DATA STRUCTURE LABORATORY(LCPIT-101) PRACTICAL FILE

SUBMITTED TO:

ER. PARMINDER KAUR WADHWA

ASSISTANT PROFESSOR(IT)

GNDEC,LUDHIANA

SUBMITTED BY:

NAME : SAKSHI BHUMBLA

UNIV ROLL NO : 1905391

CLASS : IT (D2)

SECTION : B2

DATA STRUCTURE LABORATORY(LCPIT-101) PRACTICAL FILE

SUBMITTED TO:

ER. PARMINDER KAUR WADHWA

ASSISTANT PROFESSOR(IT)

GNDEC,LUDHIANA

SUBMITTED BY:

NAME : SAKSHI BHUMBLA

UNIV ROLL NO : 1905391

CLASS : IT (D2)

SECTION : B2

EXPERIMENT 1 Program to insert a new element.

(i) At end as well as

(ii) At a given position of an array

CODE:

```
#include<iostream>
using namespace std;
int main(){

int size=5;
```

```
int arr[size] = {15,23,65,98,72};
cout << "Elements in Array : \n";
for(int i=0;i<size;i++){
cout << arr[i] << "\t";

}
cout << endl << endl;
int newElement;
cout << "Enter New Element : ";
cin >> newElement;
cout << endl;
cout << "Choices for Insertion : \n";


cout << "1. You want insert element at Particular Postion or \n";
cout << "2. Insert as Last Element in List.....\n"<<endl;
int choice;
cout << "Please Select Choice for Insertion : "; cin >> choice;
cout << endl;
if(choice==1)
{
int elemnum;
cout << "Enter the number (element number) Which u want to insert :";
cin >> elemnum;
int pos;
pos=elemnum-1;

cout << endl;
int        j;
for(j=size-1;j>=pos;j--)
{ arr[j+1] = arr[j];
}
arr[pos] = newElement;
size = size+1;

for(int i=0;i<size;i++)
{ cout << arr[i] << "\t";
}
}
```

```
else  if(choice==2)
{
     int lastpos=size;
   arr[lastpos] = newElement;
   size=lastpos+1;

for(int i=0;i<size;i++)
{ cout << arr[i] << "\t";

}
}
else
cout << "You Have not entered any Choice";
return 0;

}
```

OUTPUT : (1) ELEMENT INSERTED AT LAST

C:\Users\SAKSHI\Documents\labds1.exe

```
Elements in Array :
15      23      65      98      72

Enter New Element : 17

Choices for Insertion :
1. You want insert element at Particular Postion or
2. Insert as Last Element in List.....

Please Select Choice for Insertion : 1

Enter Position Where u want to insert :4

15      23      65      98      17      72
--------------------------------
Process exited after 32.3 seconds with return value 0
Press any key to continue . . .
```

(2) ELEMENT INSERTED AT PARTICULAR POSITION

C:\Users\SAKSHI\Documents\labds1.exe

```
Elements in Array :
15      23      65      98      72

Enter New Element : 17

Choices for Insertion :
1. You want insert element at Particular Postion or
2. Insert as Last Element in List.....

Please Select Choice for Insertion : 2

15      23      65      98      72      17
--------------------------------
Process exited after 14.45 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 2: PROGRAM TO DELETE AN ELEMENT

(1)FEOM A GIVEN WHOSE VALUE IS GIVEN

(2) OR WHOSE POSITION IS GIVEN

CODE:

```
#include<iostream>
using namespace std;
int main()
{
int size=5;
int arr[size] = {45,35,65,70,85};
cout << "Elements in Array : \n";
 for(int i=0;i<size;i++)
 {
cout << arr[i] << "\t";

 }
cout << endl << endl;
```

```
cout << "Choices for Deletion : \n";
cout << "1. Enter the Element which u want to delete or \n";
cout << "2. Enter the position of element.....\n"<<endl;
int choice;


cout << "Please Select Choice for Deletion : ";
cin >> choice;
cout << endl;
if(choice==1)
{
  int delElement;
  cout << "Enter the element to be deleted : ";
  cin >> delElement;
  for(int i=0;i<size;i++)
   {
     if(arr[i]==delElement)
      {
        for(int j=i;j<size;j++)
         {

              arr[j]=arr[j+1];
          }
       break;

