
        {
        case 1:

            cout << "Enter no of students : ";
            cin >> n;

            enterData(n);

            break;
        case 2:
            display(n);
            break;
        case 3:
            search_roll(n);
            break;
        case 4:
            search_name(n);
            break;
        case 5:
            compare(n);

            break;
        case 6:
            exit(1);
            break;
```

```
        default:
            cout << "Invalid input!\n";
            break;
        }
    }

    return 0;
}
```

Output 3:

```
Bushra Shahzad : 21BCS046

Press '1' to enter student data.
Press '2' to display student records.
Press '3' to search student by Roll No.
Press '4' to search student by name.
Press '5' to display topper student's details.
Press '6' to Exit!
1
Enter no of students : 3
Enter data of student 1
Name  : bushra
Roll No : 1
Semester : 1
Marks : 90
Enter data of student 2
Name  : shahzad
Roll No : 2
Semester : 1
Marks : 89
Enter data of student 3
Name  : alex
Roll No : 3
Semester : 1
Marks : 100

Bushra Shahzad : 21BCS046
```

```
2
Roll No         Name                    Semester                Marks
1               bushra                  1                       90
2               shahzad                 1                       89
3               alex                    1                       100

Bushra Shahzad : 21BCS046
```

```
3
Enter the roll no. you want to search : 3
Following are the records of roll no.3 :
Roll No         Name                    Semester                Marks
3               alex                    1                       100

Bushra Shahzad : 21BCS046
```

```
4
Enter the name you want to search : bush

No student by such name exists!

Bushra Shahzad : 21BCS046
```

```
Press '0' to Exit!
5
Following are the records of topper :
Roll No         Name                    Semester                Marks
3               alex                    1                       100

Bushra Shahzad : 21BCS046
```

Question 4: WAP to implement stack ADT.

Code:

```
#include <iostream>
using namespace std;
struct stack
{
    int top;
    int size;
    int *s;
};
stack st;
void display()
{
    int i;
    cout << "Displaying Stack:\n";
    for (int i = st.top; i >= 0; i--)
    {
        cout << st.s[i] << " ";
    }
}
void push(int x)
{
    if (st.top == st.size - 1)
    {
        cout << "Stack is full!\n";
    }
    else
    {
        st.top++;
        st.s[st.top] = x;
    }
}
void pop()
{
    if (st.top == -1)
    {
        cout << "Satck is empty.\n";
    }
    else
    {
        st.s[st.top] = NULL;
    }
    st.top--;
}
int isEmpty()
{
    if (st.top == -1)
```

```
        return 1;
    return 0;
}
int isFull()
{
    if (st.top == st.size - 1)
        return 1;
    return 0;
}
int top()
{
    if (st.top == -1)
        return -1;
    return st.s[st.top];
}

int main()
{
    cout << "Enter the size of a stack" << endl;
    cin >> st.size;
    st.s = new int[st.size];
    st.top = -1;
    int n, key;
    while (1)
    {
        cout << "\nBushra Shahzad : 21BCS046\n";
        cout << "\nPress '1' to push an element.\n";
        cout << "Press '2' to pop an element.\n";
        cout << "Press '3' for checking if stack is empty.\n";
        cout << "Press '4' for checking if stack is full.\n";
        cout << "Press '5' for displaying the top element.\n";
        cout << "Press '6' to Exit!.\n";
        cin >> key;
        switch (key)
        {
        case 1:
            cout << "Enter the element to be pushed in stack" << endl;
            cin >> n;
            push(n);
            display();
            break;
        case 2:

            pop();
            display();
            break;
        case 3:
            if (isEmpty())
```

```
                cout << "yes, stack is empty" << endl;
            else
                cout << "noi, stack is not empty!" << endl;
            break;
        case 4:
            if (isFull())
                cout << "yes, stack is full" << endl;
            else
                cout << "noi, stack is not full!" << endl;

            break;
        case 5:
            cout << "The top element of stack is " << top() << endl;
            break;
        case 6:
            exit(1);
            break;
        default:
            cout << "Invalid input!" << endl;
            break;
        }
    }
    return 0;
}
```

Output 4:

```
Press '1' to push an element.
Press '2' to pop an element.
Press '3' for checking if stack is empty.
Press '4' for checking if stack is full.
Press '5' for displaying the top element.
Press '6' to Exit!.
1
Enter the element to be pushed in stack
25
Displaying Stack:
25 24 23 22 21
Bushra Shahzad : 21BCS046

Press '1' to push an element.
Press '2' to pop an element.
Press '3' for checking if stack is empty.
Press '4' for checking if stack is full.
Press '5' for displaying the top element.
Press '6' to Exit!.
1
Enter the element to be pushed in stack
0
Stack is full!
Displaying Stack:
25 24 23 22 21
Bushra Shahzad : 21BCS046
```

```
Displaying Stack:
25 24 23 22 21
Bushra Shahzad : 21BCS046

Press '1' to push an element.
Press '2' to pop an element.
Press '3' for checking if stack is empty.
Press '4' for checking if stack is full.
Press '5' for displaying the top element.
Press '6' to Exit!.
2
Displaying Stack:
24 23 22 21
Bushra Shahzad : 21BCS046
```

```
3
noi, stack is not empty!

Bushra Shahzad : 21BCS046

Press '1' to push an element.
Press '2' to pop an element.
Press '3' for checking if stack is empty.
Press '4' for checking if stack is full.
Press '5' for displaying the top element.
Press '6' to Exit!.
4
yes, stack is full

Bushra Shahzad : 21BCS046
```

```
5
The top element of stack is 25

Bushra Shahzad : 21BCS046
```

Question 5: WAP using recursive function: factorial, power, sum of array, Fibonacci term, Fibonacci sum.

Code:

```
// recursive functions
#include <iostream>
using namespace std;
int factorial(int n)
{
    if (n == 0 || n == 1)
        return 1;
    return n * factorial(n - 1);
}
int power(int x, int y)
{

    if (y == 1)
        return x;

    return x * power(x, y - 1);
}
int sumArray(int *arr, int n)
{
    if (n == 0)
        return 0;
    return arr[n - 1] + sumArray(arr, n - 1);
}
int fibonacciTerm(int n)
{
    if (n == 0 || n == 1)
        return n;
    return fibonacciTerm(n - 1) + fibonacciTerm(n - 2);
}
int fibonacciSum(int n)
{
    if (n == 0 || n == 1)
        return n;
    return fibonacciTerm(n) + fibonacciSum(n - 1);
}
int main()

{
    int a, b, n;
    int choice;
    while (1)
    {
        cout << "\n--\nEnter '1' to find factorial of number.\n";
        cout << "Enter '2' to find power a to b.\n";
```

