Page No.

Roll No.

Date

Program 1

Write a program to perform insertion on array.

```
#include <iostream>
using namespace std;
int insertion(int lb, int ub, int in, int m,
              int item, list[])
{   int i;
    if (ub > max || index > ub)
    {
        cout << "\n error";
        return 0;
    }
    for (i = ub; i > in; i--)
    list[i] = list[i-1];
    list[in] = item;
    cout << "\n After insertion: ";
    for (int i = lb; i <= ub; i++)
    cout << list[i] << " ";

    return 0;
}
```

Page No.

Roll No.

Date

```
int main()
{
    int lb, int ub, in, max, item;
    cout << "Enter size of array: ";
    cin >> max;
    int ~~max~~ list [max];

    cout << "Enter lb: ";
    cin >> lb;
    cout << "Enter ub: ";
    cin >> ub;

    cout << "Enter elements of array: ";
    for (int i = lb; i <= ub; i++)
    cin >> list [i];

    cout << "Enter item to be inserted: ";
    ~~cout~~ cin >> ~~int~~ item;

    cout << "Enter index number: ";
    cin >> in;

    cout << "\n Before insertion: ";
    for (int i = lb; i <= ub; i++)
    cout << list[i] << " ";

    insertion (lb, ub, in, max, item, list);
    return 0;
}
```

OUTPUT:

```
Enter size of array : 5
Enter lb : 0
Enter ub : 4
Enter elements of array: 2 3 1 6 8
Enter item to be inserted : 5
Enter index number : 0

Before insertion : 2 3 1 6 8
After insertion : 5 2 3 1 6
```

Page No.

Roll No.

Date

Program - 2

Aim :- Write a program to perform deletion on array.

```
#include <iostream>
using namespace std;
void print(int lb, int ub, int list[])
{
    for(int i=lb; i<=ub; i++)
    cout<< list[i] << " ";
}
int deletion(int lb, int ub, int index, int
        max, int list[])
{
    if(index<lb || ub>max || index>ub)
    {
        cout<<"error";
        return 0;
    }
    for(int i= index; i<=ub; i++)
    list[i] = list[i+1];
    list[ub] = 0;
    cout<<"\n After Deletion:";
    print(lb, ub, list);
    return 0;
}
```

Page No.

Date

Roll No.

```
int main()
{
    int lb, ub, max, index;
    cout << " Enter size of array ;
    cin >> max;
    int list[max];
    Cout << " Enter lb: " ;
    cin >> lb;
    Cout << " Enter ub: " ;
    cin >> ub;
    Cout << " Enter array elements : " ;
    for( int i = lb ; i<=ub ; i++)
    cin >> list[i];

    cout << " Enter index number you want
            to delete:" ;
    ~~printf~~
    cin >> index ;
    Cout << " \n Before deletion: " ;
    print ( lb , ub , list );

    deletion (lb, ub , index , max , list);
    return 0;
}
```

OUTPUT:-

```
Enter size of array: 5
Enter lb: 0
Enter ub: 4
Enter array elements: 1
2
3
4
5
Enter index number you want to delete: 2

Before deletion: 1 2 3 4 5
After deletion: 1 2 4 5 0
```

Page No.

Roll No.

Date

Program - 3

Write a program to perform reverse of array

```
#include<iostream>
using namespace std;
int reverse(int lb, int ub, int max, int list[])
{
    if (ub>max)
    {
        cout<<"error";
        return 0;
    }
    int temp
    for(int i=lb, j=ub; i<=ub/2; i++, j--)
    {
        temp = list[i];
        list[i] = list[j];
        list[j] = temp;
    }
    cout<<"\n After reversing:";
    for(int i=lb; i<=ub; i++)
    cout<<list[i]<<"  ";
    return 0;
}
```

ACE

Page No.

Roll No.

Date

```
int main()
{
    int lb, ub, max;
    cout<<"Enter size of array:";
    cin>>max;
    int list[max];
    cout<<"Enter lb:";
    cin>>lb;
    cout<<"Enter ub:";
    cin>>ub;

    cout<<"Enter elements of array:";
    for(int i=lb; i<=ub; i++)
    cin>>list[i];

    cout<<"\n Before reversing:";
    for(int i=lb; i<=ub; i++)
    cout<<list[i]<<" ";

    reverse(lb, ub, max, list);
    return 0;
}
```

Output :-

```
Enter size of array : 7
Enter lb : 0
Enter ub : 6
Enter elements of array : 1
2
3
4
5
6
7

Before reversing : 1 2 3 4 5 6 7
After reversing : 7 6 5 4 3 2 1
```

Page No.