     }
  }

for(int i=0;i<(size-1);i++)
{

cout<<arr[i]<<"\t";
}


}
else if(choice==2)
 { int position;
cout << "Enter position of element u want to delete : ";
cin >> position;
```

```
int elemnum;
elemnum=position-1;
cout <<endl;
int item = arr[elemnum];
int j = size;
for(j=elemnum;j<size;j++){
arr[j] = arr[j+1];

}
size = size-1;
for(int i=0;i<size;i++)
{ cout << arr[i] << "\t";


}
}
else

cout << "You Have not entered any Choice";

 return 0;
}
```

(1)ELEMENTS WHOSE VALUE IS GIVEN

```
C:\Users\SAKSHI\Documents\DS1.exe
Elements in Array :
45      35      65      70      85

Choices for Deletion :
1. Enter the Element which u want to delete or
2. Enter the position of element.....

Please Select Choice for deletion: 1

Enter the elements to be deleted : 35
45      65      70      85
--------------------------------
Process exited after 17.2 seconds with return value 0
Press any key to continue . . .
```

(2)ELEMENT WHOSE POSITION IS GIVEN

```
C:\Users\SAKSHI\Documents\DS1.exe
Elements in Array :
45      35      65      70      85

Choices for Deletion :
1. Enter the Element which u want to delete or
2. Enter the position of element.....

Please Select Choice for deletion: 2

Enter position of element u want to delete : 4

45      35      65      70
--------------------------------
Process exited after 18.25 seconds with return value 0
Press any key to continue . . .
```

Experiment 3: Program to find the location of a given element using Linear Search.

CODE:

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

int linearsearch(int arr[],int size, int searchElement)
{ for(int i=0;i<=size-1;i++)
  if(arr[i]==searchElement)
{

  int searched_at=(i+1);
  return searched_at;
}

return -1;
}
```

```
int main()
{
int size = 10;
int arr[size] = {2,5,6,7,12,15,25,36,49,55};
cout << "Elements in array : \n";
for(int i=0;i<10;i++)
{
cout << arr[i] << "\t";

}


int searchElement;
cout << "\nEnter the Element you want to search in the array (using linear
search) : ";
cin >> searchElement;
int location = linearsearch(arr,size,searchElement);
 if(location == -1)
cout << "Element Not Found";


else


cout << "Element Found at Location in the array (element number in array)): "
<< location;
 return 0;
}
```

(1) ELEMENT FOUND AT PARTICULAR POSITON

C:\Users\SAKSHI\Documents\DS2.exe

```
Elements in array :
2       5       6       7       12      15      25      36      49      55
Enter the Element : 7
Element Found at Location : 3
--------------------------------
Process exited after 14.06 seconds with return value 0
Press any key to continue . . .
```

(2) ELEMENT NOT FOUND IN ARRAY

C:\Users\SAKSHI\Documents\DS2.exe

```
Elements in array :
2       5       6       7       12      15      25      36      49      55
Enter the Element : 11
Element Not Found
--------------------------------
Process exited after 15.23 seconds with return value 0
Press any key to continue . . .
```

Experiment 4: Program to find the location of a given element using Binary Search.

```
#include<iostream>
using namespace std;
int binarysearch(int data[],int lb, int ub, int item)
{
    int beg=lb;
    int end=ub;
    int loc;
  int mid = (beg+end)/2;
 while((beg<=end)&& (data[mid]!=item))
  {
    if(item<data[mid])
    {
        end=mid-1;
        }
        else
        {
```

```
            beg=mid+1;
            }
            mid=(beg+end)/2;
        }
        if (data[mid]==item)
        {
            loc=mid;


        }
        else
        {
            loc=-1;
        }
        return loc;
}
    int main()
{
 int size = 10;
int data[size] = {2,5,6,7,12,15,25,36,49,55};
cout << "Elements in array : \n";
for(int i=0;i<10;i++)
{
cout << data[i] << "\t";
}
int item, location;
cout << "\nEnter the Element that you want to search using binary search: ";
```

```
cin >> item;
location = binarysearch(data,0,size-1,item);
if(location==-1)
{
    cout<<"Element is not found";
}
else
{
int elemnum=location+1;
cout << "Element found at location (the element number in the array) : " <<
elemnum;
}
return 0;
}
```