```
        cout << "Enter '3' to find sum of array. \n";
        cout << "Enter '4'to print fibonacci series till n.\n";
        cout << "Enter '5' to find sum of n terms of fibonacci.\n";
        cout << "Enter '6' to Exit!\n";
        cin >> choice;
        switch (choice)
        {
        case 1:
            cout << "Enter the number:\n";
            cin >> n;
            cout << "Factorial of "<<n<<" is " << factorial(n)<<".";
            break;

            break;
        case 2:
            cout << "Enter a and b respectively:\n";
            cin >> a;
            cin >> b;
            cout <<a<< "^" <<b<<" = " << power(a, b) << endl;
            break;
        case 3:
            cout << "Enter the number of elements:\n";
            cin >> n;
            int array[100];
            cout << "Enter the elements:\n";
            for (int i = 0; i < n; i++)
            {
                cin >> array[i];
            }
            cout << "Sum of array ele is " << sumArray(array, n);

            break;
        case 4:

            cout << "Enter the number of elements:\n";
            cin >> n;
            cout << "Printing fib:\n";
            for (int i = 0; i <= n; i++)
            {
                cout << fibonacciTerm(i) << " ";
            }

            break;
        case 5:
            cout << "Enter the number of elements:\n";
            cin >> n;
            cout << "Sum of "<<n<<" fibonacci terms is " << fibonacciSum(n);
            break;
```

```
        case 6:
            exit(1);
            break;

        default:
            cout << "Invalid input!\n";
            break;
        }
    }

    return 0;
}
```

Output 5:

```
Bushra Shahzad : 21BCS046

Enter '1' to find factorial of number.
Enter '2' to find power a to b.
Enter '3' to find sum of array.
Enter '4'to print fibonacci series till n.
Enter '5' to find sum of n terms of fibonacci.
Enter '6' to Exit!
1
Enter the number:
5
Factorial of 5 is 120.
Bushra Shahzad : 21BCS046

Enter '1' to find factorial of number.
Enter '2' to find power a to b.
Enter '3' to find sum of array.
Enter '4'to print fibonacci series till n.
Enter '5' to find sum of n terms of fibonacci.
Enter '6' to Exit!
2
Enter a and b respectively:
2
4
2^4 = 16

Bushra Shahzad : 21BCS046
```

```
3
Enter the number of elements:
5
Enter the elements:
1
2
3
4
5
Sum of array ele is 15
Bushra Shahzad : 21BCS046

Enter '1' to find factorial of number.
Enter '2' to find power a to b.
Enter '3' to find sum of array.
Enter '4'to print fibonacci series till n.
Enter '5' to find sum of n terms of fibonacci.
Enter '6' to Exit!
4
Enter the number of elements:
5
Printing fib:
0 1 1 2 3 5
Bushra Shahzad : 21BCS046

Enter '1' to find factorial of number.
Enter '2' to find power a to b.
Enter '3' to find sum of array.
Enter '4'to print fibonacci series till n.
Enter '5' to find sum of n terms of fibonacci.
Enter '6' to Exit!
5
Enter the number of elements:
4
Sum of 4 fibonacci terms is 7
Bushra Shahzad : 21BCS046
```

Question 6: WAP to implement singly linked list.

Code:

```
// linked list
#include <iostream>
using namespace std;
struct node
{
    int data;
    node *next;
} *head = NULL;
// display functon
void display()
{
    if (head == NULL)
    {
        cout << "List is empty!\n";
        return;
    }
    struct node *temp;
    temp = head;
    cout << "The linked list is : \n";
    while (temp != NULL)
    {
        cout << temp->data << " ";
        if (temp->next != NULL)
        {
            cout << " -> ";
        }
        temp = temp->next;
    }
    cout << endl;
}
// insert at beginning
void insertHead()
{

    int value;
    cout << "Enter the value to be inserted.\n";
    cin >> value;
    struct node *temp;
    temp = new node;
    temp->data = value;

    if (head == NULL)
    {
        temp->next = NULL;
        head = temp;
```

```
    }
    else
    {
        temp->next = head;
        head = temp;
    }
}
// insert at end
void insertEnd()
{
    int value;
    cout << "Enter the value to be inserted.\n";
    cin >> value;
    struct node *temp, *traversal;
    temp = new node;
    traversal = head;
    temp->data = value;
    while (traversal->next != NULL)
    {
        traversal = traversal->next;
    }
    temp->next = NULL;
    traversal->next = temp;
}
// insert at any position
void insertAtPos()
{
    int value, pos, counter = 0;
    cout << "Enter the value to be inserted: ";
    cin >> value;
    struct node *temp, *traversal, *ptr;
    temp = new node;
    cout << "Enter the postion at which node to be inserted: ";
    cin >> pos;
    int i;
    traversal = head;
    temp->data = value;
    while (traversal != NULL)
    {
        traversal = traversal->next;
        counter++;
    }
    if (pos == 1)
    {
        if (head == NULL)
        {
            head = temp;
            head->next = NULL;
```

```
        }
        else
        {
            ptr = head;
            head = temp;
            head->next = ptr;
        }
    }
    else if (pos > 1 && pos <= counter)
    {
        traversal = head;
        for (i = 1; i < pos; i++)
        {
            ptr = traversal;
            traversal = traversal->next;
        }
        ptr->next = temp;
        temp->next = traversal;
    }
    else
    {
        cout << "Positon out of range" << endl;
    }
}
// delete at beginning
void deleteHead()
{
    struct node *temp;
    temp = head;
    head = head->next;
    temp->next = NULL;
    delete (temp);
}
// delete at end
void deleteEnd()
{
    struct node *cur, *prev;
    cur = head;
    prev = NULL;
    while (cur->next != NULL)
    {
        prev = cur;
        cur = cur->next;
    }
    prev->next = NULL;
    delete cur;
}
// delete at position
```

```
void deletePosition()
{
    int pos, counter = 0, i;
    if (head == NULL)
    {
        cout << "List is empty" << endl;
        return;
    }
    cout << "Enter the position whose next node you want to delete : ";
    cin >> pos;

    if (pos == 1)
    {
        deleteHead();
    }
    else if (pos > 1)
    {
        struct node *cur, *prev;
        cur = head;
        prev = NULL;
        for (int i = 2; i <= pos; i++)
        {
            prev = cur;
            cur = cur->next;
        }
        if (cur->next == NULL)
        {
            cout << "No node exist after " << pos << "\n";
        }
        else
        {
            prev->next = cur->next;
            cur->next = NULL;
            delete cur;
        }
    }
}

int main()
{

    int choice;

    while (1)
    {
        cout << "\nPress '1' to insert node at beginning.\n";
        cout << "Press '2' to insert node at end.\n";
        cout << "Press '3' to insert node at given position.\n";
```