Roll No.

Date

Program 4 :-

Write a program to perform binary search on linear array.

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

int binarysearch (int, int, int[], int, int);
void elements (int, int[]);
void sort (int, int, int[]);
void disp (int, int, int[]);

int main()
{
    int max, lb, ub, item;
    cout << " Enter size of array: ";
    cin >> max;
    int list [max];

    cout << " Enter lb: ";
    cin >> lb;
    cout << " Enter ub: ";
    cin >> ub;
    if ( lb > ub || ub >= max )
    { cout << " error\n";
      return 0;
    }
```

Page No.

Roll No.

Date

```
cout << " enter elements of array: ";
elements (max, list);

cout << " \n before sorting \n ";
disp (lb, ub, list);
sort (lb, ub, list);
cout << " \n After sorting \n ";
disp (lb, ub, list);

cout << " \n Enter search item: ";
cin >> sitem;
int result = binarysearch (lb, ub, list, max, sitem);

if (result == -1)
    cout << " search item not found ";
else
    cout << " search item found in index
            number: " << result;
return 0;

void elements (int max, int list[])
{
    for (int i=0; i<=max-1; i++)
    cin >> list [i];
}
```

Page No.

Roll No.

Date

```
int binarysearch (int lb, int ub, int list[ ], int
                  max, int sitem)
{
    while( lb<=ub)
    {
      int mid =(ub+lb)/2;
      if (list[mid]==sitem)
       {
          return mid;
       }
      else if ( list [mid] < sitem)
        {  lb= mid+1;
        }
      else
        {  ub = mid-1;
        }
    }
    return -1;
}
void sort(int lb, int ub, int list[ ])
{
    for(int i= lb; i<=ub ; i++)
    {
      for(int j=i+1; j<=ub ; j++)
      {
        if ( list[j] < list [i] )
        {  int temp = list [i];
           list[ i ] = list [j]
           list[j] = temp;
        }
```

Page No.

Roll No.

Date

```
    }
  }
}

void disp(int lb, int ub, int list[])
{
    for(int i = lb; i<=ub; i++)
    cout<< list[i]<< " ";
}
```

OUTPUT:-

```
Enter size of array : 5
Enter lb : 0
Enter ub : 4
Enter elements of array: 2
 8
 1
 9
 4

before sorting
2 8 1 9 4
After sorting
1 2 4 8 9
Enter search item : 8
Search item found in index number: 3
```

Page No.

Roll No.

Date

Program 5

Write a program to perform Sequential search on linear array.

```
#include <iostream>
using namespace std;
int Sequentialsearch(int lb, int ub, int max,
                     int list[], int sitem)
{
    if (lb>ub || ub>=max)
    {
        cout << "error";
        return -1;
    }
    int temp = 1;
    for (int i=lb; i<=ub; i++)
    {
        if (list[i] == sitem)
        {
            temp = 0
            return i;
        }
    }
    if (temp == 1)
    return -1;
}
```

Page No.

Roll No.

Date

```
int main()
{
    int lb, ub, max, sitem;
    cout << " Enter size of array: ";
    cin >> max;
    int list [max];
    cout << " Enter lb: ";
    cin >> lb;
    cout << " Enter ub: ";
    cin >> ub;
    cout << " Enter elements of array: ";
    for (int i = lb ; i <= ub; i++)
    cin >> list [i]
    cout << " Enter search item: ";
    cin >> sitem;

    int result = sequentialsearch(lb, ub, max,
                                  list, sitem);
    if ( result == -1)
    cout << "\nsearch item not found ";
    else
    cout << "\n search item ~~not~~ found in
            index number: " << result;
    return 0;
}
```

OUTPUT :-

```
Enter size of array : 5
Enter lb : 0
Enter ub : 4
Enter elements of array : 8
5
9
4
3
Enter search item : 8
Search item found in index number : 0
```

Page No.

Roll No.

Date

Program - 6

Write a program to perform insertion sort on array.