(1) ELEMENT AT PARTICULAR POSITION

C:\Users\SAKSHI\Documents\DS3.exe

```
Elements in array :
2       5       6       7       12      15      25      36      49      55
Enter the Element : 7
Element found at location : 3
--------------------------------
Process exited after 14.09 seconds with return value 0
Press any key to continue . . .
```

(2) ELEMEMT NOT FOUND

C:\Users\SAKSHI\Documents\SAKSHI_1921091.exe

```
Elements in array :
2       5       6       7       12      15      25      36      49      55
Enter the Element that you want to search using binary search: 8
Element is not found
--------------------------------
Process exited after 11.83 seconds with return value 0
Press any key to continue . . .
```

Experiment 5: Program to implement push and pop operations on a stack using linear array.

CODE:

```
# include<iostream>
#define size 5
using namespace std;
int top=-1,stack[size];
void push(int newElement)
{ if(top==size-1)
{
    cout << "\n.....\n";
cout << "Stack is Full.....cannot insert this value.....\n";
cout << "\n.....\n";
}
else{
top=top+1;
stack[top]= newElement;
}
}

void pop()
{
if(top==-1)
{
    cout << "\n.....\n";
cout << "\nStack is empty!!";
cout << "\n.....\n";
```

```
}
else{
    cout << "\n.............................\n";
    cout << "\n..........deleting the elemnt....................\n";
    cout<<"\n..\n";
    cout<<"\n..\n";
    cout<<"\n..\n";
    cout<<"\n..\n";
    cout<<"\n..\n";
 cout <<"\nDeleted element is " << stack[top];
 top=top-1;
 cout<<"\n..\n";
 cout<<"\n..\n";
 cout<<"\n..\n";
 cout<<"\n..\n";
 cout<<"\n..\n";
 cout << "\n.............................\n";
}
}

void display()
{
 int i;
 if(top==-1)
 {
cout <<"\nStack is empty!!";
```

```
}
else{
cout << "\nStack is...(Maximun capacity as 5 elements)\n";
cout << "\n.............................\n";
for(i=top;i>=0;--i)
cout << stack[i] <<"\n";
}
cout << "\n.............................\n";
cout << endl;
}
int main(){
int choice;
int newElement;
while(1){
cout << "\nSelect Your Choice : ........";
cout << "1.Push  2. Pop    3.Display   4.Exit" <<endl;


cout << "\nEnter Your Choice : ";
cin >> choice;
cout << endl;
switch(choice)
{
case 1:
{
    cout << "\n.............................\n";
```

```
cout << "Enter Element          : ";
cin >> newElement;
cout << "\n.............................\n";
push(newElement);
break;
}
case 2:
pop();
break;
case 3:
display();
break;
case 4:
exit(0);
break;
default :
cout << "Invalid Choice";
break;
}
}
//return 0;
}
```

C:\Users\SAKSHI\Documents\SASKHI1921091.exe

```
Select Your Choice : ........1.Push  2. Pop    3.Display   4.Exit

Enter Your Choice : 1


.............................
Enter Element   : 3

.............................

Select Your Choice : ........1.Push  2. Pop    3.Display   4.Exit

Enter Your Choice : 1


.............................
Enter Element   : 5

.............................

Select Your Choice : ........1.Push  2. Pop    3.Display   4.Exit

Enter Your Choice : 2


.............................

..........deleting the elemnt....................

..

..

..

..

..

Deleted element is 5
..
```

```
 ............................

elect Your Choice : ........1.Push  2. Pop    3.Display   4.Exit

nter Your Choice : 3


tack is...(Maximun capacity as 5 elements)

 ............................
3

 ............................


elect Your Choice : ........1.Push  2. Pop    3.Display   4.Exit

nter Your Choice : 4


--------------------------------
rocess exited after 75.22 seconds with return value 0
ress any key to continue . . .
```

Experiment 6: Program to convert an infix expression to a postfix expression using stacks.