```
        cout << "Press '4' to delete node from beginning.\n";
        cout << "Press '5' to delete node from end.\n";
        cout << "Press '6' to delete node at given position.\n";
        cout << "Press '7' to exit.\n";
        cin >> choice;
        switch (choice)
        {
        case 1:
            insertHead();
            display();

            break;
        case 2:
            insertEnd();
            display();
            break;
        case 3:
            insertAtPos();
            display();
            break;
        case 4:
            deleteHead();
            display();
            break;
        case 5:
            deleteEnd();
            display();
            break;
        case 6:
            deletePosition();
            display();
            break;
        case 7:
            exit(1);
            break;

        default:
            cout << "Invalid Input!\n";
            break;
        }
    }

    return 0;
}
```

Output 6:

```
Bushra Shahzad : 21BCS046

Press '1' to insert node at beginning.
Press '2' to insert node at end.
Press '3' to insert node at given position.
Press '4' to delete node from beginning.
Press '5' to delete node from end.
Press '6' to delete node at given position.
Press '7' to exit.
1
Enter the value to be inserted.
15
The linked list is :
15  -> 13  -> 12

Bushra Shahzad : 21BCS046

Press '1' to insert node at beginning.
Press '2' to insert node at end.
Press '3' to insert node at given position.
Press '4' to delete node from beginning.
Press '5' to delete node from end.
Press '6' to delete node at given position.
Press '7' to exit.
2
Enter the value to be inserted.
10
The linked list is :
15  -> 13  -> 12  -> 10

Bushra Shahzad : 21BCS046
```

```
The linked list is :
15  -> 13  -> 12  -> 10

Bushra Shahzad : 21BCS046

Press '1' to insert node at beginning.
Press '2' to insert node at end.
Press '3' to insert node at given position.
Press '4' to delete node from beginning.
Press '5' to delete node from end.
Press '6' to delete node at given position.
Press '7' to exit.
3
Enter the value to be inserted: 300
Enter the postion at which node to be inserted: 2
The linked list is :
15  -> 300  -> 13  -> 12  -> 10

Bushra Shahzad : 21BCS046
```

```
15  -> 300  -> 13  -> 12  -> 10

Bushra Shahzad : 21BCS046

Press '1' to insert node at beginning.
Press '2' to insert node at end.
Press '3' to insert node at given position.
Press '4' to delete node from beginning.
Press '5' to delete node from end.
Press '6' to delete node at given position.
Press '7' to exit.
4
The linked list is :
300  -> 13  -> 12  -> 10

Bushra Shahzad : 21BCS046

Press '1' to insert node at beginning.
Press '2' to insert node at end.
Press '3' to insert node at given position.
Press '4' to delete node from beginning.
Press '5' to delete node from end.
Press '6' to delete node at given position.
Press '7' to exit.
5
The linked list is :
300  -> 13  -> 12

Bushra Shahzad : 21BCS046
```

```
The linked list is :
300  -> 13  -> 12

Bushra Shahzad : 21BCS046

Press '1' to insert node at beginning.
Press '2' to insert node at end.
Press '3' to insert node at given position.
Press '4' to delete node from beginning.
Press '5' to delete node from end.
Press '6' to delete node at given position.
Press '7' to exit.
6
Enter the position whose next node you want to delete : 2
The linked list is :
300  -> 12

Bushra Shahzad : 21BCS046
```

Question 7: WAP to implement circular queue using array.

Code:

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

struct Queue
{
    int rear, front;
    int size;
    int *arrayQueue;
};
Queue que;
void enqueue(int value)
{

    if ((que.front == 0 && que.rear == que.size - 1) ||
        (que.rear == (que.front - 1)))
    {
        cout << "\nQueue is full";
        return;
    }
    else if (que.front == -1)
    {
        que.front = que.rear = 0;
        que.arrayQueue[que.rear] = value;
    }
    else if (que.rear == que.size - 1 && que.front != 0)
    {
        que.rear = 0;
        que.arrayQueue[que.rear] = value;
    }
    else
    {
        que.rear++;
        que.arrayQueue[que.rear] = value;
    }
}

void dequeue()
{
    if (que.front == -1)
    {
        cout << "\nQueue is empty";
    }

    que.arrayQueue[que.front] = -1;
    if (que.front == que.rear)
```

```
    {
        que.front = -1;
        que.rear = -1;
    }
    else if (que.front == que.size - 1)
        que.front = 0;
    else
        que.front++;
}

void display()
{
    if (que.front == -1)
    {
        cout << "\nQueue is empty";
        return;
    }
    cout << "\nCircular queue : ";
    if (que.rear >= que.front)
    {
        for (int i = que.front; i <= que.rear; i++)
            cout << " " << que.arrayQueue[i];
    }
    else
    {
        for (int i = que.front; i < que.size; i++)
            cout << " " << que.arrayQueue[i];
        for (int i = 0; i <= que.rear; i++)
            cout << " " << que.arrayQueue[i];
    }
}
void frontRear()
{
    cout << "Front = " << que.arrayQueue[que.front] << endl;
    cout << "Rear = " << que.arrayQueue[que.rear] << endl;
}
int isEmpty()
{
    if (que.front == -1)
        return 1;
    return 0;
}
int isFull()
{
    if ((que.front == 0 && que.rear == que.size - 1) ||
        (que.rear == (que.front - 1)))
        return 1;
    return 0;
```

```
}

int main()
{
    int choice, value;
    cout << "Enter the size of queue: ";
    cin >> que.size;
    que.arrayQueue = new int[que.size];
    que.front = que.rear = -1;

    while (1)
    {
        cout << "\nBushra Shahzad : 21BCS046\n";
        cout << "\nPress '1' to enqueue an element.";
        cout << "\nPress '2' to dequeue an element.";
        cout << "\nPress '3' to check if queue is full.";
        cout << "\nPress '4' to check if queue is empty.";
        cout << "\nPress '5' to display front and rear. ";
        cout << "\nPress '6' to display circular queue!.";
        cout << "\nPress '7' to exit!.\n";
        cin >> choice;
        switch (choice)
        {
        case 1:
            cout << "Enter the element to be inserted : ";
            cin >> value;
            enqueue(value);
            break;
        case 2:
            dequeue();
            break;
        case 3:
            if (isFull())
                cout << "yes, queue is full" << endl;
            else
                cout << "noi, queue is not full!" << endl;

            break;
        case 4:
            if (isEmpty())
                cout << "yes, queue is empty" << endl;
            else
                cout << "noi, queue is not empty!" << endl;
            break;
        case 5:
            frontRear();
            break;
        case 6:
```

```
            display();
            break;
        case 7:
            exit(1);
            break;

        default:
            cout << "Invalid input!" << endl;
            break;
        }
    }
    return 0;
}
```