```
#include <iostream>
using namespace std;
int insertion_sort(int lb, int ub, int max, int *list[])
{
    if (ub > max)
    {
        cout << "error";
        return 0;
    }
    int i, temp, j;
    i = lb;
    while (i<ub)
    {
        j = i;
        while (j >= lb)
        {
            if (list[j] > list[j+1])
            {
                temp = list[j];
                list[j] = list[j+1];
                list[j+1] = temp;
            }
```

Page No.

Roll No.

Date

```
            j--;
        }
        i++;
    }
    return 0;
}
int main
{
    int lb, ub, max;
    cout << "Enter size: ";
    cin >> max;
    cout << "Enter lb: ";
    cin >> lb;
    cout << "Enter ub: ";
    cin >> ub;
    int list[max];
    cout << "Enter array elements: ";
    for(int i=lb; i<=ub; i++)
    cin >> list[i];
    cout << "\n Before sorting: ";
    for(int i=lb; i<=ub; i++)
    cout << list[i] << "  ";
    insertion_sort(lb, ub, max, list);
    cout << "\n After sorting: ";
    for(int i=lb; i<=ub; i++)
    cout << list[i] << "  ";
    return 0;
}
```

OUTPUT:-

```
Enter size : 5
Enter lb : 0
Enter ub : 4
Enter array elements : 55
 33
 77
 22
 11

Before sorting : 55 33 77 22 11
After sorting : 11 22 33 55 77
```

Page No.

Roll No.

Date

Program-7

Write a program to perform bubble sort on input array.

```
#include <iostream>
using namespace std;
int bubble_Sort(int lb, int ub, int max, int list[])
{
    int i = lb;
    while(i<=ub)
    {
        int j = lb;
        while(j<ub)
        {
            if(list[j]> list[j+1])
            {
                int temp = list[j];
                list[j] = list[j+1];
                list[j+1] = temp;
            }
        j++;
    }
    i++;
}
return 0;
}
```

Page No.

Roll No.

Date

```
int main()
{
    int lb, ub, max;
    cout << "Enter size of array: ";
    cin >> max;
    cout << "Enter lb: ";
    cin >> lb;
    cout << "Enter ub: ";
    cin >> ub;
    int list[max];

    cout << "Enter elements of array: ";
    for (int i = lb; i <= ub; i++)
    cin >> list[i];

    cout << "\nBefore Sorting: ";
    for(int i = lb; i <= ub; i++)
    cout << list[i] << "  ";

    bubble_sort(lb, ub, max, list);

    cout << "\nAfter Sorting: ";
    for(int i = lb; i <= ub; i++)
    cout << list[i] << "  ";
    return 0;
}
```

OUTPUT:

```
Enter size of array : 5
Enter lb : 0
Enter ub : 4
Enter elements of array : 88
 11
 44
 55
 22

Before Sorting : 88  11  44  55  22
After Sorting : 11  22  44  55  88
```

Page No.

Roll No.

Date

Program 8:-

Write a program to perform to perform selection sort on input array.

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

int selection_sort(int lb, int ub, int max, int list[])
{
    int i, j, temp;
    if (ub > max)
    {
        cout << " error ";
        return 0;
    }
    i = lb;
    while (i<ub)
    {
        j = i+1
        while (j<=ub)
        {
            if (list[i] > list[j])
            {
                temp = list[j];
                list[j] = list[i];
                list[i] = temp;
            }
```

Page No.

Roll No.

Date

```
        j++;
    }
    i++;
}
return 0;
}
int main()
{
    int lb, ub, max;
    cout<<"Enter size of array:";
    cin>>max;
    int list[max];
    cout<<"Enter lb:";
    cin>>lb;
    cout<<"Enter ub:";
    cin>>ub;
    cout<<"Enter array elements:";
    for(int i=lb; i<=ub; i++)
    cin>>list[i];

    cout<<"\n Before Sorting:";
    for(int i=lb; i<=ub; i++)
    cout<<list[i]<<"  ";

    Selection_Sort(lb, ub, max, list);
    cout<<"\nAfter Sorting:";
    for(int i=lb; i<=ub; i++)
    cout<<list[i]<<"  ";
    return 0;
}
```

OUTPUT :-

```
Enter size of array : 5
Enter lb : 0
Enter ub : 4
Enter array elements : 8
5
9
3
2

Before sorting : 8 5 9 3 2
After sorting : 2 3 5 8 9
```

Page No.

Roll No.

Date

Program-9

Write a program to implement stack and perform push and pop operation.

```
#include <iostream>
using namespace std;
#define max 100

class stackOP
{
    int top;
  Public:
    int stack[max];

    stackOP()          // Constructor
    { top = -1;
    }
  // function declaration section
    int push(int item);
    int pop();
    int display();

};
```

Page No.

Roll No.

Date

```
// function definition section for push
int stackop :: push (int item)
{
    if (top >= max)
    {
        cout "Overflow";
        return 0;
    }

    top = top+1
    stack [top] = item;
    return 0;
}

// function definition for pop
int stackop :: pop( )
{
    if (top < 0)
    {
        cout << "underflow";
        return 0;
    }
    int popitem = stack [top];
    top = top - 1;
    return popitem;
}
```

Page No.

Roll No.