CODE:

```
#include<stdio.h>
#include<stdlib.h> /* for exit() */
#include<ctype.h> /* for isdigit(char ) */
#include<string.h>
#define SIZE 100
char stack[SIZE];
int top = -1;
void push(char item){
if(top >= SIZE-1){
printf("\nStack Overflow.");
}
else{
top = top+1;
stack[top] = item;
}
}
char pop(){
char item ;
if(top <0)
{
printf("stack under flow: invalid infix expression");
getchar();
```

```
exit(1);
}
else{
item = stack[top];
top = top-1;
return(item);
}
}
int is_operator(char symbol){
if(symbol == '^' || symbol == '*' || symbol == '/' || symbol == '+' || symbol
=='-'){
return 1;
}
else{
return 0;
}
}
int precedence(char symbol){
if(symbol == '^'){
return(3);
}
else if(symbol == '*' || symbol == '/'){
return(2);
}
else if(symbol == '+' || symbol == '-'){
```

```
return(1);
}
else{
return(0);
}
}
void InfixToPostfix(char infix_exp[], char postfix_exp[]){
int i, j;
char item;
char x;
push('(');
strcat(infix_exp,")");
i=0;
j=0;
item=infix_exp[i];
while(item != '\0'){
if(item == '('){
push(item);
}
else if( isdigit(item) || isalpha(item)){
postfix_exp[j] = item;
j++;
}
else if(is_operator(item) == 1){
x=pop();
```

```
while(is_operator(x) == 1 && precedence(x)>= precedence(item)){
postfix_exp[j] = x;
j++;
x = pop();
}
push(x);
push(item);
}
else if(item == ')'){
x = pop();
while(x != '('){
postfix_exp[j] = x;
j++;
x = pop();
}
}
else{
printf("\nInvalid infix Expression.\n");
getchar();
exit(1);
}
i++;
item = infix_exp[i];
}
if(top>0){
```

```
printf("\nInvalid infix Expression.\n");
getchar();
exit(1);
}
postfix_exp[j] = '\0';
}
int main(){
char infix[SIZE], postfix[SIZE];
printf("\nEnter Infix expression : ");
gets(infix);
InfixToPostfix(infix,postfix);
printf("Postfix Expression: ");
puts(postfix);
return 0;
}
```

C:\Users\SAKSHI\Documents\Sakshi1921091.exe

```
Enter Infix expression : A+(B*C-(D/E^F)*G)*H
Postfix Expression: ABC*DEF^/G*-H*+

--------------------------------
Process exited after 160.3 seconds with return value 0
Press any key to continue . . .
```

Experiment 7: Program to evaluate a postfix expression using stacks.