Output 7:

```
1
Enter the element to be inserted : 17

Queue is full
Bushra Shahzad : 21BCS046

Press '1' to enqueue an element.
Press '2' to dequeue an element.
Press '3' to check if queue is full.
Press '4' to check if queue is empty.
Press '5' to display front and rear.
Press '6' to display circular queue!.
Press '7' to exit!.
6

Circular queue :  12 13 14 15 16
Bushra Shahzad : 21BCS046
```

```
Circular queue :  12 13 14 15 16
Bushra Shahzad : 21BCS046

Press '1' to enqueue an element.
Press '2' to dequeue an element.
Press '3' to check if queue is full.
Press '4' to check if queue is empty.
Press '5' to display front and rear.
Press '6' to display circular queue!.
Press '7' to exit!.
2

Bushra Shahzad : 21BCS046

Press '1' to enqueue an element.
Press '2' to dequeue an element.
Press '3' to check if queue is full.
Press '4' to check if queue is empty.
Press '5' to display front and rear.
Press '6' to display circular queue!.
Press '7' to exit!.
6

Circular queue :  13 14 15 16
Bushra Shahzad : 21BCS046
```

```
3
noi, queue is not full!

Bushra Shahzad : 21BCS046

Press '1' to enqueue an element.
Press '2' to dequeue an element.
Press '3' to check if queue is full.
Press '4' to check if queue is empty.
Press '5' to display front and rear.
Press '6' to display circular queue!.
Press '7' to exit!.
4
noi, queue is not empty!

Bushra Shahzad : 21BCS046

Press '1' to enqueue an element.
Press '2' to dequeue an element.
Press '3' to check if queue is full.
Press '4' to check if queue is empty.
Press '5' to display front and rear.
Press '6' to display circular queue!.
Press '7' to exit!.
5
Front = 13
Rear = 16

Bushra Shahzad : 21BCS046
```

Question 8: WAP to implement CBT using array.

Code:

```
#include <iostream>
using namespace std;
struct tree
{
    char array[100];
};
tree tr;
int j = 1;
void create()
{
    char x;
    int i;
    if (j == 1)
    {
        cout << "Enter root value : ";
        cin >> x;
        tr.array[j++] = x;
        return;
    }
    else
    {
        i = j % 2;
        if (i == 0)
        {
            cout << "Enter left child of " << tr.array[j / 2] << " : ";
            cin >> x;
            tr.array[j++] = x;
        }
        if (i == 1)
        {
            cout << "Enter right child of " << tr.array[j / 2] << " : ";
            cin >> x;
            tr.array[j++] = x;
        }
        return;
    }
}
void nodeInfo()
{
    int flag = 0;
    char x;
    cout << "Enter the charachter : ";
    cin >> x;
    for (int i = 1; i <= j; i++)
    {
```

```
        if (tr.array[i] == x)
        {
            flag = 1;
            cout << "Left child of " << x << " is " << tr.array[2 * i] <<
".\n";
            cout << "Right child of " << x << " is " << tr.array[(2 * i) + 1]
<< ".\n";
            cout << "Parent of " << x << " is " << tr.array[i / 2] << ".\n";
            break;
        }
    }
    if (flag == 0)
        cout << "Charchter is not present.\n";
}
int nodes()
{
    return j - 1;
}
void display()
{
    int i = 1;
    cout << "Tree Display : \n";
    if (j >= 1)
        cout << "       " << tr.array[i++] << "      " << endl;
    if (i <= j)
    {
        for (; i <= 3; i++)
            cout << "    " << tr.array[i] << "  ";
        cout << endl;
        if (i <= j)
        {
            for (; i <= 7; i++)
                cout << "  " << tr.array[i] << " ";
            cout << endl;
        }
        if (i <= j)
        {
            for (; i <= 15; i++)
                cout << " " << tr.array[i] << "";
            cout << endl;
        }
        if (i <= j)
        {
            for (; i <= 31; i++)
                cout << "  " << tr.array[i] << " ";
            cout << endl;
        }
    }
```

```
    cout << endl;
}
int main()
{
    for (int i = 0; i < 99; i++)
    {
        tr.array[i] = '-';
    }

    int choice;
    while (1)
    {
        cout << "\nBushra Shahzad : 21BCS046\n";
        cout << "Press 1 to Insert." << endl;
        cout << "Press 2 to get information of any node." << endl;
        cout << "Press 3 to find total no of nodes." << endl;
        cout << "Press 4 to display tree." << endl;
        cout << "Press 5 to exit!" << endl;
        cin >> choice;
        switch (choice)
        {
        case 1:
            create();
            break;
        case 2:
            nodeInfo();
            break;
        case 3:
            cout << "Total number of nodes are : " << nodes() << ".\n";
            break;
        case 4:
            display();
            break;
        case 5:
            exit(1);
            break;

        default:
            cout << "Invalid input.\n";
            break;
        }
    }

    return 0;
}
```

Output 8:

```
Bushra Shahzad : 21BCS046
Press 1 to Insert.
Press 2 to get information of any node.
Press 3 to find total no of nodes.
Press 4 to display tree.
Press 5 to exit!
1
Enter right child of e : i

Bushra Shahzad : 21BCS046
Press 1 to Insert.
Press 2 to get information of any node.
Press 3 to find total no of nodes.
Press 4 to display tree.
Press 5 to exit!
4
Tree Display :
       a
    b      c
  d   e   f   -
 g h - i - - - -
```

```
2
Enter the charachter : d
Left child of d is g.
Right child of d is h.
Parent of d is b.

Bushra Shahzad : 21BCS046
Press 1 to Insert.
Press 2 to get information of any node.
Press 3 to find total no of nodes.
Press 4 to display tree.
Press 5 to exit!
3
Total number of nodes are : 11.
```

Question 9: WAP to implement CBT using linked list.