Date

```
int stackop :: display()
{
    if (top<0)
    {
        cout << "Stack is empty";
        return 0;
    }
    int x = stack[top]
    return x;
}

int main()
{
    stackop s;
    s.push(10);
    s.push(20);
    s.push(30);

    cout << s.pop() << ": popped from stack\n";
    cout << s.display() << endl;
    return 0;
}
```

OUTPUT :-

30 : popped from stack
20

Page No.

Roll No.

Date

Program 10

Write a program to perform Traverse, insertion, deletion over singly link list.

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

class node
{
    Public :
     int data;
     node * link;
};

class list
{
   node* Start;
   Public:
    list()
    {
       Start = NULL;
    }
   int traverse();
   int insertion(int item);
   int isempty();
```

Page No.

Roll No.

Date

```
    int deletion();
};

int list::isempty()
{
    if (start==NULL)
        return 1;
    else
        return 0;
}

int list::traverse()
{
    if (isempty())
    {
        cout<<"list is empty\n";
        return -1;
    }
    node *ptr = new node;
    ptr = start;
    while (ptr!=NULL)
    {
        cout<<ptr->data<<"  ";
        ptr = ptr->link;
    }
}
```

Page No.

Roll No.

Date

```
int list :: insertion (int item)
{
    node * newnode = new node();
    newnode -> data = item;
    if (isempty())
    {
        start = newnode;
        start -> link = NULL;
    }
    else
    {
        node * ptr = start;
        while(1)
        {
            if (ptr -> link == NULL)
            {
                ptr -> link = newnode;
                newnode -> link = NULL;
                break;
            }
            ptr = ptr -> link;
        }
    }
}
```

Page No.

Roll No.

Date

```
int list :: deletion()
{
    if(isempty())
    {
        cout<<" list is empty";
        return -1;
    }
    node * ptr = start;
    int x = ptr -> data;
    start = ptr -> link;
    return x;
}
int main()
{
    list l1;
    l1.insertion(10);
    l1.insertion(20);
    l1.insertion(30);
    cout<<" Before deletion:";
    l1.traverse();
    cout<<"\n"<<l1.deletion()<<": deleted item\n";
    cout<<" After deletion:";
    l1.traverse();
    return 0;
}
```

OUTPUT:-

```
Before deletion : 10  20  30
10 : deleted item
After deletion : 20  30
```

Page No.

Roll No.

Date

Program - 11

Write a program to convert infix to postfix expression.

```
#include <iostream>
#include< stack>
using namespace std;
void ConvertPostfix (char *a)
{
    stack <char> s;
    char output[50], t;
  for(int i=0; a[i]!='\0'; i++)
  {
  char ch = a[i];
  Switch (ch)
  {
   case '^':
   case '-':
   case '+':
   case '/':
   case '*': s.push (ch);
             break;
  case ')': t = s.top();
            s.pop();
            cout << t;
            break;
  }
```

ACE

Page No.

Roll No.

Date

```
    if (isalpha(ch))
        cout<<ch;
    }
}
int main()
{
    char a[] = "(((a*b)+(c/d))-e)";
    ConvertPostfix(a);
    return 0;
}
```

OUTPUT

Infix expression :- (((a*b)+(c/d))-e)

Postfix expression :- ab*cd/+e-

Program-12

Write a program to perform Quick sort on input array.

```
#include<iostream>
using namespace std;
void swap(int *a, int *b)
{
   int t = *a;
   *a = *b;
   *b = t;
}
void print(int array[], int size)
{
 int i;
 for(i=0; i<size; i++)
 cout << array[i] << " ";
 cout << endl;
}
int divide(int array[], int low, int high)
{
 int pivot = array[high];
  int i = (low-1);
```

```
for (int j = low ; j < high ; j++)
{
   if (array[j] <= pivot)
   {
      i++;
      swap (&array[i] , &array[j]);
}
}
}

print (array, 7);
cout << "..... \n";
swap (&array[i+1] , &array[high]);
return (i+1);
}

void quicksort(
 if (low< high){
  int pi = divide (array, low, high);
  quicksort(array , low , pi-1);
  quicksort (array, pi+1 , high);
  }
}
 int main(){
 int data[] = {8,7,6,1,0,9,2};
 int n = size of (data) / size of (data[0]);
 quicksort (data, 0, n-1);
 cout << " Sorted array in ascending order: \n";
 print (data, n);
```

OUTPUT

1 0 6 8 7 9 2

1 0 2 8 7 9 6

0 1 2 8 7 9 6

0 1 2 6 7 9 8

Sorted array in asseeending order

0 1 2 6 7 8 9