CODE:

```
// CPP program to evaluate value of a postfix
 // expression having multiple digit operands
 #include <bits/stdc++.h>
 using namespace std;

 // Stack type
 class Stack
 {
     public:
     int top;
     unsigned capacity;
     int* array;
 };

 // Stack Operations
 Stack* createStack( unsigned capacity )
 {
     Stack* stack = new Stack();

     if (!stack) return NULL;

     stack->top = -1;
     stack->capacity = capacity;
     stack->array = new int[(stack->capacity * sizeof(int))];

     if (!stack->array) return NULL;
```

```
    return stack;
}

int isEmpty(Stack* stack)
{
    return stack->top == -1 ;
}

int peek(Stack* stack)
{
    return stack->array[stack->top];
}

int pop(Stack* stack)
{
    if (!isEmpty(stack))
        return stack->array[stack->top--] ;
    return '$';
}

void push(Stack* stack,int op)
{
    stack->array[++stack->top] = op;
}
```

```
// The main function that returns value
// of a given postfix expression
int evaluatePostfix(char* exp)
{
    // Create a stack of capacity equal to expression size
    Stack* stack = createStack(strlen(exp));
    int i;

    // See if stack was created successfully
    if (!stack) return -1;

    // Scan all characters one by one
    for (i = 0; exp[i]; ++i)
    {
        //if the character is blank space then continue
        if(exp[i] == ' ')continue;

        // If the scanned character is an
        // operand (number here),extract the full number
        // Push it to the stack.
        else if (isdigit(exp[i]))
        {
            int num=0;

            //extract full number
```

```
        while(isdigit(exp[i]))
        {
        num = num * 10 + (int)(exp[i] - '0');
                i++;
        }
        i--;

        //push the element in the stack
        push(stack,num);
}

// If the scanned character is an operator, pop two
// elements from stack apply the operator
else
{
        int val1 = pop(stack);
        int val2 = pop(stack);

        switch (exp[i])
        {
        case '+': push(stack, val2 + val1); break;
        case '-': push(stack, val2 - val1); break;
        case '*': push(stack, val2 * val1); break;
        case '/': push(stack, val2/val1); break;

        }
```

```
            }
        }
        return pop(stack);
    }

    // Driver code
    int main()
    {
        char exp[] = "60 6 / 5 2 * 5 - +";
        cout<< "The result of the evaluated postfix expression is =  \n";
        cout<<".............\n";
      cout<<".............\n";
        cout << evaluatePostfix(exp);
        cout<<"\n.............\n";
        cout<<".............\n";
        return 0;
    }
```

C:\Users\SAKSHI\Documents\SAKSHI_1921091.exe

```
The result of the evaluated postfix expression is =
.............
.............
15
.............
.............

--------------------------------
Process exited after 6.537 seconds with return value 0
Press any key to continue . . .
```

Experiment 8: Implement recursive function for Tower of Hanoi problem.

CODE:

```
#include<iostream>
using namespace std;
void TOH(int n,char Beg, char Aux,char End){
if(n==1){
cout<<"Move Disk "<<n<<" from
"<<Beg<<" to "<<End<<endl;
return;
}
else{
TOH(n-1,Beg,End,Aux);
cout<<"Move Disk "<<n<<" from
"<<Beg<<" to "<<End<<endl;
TOH(n-1,Aux,Beg,End);
}
}
int main(){
int n;
cout<<"Enter no. of disks:";
cin>>n;
TOH(n,'A','B','C');
return 0;
```

}

C:\Users\SAKSHI\Documents\DS7.exe

```
Enter no. of disks:4
Move Disk 1 fromA to B
Move Disk 2 fromA to C
Move Disk 1 fromB to C
Move Disk 3 fromA to B
Move Disk 1 fromC to A
Move Disk 2 fromC to B
Move Disk 1 fromA to B
Move Disk 4 fromA to C
Move Disk 1 fromB to C
Move Disk 2 fromB to A
Move Disk 1 fromC to A
Move Disk 3 fromB to C
Move Disk 1 fromA to B
Move Disk 2 fromA to C
Move Disk 1 fromB to C

--------------------------------
Process exited after 3.47 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 9

Insertion, deletion in a queue.

```
#include<iostream>

using namespace std;

int queue[3],size=3,front=-1,rear=-1;

void enqueue()

{

int new_element;


if(rear+1==size)

{
```

```
cout << ".......SORRY!..........Queue Is Full ...........\n";


}
else if(front == -1)
{
front = 0;
rear = 0;
cout << ".............Enter the Element : ..........";
cin >> new_element;
queue[rear]=new_element;
cout<<"...........The program has inserted the FIRST element in this queue!   \n";
cout<<".....The FRONT and REAR are same now................";
}
else
{
        rear = rear + 1;
        cout << ".............Enter the Element : ..........";
cin >> new_element;
queue[rear]=new_element;
cout<<"...........The program has inserted the element at the rear of this
queue!................";


}
}
void dequeue()
{
```

```
int deleted_element;
if(front==-1)
{
cout << ".....Oh!.......Queue UnderFlow has occurred!..... \n";
}
else if(front==rear)
{

deleted_element = queue[front];
cout<<".......The program has deleted the ONLY element from front of this queue!  \n";
cout<<" .........and queue is EMPTY NOW..... \n";

front = -1;
rear = -1;