Code:

```cpp
// to implement complete binary tree using linked list
#include <iostream>
#include <bits/stdc++.h>
using namespace std;
struct node
{
    int data;
    node *right_child;
    node *left_child;
} *root = NULL;
void createBinaryTree()
{
```

```
    node *temp, *newNode;
    int x;
    queue<node *> q;
    cout << "Enter the root: ";
    cin >> x;
    root = new node;
    root->data = x;
    root->left_child = root->right_child = NULL;
    q.push(root);
    while (!q.empty())
    {
        temp = q.front();
        q.pop();
        cout << "Enter the left child of " << temp->data << " : ";
        cin >> x;
        if (x != -1)
        {
            newNode = new node;
            newNode->data = x;
            newNode->left_child = newNode->right_child = NULL;
            temp->left_child = newNode;
            q.push(newNode);
        }
        cout << "Enter the right child of " << temp->data << " : ";
        cin >> x;
        if (x != -1)
        {
            newNode = new node;
            newNode->data = x;
            newNode->left_child = newNode->right_child = NULL;
            temp->right_child = newNode;
            q.push(newNode);
        }
    }
}
void preOrder(node *root)
{

    if (root)
    {
        cout << root->data << " ";
        preOrder(root->left_child);
        preOrder(root->right_child);
    }
}
void inorder(node *root)
{
```

```
    if (root)
    {
        inorder(root->left_child);
        cout << root->data << " ";
        inorder(root->right_child);
    }
}
void postorder(node *root)
{

    if (root)
    {
        postorder(root->left_child);
        postorder(root->right_child);
        cout << root->data << " ";
    }
}
void levelOrder(node *root)
{
    queue<node *> Q;
    node *temp;
    cout << "Level-Order : ";
    cout << root->data << " ";
    Q.push(root);
    while (!Q.empty())
    {
        temp = Q.front();
        Q.pop();
        if (temp->left_child)
        {
            cout << temp->left_child->data << " ";
            Q.push(temp->left_child);
        }
        if (temp->right_child)
        {
            cout << temp->right_child->data << " ";
            Q.push(temp->right_child);
        }
    }
}
int height(node *root)
{
    int x = 0, y = 0;
    if (root == NULL)
        return 0;
    x = height(root->left_child);
    y = height(root->right_child);
    if (x > y)
```

```
        return x + 1;
    else
        return y + 1;
}
int count(node *root)
{
    int x, y;
    if (root != NULL)
    {
        x = count(root->left_child);
        y = count(root->right_child);
        return x + y + 1;
    }
}
int main()
{
    int choice, choice1;
    while (1)
    {
        cout << "\nBushra Shahzad : 21BCS046\n";
        cout << "\nEnter '1' to create binary tree.";
        cout << "\nEnter '2' to traverse nodes of tree.";
        cout << "\nEnter '3' to find height of tree.";
        cout << "\nEnter '4' to count the number of nodes in the tree. \n";
        cin >> choice;
        switch (choice)
        {
        case 1:
            createBinaryTree();
            break;
        case 2:
            cout << "\nEnter '1' for pre-order traversal.";
            cout << "\nEnter '2' for in-order traversal.";
            cout << "\nEnter '3' for post-order traversal.";
            cout << "\nEnter '4' for level-order traversal.";
            cin >> choice1;
            switch (choice1)
            {
            case 1:
                cout << "Pre-Order : ";
                preOrder(root);
                cout << endl;
                break;
            case 2:
                cout << "In-Order : ";
                inorder(root);
                cout << endl;
                break;
```

```
            case 3:
                cout << "Post-Order : ";
                postorder(root);
                cout << endl;
                break;
            case 4:
                cout << "Level-Order : ";
                levelOrder(root);
                cout << endl;
                break;
            }

            break;
        case 3:
            cout << "The height of the tree is " << height(root) << endl;
            break;
        case 4:
            cout << "The no of nodes in tree are " << count(root) << endl;
            break;

        default:
            cout << "Invalid Input!\n";
            break;
        }
    }

    return 0;
}
```

Output 9:

```
Bushra Shahzad : 21BCS046

Enter '1' to create binary tree.
Enter '2' to traverse nodes of tree.
Enter '3' to find height of tree.
Enter '4' to count the number of nodes in the tree.
1
Enter the root: 1
Enter the left child of 1 : 2
Enter the right child of 1 : 3
Enter the left child of 2 : 4
Enter the right child of 2 : 5
Enter the left child of 3 : 6
Enter the right child of 3 : 7
Enter the left child of 4 : 8
Enter the right child of 4 : 9
Enter the left child of 5 : 10
Enter the right child of 5 : -1
Enter the left child of 6 : -1
Enter the right child of 6 : -1
Enter the left child of 7 : -1
Enter the right child of 7 : -1
Enter the left child of 8 : -1
Enter the right child of 8 : -1
Enter the left child of 9 : -1
Enter the right child of 9 : -1
Enter the left child of 10 : -1
Enter the right child of 10 : -1
```

```
Pre-Order : 1 2 4 8 9 5 10 3 6 7

Bushra Shahzad : 21BCS046
```

```
In-Order : 8 4 9 2 10 5 1 6 3 7

Bushra Shahzad : 21BCS046
```

```
Post-Order : 8 9 4 10 5 2 6 7 3 1

Bushra Shahzad : 21BCS046
```

```
Level-Order : Level-Order : 1 2 3 4 5 6 7 8 9 10

Bushra Shahzad : 21BCS046
```

```
3
The height of the tree is 4

Bushra Shahzad : 21BCS046

Enter '1' to create binary tree.
Enter '2' to traverse nodes of tree.
Enter '3' to find height of tree.
Enter '4' to count the number of nodes in the tree.
4
The no of nodes in tree are 10

Bushra Shahzad : 21BCS046
```

Assignment Questions

Question 10: Conversion of number system to all.

Code:

```
#include <iostream>

#include <bits/stdc++.h>
using namespace std;
void dec_to_bin(int no)
{   int decimalNo=no;
    stack<char> remainders;
    int rem, i = 0;
    char *output;

    while (no != 0)
    {
        rem = no % 2;
        if (rem >= 10)
            rem += 55;
        else
            rem += 48;

        remainders.push(rem);
        no = no / 2;
    };
    cout << "The binary no. for " << decimalNo << " is ";
    while (!remainders.empty())
    {
        cout << remainders.top();
        remainders.pop();
    }
    cout << "\n";
}
void dec_to_oct(int no)
{   int decimalNo=no;
    stack<char> remainders;
    int rem, i = 0;
    char *output;

    while (no != 0)
    {
        rem = no % 8;
        if (rem >= 10)
            rem += 55;
        else
            rem += 48;

        remainders.push(rem);
```

```
        no = no / 8;
    };
    cout << "The octal no. for " << decimalNo << " is ";
    while (!remainders.empty())
    {
        cout << remainders.top();
        remainders.pop();
    }
    cout << "\n";
}
void dec_to_hexadecimal(int no)
{   int decimalNo=no;
    stack<char> remainders;
    int rem, i = 0;
    char *output;

    while (no != 0)
    {
        rem = no % 16;
        if (rem >= 10)
            rem += 55;
        else
            rem += 48;

        remainders.push(rem);
        no = no / 16;
    };
    cout << "The hexa decimal no. for " << decimalNo << " is ";
    while (!remainders.empty())
    {
        cout << remainders.top();
        remainders.pop();
    }
    cout << "\n";
}
void dec_to_any_base(int no,int base)
{   int decimalNo=no;
    stack<char> remainders;
    int rem, i = 0;
    char *output;

    while (no != 0)
    {
        rem = no % base;
        if (rem >= 10)
            rem += 55;
        else
            rem += 48;
```

```
        remainders.push(rem);
        no = no / base;
    };
    cout << "The no. in base "<<base<<" for " << decimalNo << " is ";
    while (!remainders.empty())
    {
        cout << remainders.top();
        remainders.pop();
    }
    cout << "\n";
}
int to_decimal(char *no, int base)
{
    int i, ans = 0, factor = 1, count = 0;
    for (i = 0; no[i] != '\0'; i++)
        count++;
    for (i = count - 1; i >= 0; i--)
    {
        if (no[i] >= 'A' && no[i] <= 'F')
            ans += (no[i] - 55) * factor;
        else if (no[i] >= 'a' && no[i] <= 'f')
            ans += (no[i] - 87) * factor;
        else if (no[i] >= '0' && no[i] <= '9')
            ans += (no[i] - 48) * factor;
        factor *= base;
    }
    return ans;
}
int main()
{

    int choice, base, num_decimal, deci, base2, count = 0;
    char num_any_base[100];
    char *answer_decimal;
    while (1)
    {
         cout << "\nBushra Shahzad : 21BCS046\n";
        cout << "\n1. Decimal to Binary.";
        cout << "\n2. Decimal to Octal.";
        cout << "\n3. Decimal to Hexadecimal.";
        cout << "\n4. Decimal to any base.";
        cout << "\n5. Binary to decimal.";
        cout << "\n6. Octal to decimal";
        cout << "\n7. Hexadecimal to decimal.";
        cout << "\n8. Any base to decimal.";

        cout << "\n9. to Exit!\n";
```

```
        cin >> choice;
        switch (choice)
        {
        case 1:
            cout << "Enter the decimal no. : ";
            cin >> num_decimal;
            dec_to_bin(num_decimal);

            break;
        case 2:
            cout << "Enter the decimal no. : ";
            cin >> num_decimal;
            dec_to_oct(num_decimal);


            break;
        case 3:
            cout << "Enter the decimal no. : ";
            cin >> num_decimal;
           dec_to_hexadecimal(num_decimal);

            break;
        case 4:
            cout << "Enter the no. : ";
            cin >> num_decimal;
            cout << "Enter the base of the no. in which you want to convert :
";
            cin >> base;
            dec_to_any_base(num_decimal,base);

            break;
        case 5:
            cout << "Enter the binary no. : ";
            cin >> num_any_base;
            cout << "The decimal no. for " << num_any_base << " is " <<
to_decimal(num_any_base, 2) << endl;
            break;
        case 6:
            cout << "Enter the octal no. : ";
            cin >> num_any_base;
            cout << "The decimal no. for " << num_any_base << " is " <<
to_decimal(num_any_base, 8) << endl;
            break;
        case 7:
            cout << "Enter the hexadecmial no. : ";
            cin >> num_any_base;
            cout << "The decimal no. for " << num_any_base << " is " <<
to_decimal(num_any_base, 16) << endl;
```

```
            break;
        case 8:
            cout << "Enter the base of the no.in which you want to convert :
";
            cin >> base;
            cout << "Enter the no. : ";
            cin >> num_any_base;
            cout << "The decimal no. for " << num_any_base << " is " <<
to_decimal(num_any_base, base) << endl;
            break;
        case 9:
            exit(1);

            break;

        default:
            cout << "Invalid input!\n";
            break;
        }
    }

    return 0;
}
```

Output 10:

```
Enter the decimal no. : 123
The binary no. for 123 is 1111011

Bushra Shahzad : 21BCS046

1. Decimal to Binary.
2. Decimal to Octal.
3. Decimal to Hexadecimal.
4. Decimal to any base.
5. Binary to decimal.
6. Octal to decimal
7. Hexadecimal to decimal.
8. Any base to decimal.
9. to Exit!
2
Enter the decimal no. : 123
The octal no. for 123 is 173

Bushra Shahzad : 21BCS046
```

```
3
Enter the decimal no. : 123
The hexa decimal no. for 123 is 7B

Bushra Shahzad : 21BCS046

1. Decimal to Binary.
2. Decimal to Octal.
3. Decimal to Hexadecimal.
4. Decimal to any base.
5. Binary to decimal.
6. Octal to decimal
7. Hexadecimal to decimal.
8. Any base to decimal.
9. to Exit!
4
Enter the no. : 123
Enter the base of the no. in which you want to convert : 9
The no. in base 9 for 123 is 146

Bushra Shahzad : 21BCS046
```

```
5
Enter the binary no. : 10
The decimal no. for 10 is 2

Bushra Shahzad : 21BCS046

1. Decimal to Binary.
2. Decimal to Octal.
3. Decimal to Hexadecimal.
4. Decimal to any base.
5. Binary to decimal.
6. Octal to decimal
7. Hexadecimal to decimal.
8. Any base to decimal.
9. to Exit!
6
Enter the octal no. : 173
The decimal no. for 173 is 123
```

```
7
Enter the hexadecmial no. : 7b
The decimal no. for 7b is 123

Bushra Shahzad : 21BCS046

1. Decimal to Binary.
2. Decimal to Octal.
3. Decimal to Hexadecimal.
4. Decimal to any base.
5. Binary to decimal.
6. Octal to decimal
7. Hexadecimal to decimal.
8. Any base to decimal.
9. to Exit!
8
Enter the base of the no.in which you want to convert : 9
Enter the no. : 146
The decimal no. for 146 is 123

Bushra Shahzad : 21BCS046
```

Question 11: WAP to implement doubly linked list.