}
else
{
deleted_element = queue[front];
cout<<".....The program has deleted the element from front of this queue!.....";
front++;
}
}
void dispaly()
{
```

```
if(front == -1)
{
cout << ".....OOps!.....Queue is Empty..... \n";
}
else
{
cout << ".....Displaying the Queue Elements are : .....\n" << endl;
for(int i=front;i<=rear;i++)
{
cout << queue[i] << " ";
cout << endl;
}
}
}

int main()
{
int operation_choice;
while(1)
{
cout << ".....Select Your Choice : ....." << endl;
cout << "1. Enqueue Operation" << endl;
cout << "2. Dequeue Operation" << endl;
cout << "3. Display" << endl;
cout << "4. Exit" << endl;
cout << "Enter Your Choice : ";
```

```
cin >> operation_choice;
switch (operation_choice)
{
case 1: enqueue();
break;
case 2: dequeue();
break;
case 3: dispaly();
break;
case 4: exit(0);
break;
default: printf("Invalid Choice!!\n");
break;
}
}
return 0;
}
```

C:\Users\SAKSHI\Documents\Sakshi1921091.exe

```
.................Select Your Choice : ...............
1. Enqueue Operation
2. Dequeue Operation
3. Display
4. Exit
Enter Your Choice : 1
.............Enter the Element : ..........12
...........The program has inserted the FIRST element in this queue!
......The FRONT and REAR are same now..................................Select Your Choice : ...............
1. Enqueue Operation
2. Dequeue Operation
3. Display
4. Exit
Enter Your Choice : 2
.......The program has deleted the ONLY element from front of this queue!
 .........and queue is EMPTY NOW.....
.................Select Your Choice : ...............
1. Enqueue Operation
2. Dequeue Operation
3. Display
4. Exit
Enter Your Choice : 1
.............Enter the Element : ..........1
...........The program has inserted the FIRST element in this queue!
......The FRONT and REAR are same now..................................Select Your Choice : ...............
1. Enqueue Operation
2. Dequeue Operation
3. Display
4. Exit
Enter Your Choice : 3
..........Displaying the Queue Elements are : ..............

1
.................Select Your Choice : ...............
1. Enqueue Operation
2. Dequeue Operation
3. Display
4. Exit
Enter Your Choice : 4

--------------------------------
Process exited after 60.3 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 10

Implement a linked list

// Linked list operations in C++

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

```
#include <iostream>

using namespace std;

// Create a node
struct Node {
  int item;
  struct Node* next;
};

void insertAtBeginning(struct Node** ref, int data) {

  // Allocate memory to a node
  struct Node* new_node = (struct Node*)malloc(sizeof(struct Node));

  // insert the item
  new_node->item = data;
  new_node->next = (*ref);

  // Move head to new node
  (*ref) = new_node;
}

// Insert a node after a node
void insertAfter(struct Node* prev_node, int data) {
  if (prev_node == NULL) {
```

```
    cout << "the given previous node cannot be NULL";
    return;
  }

  struct Node* new_node = (struct Node*)malloc(sizeof(struct Node));
  new_node->item = data;
  new_node->next = prev_node->next;
  prev_node->next = new_node;
}

void insertAtEnd(struct Node** ref, int data) {
  struct Node* new_node = (struct Node*)malloc(sizeof(struct Node));
  struct Node* last = *ref;

  new_node->item = data;
  new_node->next = NULL;

  if (*ref == NULL) {
    *ref = new_node;
    return;
  }

  while (last->next != NULL)
    last = last->next;

  last->next = new_node;
```

```
  return;
}

void deleteNode(struct Node** ref, int key) {
  struct Node *temp = *ref, *prev;

  if (temp != NULL && temp->item == key) {
    *ref = temp->next;
    free(temp);
    return;
  }
  // Find the key to be deleted
  while (temp != NULL && temp->item != key) {
    prev = temp;
    temp = temp->next;
  }

  // If the key is not present
  if (temp == NULL) return;

  // Remove the node
  prev->next = temp->next;

  free(temp);
}
```

```
// Print the linked list
void printList(struct Node* node) {
  while (node != NULL) {
    cout << node->item << " ";
    node = node->next;
  }
}

// Driver program
int main() {
  struct Node* head = NULL;

  insertAtEnd(&head, 1);
  insertAtBeginning(&head, 2);
  insertAtBeginning(&head, 3);
  insertAtEnd(&head, 4);
  insertAfter(head->next, 5);

  cout << "Linked list: ";
  printList(head);

  cout << "\nAfter deleting an element: ";
  deleteNode(&head, 3);
  printList(head);
}
```