Code:

```
#include <iostream>
#include <string.h>
using namespace std;

struct node
{
    int id;
    char name[50];
    struct node *prev;
    struct node *next;
} *head = NULL;

void print()
{
    if (head == NULL)
    {
        cout << "\nEmpty List!\n";
        return;
    }
    struct node *temp;
    temp = head;

    while (temp != NULL)
    {
        cout << "[" << temp->id << " , " << temp->name << "]";
        if (temp->next != NULL)
        {
            cout << " <-> ";
        }

        temp = temp->next;
    }
    cout << endl;
}

void insertHead()
{

    struct node *temp;
    temp = new node;
    cout << "Enter the data to be inserted:\n";
    cout << "ID : ";
    cin >> temp->id;
    cout << "Name : ";
    getchar();
```

```
    cin.getline(temp->name, 50);
    temp->prev = NULL;
    temp->next = head;
    head = temp;
}
void insertAtEnd()
{
    struct node *temp, *traversal;

    temp = new node;
    temp->prev = NULL;
    temp->next = NULL;
    cout << "Enter the data to be inserted:\n";
    cout << "ID : ";
    cin >> temp->id;
    cout << "Name : ";
    getchar();
    cin.getline(temp->name, 50);
    traversal = head;
    if (head == NULL)
    {
        head = temp;
    }
    else
    {
        while (traversal->next != NULL)
        {
            traversal = traversal->next;
        }
        temp->prev = traversal;
        traversal->next = temp;
    }
}

void insertAtPosition()
{
    int pos, i = 1;
    cout << "Enter the position where data is to be inserted:\n";
    cin >> pos;
    struct node *temp, *traversal;
    traversal = head;
    temp = new node;
    temp->next = NULL;
    temp->prev = NULL;
    cout << "Enter the data to be inserted:\n";
    cout << "ID : ";
    cin >> temp->id;
    cout << "Name : ";
```

```
    getchar();
    cin.getline(temp->name, 50);
    if (head == NULL)
    {
        head = temp;
    }
    else if (pos == 1)
    {
        temp->next = head;
        temp->next->prev = temp;
        temp->prev = NULL;
        head = temp;
    }
    else
    {
        while (i < pos - 1)
        {
            traversal = traversal->next;
            i++;
        }
        temp->next = traversal->next;
        temp->prev = traversal;
        traversal->next = temp;
        traversal->next->prev = temp;
    }
}

void deleteHead()
{
    struct node *temp;
    if (head == NULL)
    {
        cout << "\nList is empty!\n";
    }
    else
    {
        temp = head;
        head = head->next;
        if (head != NULL)
        {
            head->prev = NULL;
        }
        delete temp;
    }
}

void deleteEnd()
{
```

```
    struct node *temp;
    if (head == NULL)
    {
        cout << "\nList is empty!\n";
    }
    temp = head;
    while (temp->next != NULL)
    {
        temp = temp->next;
    }
    if (head->next == NULL)
    {
        head = NULL;
    }
    else
    {
        temp->prev->next = NULL;
        delete temp;
    }
}

void deletePos()
{
    int pos, i = 1;
    struct node *temp, *traversal;
    traversal = head;
    if (head == NULL)
        cout << "\nList is empty!\n";
    else
    {
        cout << "Enter position: ";
        cin >> pos;
        if (pos == 1)
        {
            temp = head;
            head = head->next;
            if (head != NULL)
            {
                head->prev = NULL;
            }
            delete temp;
            return;
        }
        while (i < pos - 1)
        {
            traversal = traversal->next;
            i++;
        }
```

```
        temp = traversal->next;
        if (temp->next != NULL)
        {
            temp->next->prev = traversal;
        }
        traversal->next = temp->next;
        delete temp;
    }
}
void search()
{
    int key, i = 1, flag = 0;

    struct node *temp;
    temp = head;
    cout << "Enter the ID you want to search : ";
    cin >> key;
    while (temp != NULL)
    {
        if (temp->id == key)
        {
            flag = 1;

            break;
        }
        temp = temp->next;
        i++;
    }
    if (flag == 1)
    {
        cout << "The ID " << key << " is present at position " << i << " as ["
<< temp->id << " , " << temp->name << " ].\n";
    }
    else
    {
        cout << "No such ID as " << key << "is present!\n";
    }
}
int compare(char *a, char *b)
{
    int x = 1;
    for (int i = 0; a[i] != '\0' && b[i] != '\0'; i++)
    {
        if (a[i] != b[i])
        {
            x = 0;
            break;
        }
```

```
    }
    return x;
}
void searchName()
{
    int i = 1, flag = 0;
    char keyName[50];
    struct node *temp;
    temp = head;
    cout << "Enter the Name you want to search : ";
    getchar();
    cin.getline(keyName, 50);
    while (temp != NULL)
    {
        if (compare(keyName, temp->name))
        {
            flag = 1;

            break;
        }
        temp = temp->next;
        i++;
    }
    if (flag == 1)
    {
        cout << "The name '" << keyName << "' is present at position " << i <<
" as [" << temp->id << " , " << temp->name << " ].\n";
    }
    else
    {
        cout << "No such name as '" << keyName << "' is present!\n";
    }
}
void reversePrint(struct node *head)
{
    if (head == NULL)
    {
        cout << "\nEmpty List!\n";
        return;
    }
    struct node *temp;
    temp = head;

    reversePrint(temp->next);
    cout << "[" << temp->id << " , " << temp->name << "]";
    if (temp->next != NULL)
    {
        cout << " <-> ";
```

```
    }
}
int main()
{
    int choice;
    while (1)
    {
        cout << "\nBushra Shahzad : 21BCS046\n";
        cout << "\nEnter '1' to insert at beginning.\n";
        cout << "Enter '2' to insert at end.\n";
        cout << "Enter '3' to insert at given position.\n";
        cout << "Enter '4' to delete from beginning.\n";
        cout << "Enter '5' to delete from end.\n";
        cout << "Enter '6' to delete from given position.\n";
        cout << "Enter '7' to search by ID\n";
        cout << "Enter '8' to search by Name.\n";
        cout << "Enter '9' to print in reverse order.\n";
        cout << "Enter '10' to exit.\n";
        cin >> choice;
        switch (choice)
        {
        case 1:
            insertHead();
            print();
            break;
        case 2:
            insertAtEnd();
            print();
            break;
        case 3:
            insertAtPosition();
            print();
            break;
        case 4:
            deleteHead();
            print();
            break;
        case 5:
            deleteEnd();
            print();
            break;
        case 6:
            deletePos();
            print();
            break;
        case 7:
            search();
            break;
```

```
        case 8:
            searchName();
            break;
        case 9:
            reversePrint(head);

            break;
        case 10:
            exit(1);
            break;

        default:
            cout << "\nInvalid!\n";
            break;
        }
    }

    return 0;
}
```

Output 11:

```
1
Enter the data to be inserted:
ID : 1
Name : bushra
[1 , bushra]

Bushra Shahzad : 21BCS046

Enter '1' to insert at beginning.
Enter '2' to insert at end.
Enter '3' to insert at given position.
Enter '4' to delete from beginning.
Enter '5' to delete from end.
Enter '6' to delete from given position.
Enter '7' to search by ID
Enter '8' to search by Name.
Enter '9' to print in reverse order.
Enter '10' to exit.
1
Enter the data to be inserted:
ID : 2
Name : shahzad
[2 , shahzad] <-> [1 , bushra]

Bushra Shahzad : 21BCS046
```

```
2
Enter the data to be inserted:
ID : 20
Name : alex
[2 , shahzad] <-> [1 , bushra] <-> [20 , alex]

Bushra Shahzad : 21BCS046

Enter '1' to insert at beginning.
Enter '2' to insert at end.
Enter '3' to insert at given position.
Enter '4' to delete from beginning.
Enter '5' to delete from end.
Enter '6' to delete from given position.
Enter '7' to search by ID
Enter '8' to search by Name.
Enter '9' to print in reverse order.
Enter '10' to exit.
3
Enter the position where data is to be inserted:
2
Enter the data to be inserted:
ID : 45
Name : ryan
[2 , shahzad] <-> [45 , ryan] <-> [1 , bushra] <-> [20 , alex]

Bushra Shahzad : 21BCS046
```

```
[2 , shahzad] <-> [45 , ryan] <-> [23 , franz] <-> [1 , bushra] <-> [20 , alex]

Bushra Shahzad : 21BCS046

Enter '1' to insert at beginning.
Enter '2' to insert at end.
Enter '3' to insert at given position.
Enter '4' to delete from beginning.
Enter '5' to delete from end.
Enter '6' to delete from given position.
Enter '7' to search by ID
Enter '8' to search by Name.
Enter '9' to print in reverse order.
Enter '10' to exit.
4
[45 , ryan] <-> [23 , franz] <-> [1 , bushra] <-> [20 , alex]

Bushra Shahzad : 21BCS046

Enter '1' to insert at beginning.
Enter '2' to insert at end.
Enter '3' to insert at given position.
Enter '4' to delete from beginning.
Enter '5' to delete from end.
Enter '6' to delete from given position.
Enter '7' to search by ID
Enter '8' to search by Name.
Enter '9' to print in reverse order.
Enter '10' to exit.
5
[45 , ryan] <-> [23 , franz] <-> [1 , bushra]

Bushra Shahzad : 21BCS046
```

```
7
Enter the ID you want to search : 23
The ID 23 is present at position 2 as [23 , franz ].

Bushra Shahzad : 21BCS046

Enter '1' to insert at beginning.
Enter '2' to insert at end.
Enter '3' to insert at given position.
Enter '4' to delete from beginning.
Enter '5' to delete from end.
Enter '6' to delete from given position.
Enter '7' to search by ID
Enter '8' to search by Name.
Enter '9' to print in reverse order.
Enter '10' to exit.
8
Enter the Name you want to search : bushra
The name 'bushra' is present at position 3 as [1 , bushra ].

Bushra Shahzad : 21BCS046
```

```
[45 , ryan] <-> [23 , franz] <-> [1 , bushra]

Bushra Shahzad : 21BCS046

Enter '1' to insert at beginning.
Enter '2' to insert at end.
Enter '3' to insert at given position.
Enter '4' to delete from beginning.
Enter '5' to delete from end.
Enter '6' to delete from given position.
Enter '7' to search by ID
Enter '8' to search by Name.
Enter '9' to print in reverse order.
Enter '10' to exit.
9

Empty List!
[1 , bushra] <-> [23 , franz] <-> [45 , ryan] <->
Bushra Shahzad : 21BCS046
```

Question 12: WAP to implement XOR linked list.

Code:

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

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

struct Node *XOR(Node *a, Node *b)
{
    return (struct Node *)((uintptr_t)(a) ^ (uintptr_t)(b));
}

void forwardTraverse(Node *head)
{
    Node *curr = head;
    Node *prev = nullptr;
    Node *next;

    while (curr != nullptr)
    {
        cout << curr->data<<" -> ";
        next = XOR(prev, curr->link);
        prev = curr;
        curr = next;
    }
    // cout << "\n";
}

void backwardTraverse(Node *head)
{
    Node *curr = head;
    Node *prev = nullptr;
    Node *next;

    while (curr != nullptr)
    {
        next = XOR(prev, curr->link);
        prev = curr;
        curr = next;
    }
    while (prev != nullptr)
    {
        cout << prev->data << " <- ";
```

```
        Node *next = XOR(curr, prev->link);
        curr = prev;
        prev = next;
    }
}

void insertFront(Node *&headRef, int data)
{
    Node *newNode = new Node();
    newNode->data = data;
    newNode->link = XOR(headRef, nullptr);
    if (headRef)
    {
        headRef->link = XOR(newNode, XOR(headRef->link, nullptr));
    }
    headRef = newNode;
}

void deleteFront(Node *&headRef)
{
    Node *temp;
    if (!headRef)
    {
        cout << "\nList is empty!\n";
    }
    else
    {
        temp = headRef;
        headRef = XOR(nullptr, temp->link);
        if (headRef)
        {
            headRef->link = XOR(nullptr, XOR(temp, headRef->link));
        }
        delete temp;
    }
}

int main()
{
    int ch, data;
    Node *head = nullptr;

    while (1)
    {
        cout << "\nBushra Shahzad : 21BCS046\n";
        cout << "\nPress '1' to insert at beginning.\n";
        cout << "Press '2' to delete from beginning.\n";
        cout << "Press '3' to display in forward direction\n";
```

```
        cout << "Press '4' to display in backward direction.\n";
        cout << "Press '5' to Exit\n";
        cin >> ch;
        switch (ch)
        {
        case 1:
        {

            cout << "\nEnter data : ";
            cin >> data;
            insertFront(head, data);
            break;
        }
        case 2:
        {
            deleteFront(head);
            break;
        }
        case 3:
        {
            forwardTraverse(head);
            break;
        }
        case 4:
        {
            backwardTraverse(head);
            break;
        }
        case 5:
        {
            exit(1);
            break;
        }
        default:
        {
            break;
        }
        }
    }
    return 0;
}
```

Output 12:

```
1

Enter data : 16

Bushra Shahzad : 21BCS046

Press '1' to insert at beginning.
Press '2' to delete from beginning.
Press '3' to display in forward direction
Press '4' to display in backward direction.
Press '5' to Exit
3
16 -> 15 -> 14 -> 13 -> 12 ->
Bushra Shahzad : 21BCS046

Press '1' to insert at beginning.
Press '2' to delete from beginning.
Press '3' to display in forward direction
Press '4' to display in backward direction.
Press '5' to Exit
4
12 <- 13 <- 14 <- 15 <- 16 <-
Bushra Shahzad : 21BCS046
```

```
2

Bushra Shahzad : 21BCS046

Press '1' to insert at beginning.
Press '2' to delete from beginning.
Press '3' to display in forward direction
Press '4' to display in backward direction.
Press '5' to Exit
3
15 -> 14 -> 13 -> 12 ->
Bushra Shahzad : 21BCS046

Press '1' to insert at beginning.
Press '2' to delete from beginning.
Press '3' to display in forward direction
Press '4' to display in backward direction.
Press '5' to Exit
4
12 <- 13 <- 14 <- 15 <-
Bushra Shahzad : 21BCS046
```