C:\Users\SAKSHI\Documents\Sakshi1921091.exe

```
Linked list: 3 2 5 1 4
After deleting an element: 2 5 1 4
--------------------------------
Process exited after 5.398 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 11

IMPLEMENT STACK USING LINKED LIST

```
#include <iostream>
using namespace std;
struct Node {
  int data;
  struct Node *next;
};
struct Node* top = NULL;
void push(int val) {
  struct Node* newnode = (struct Node*) malloc(sizeof(struct Node));
  newnode->data = val;
  newnode->next = top;
  top = newnode;
}
void pop() {
  if(top==NULL)
  cout<<"Stack Underflow"<<endl;
  else {
    cout<<"The popped element is "<< top->data <<endl;
    top = top->next;
  }
}
void display() {
  struct Node* ptr;
  if(top==NULL)
  cout<<"stack is empty";
  else {
    ptr = top;
    cout<<"Stack elements are: ";
    while (ptr != NULL) {
```

```
        cout<< ptr->data <<" ";
        ptr = ptr->next;
      }
    }
    cout<<endl;
}
int main() {
    int ch, val;
    cout<<"1) Push in stack"<<endl;
    cout<<"2) Pop from stack"<<endl;
    cout<<"3) Display stack"<<endl;
    cout<<"4) Exit"<<endl;
    do {
      cout<<"Enter choice: "<<endl;
      cin>>ch;
      switch(ch) {
        case 1: {
          cout<<"Enter value to be pushed:"<<endl;
          cin>>val;
          push(val);
          break;
        }
        case 2: {
          pop();
          break;
        }
        case 3: {
          display();
          break;
        }
        case 4: {
          cout<<"Exit"<<endl;
          break;
        }
        default: {
          cout<<"Invalid Choice"<<endl;
        }
      }
    }while(ch!=4);
    return 0;
```

}

"C:\Users\SAKSHI\Documents\Sakshi 1905391.exe"

```
1) Push in stack
2) Pop from stack
3) Display stack
4) Exit
Enter choice:
1
Enter value to be pushed:
23
Enter choice:
1
Enter value to be pushed:
11
Enter choice:
2
The popped element is 11
Enter choice:
3
Stack elements are: 23
Enter choice:
4
Exit

Process returned 0 (0x0)   execution time : 33.657 s
Press any key to continue.
```

EXPERIMENT 12

IMPLEMENT QUEUE USING LINKED LIST

```
#include <iostream>
using namespace std;
struct node {
  int data;
  struct node *next;
};
struct node* front = NULL;
struct node* rear = NULL;
struct node* temp;
```

```
void Insert() {
  int val;
  cout<<"Insert the element in queue : "<<endl;
  cin>>val;
  if (rear == NULL) {
    rear = (struct node *)malloc(sizeof(struct node));
    rear->next = NULL;
    rear->data = val;
    front = rear;
  } else {
    temp=(struct node *)malloc(sizeof(struct node));
    rear->next = temp;
    temp->data = val;
    temp->next = NULL;
    rear = temp;
  }
}
void Delete() {
  temp = front;
  if (front == NULL) {
    cout<<"Underflow"<<endl;
    return;
  }
  else
  if (temp->next != NULL) {
    temp = temp->next;
    cout<<"Element deleted from queue is : "<<front->data<<endl;
    free(front);
    front = temp;
  } else {
    cout<<"Element deleted from queue is : "<<front->data<<endl;
    free(front);
    front = NULL;
    rear = NULL;
  }
}
void Display() {
  temp = front;
  if ((front == NULL) && (rear == NULL)) {
    cout<<"Queue is empty"<<endl;
```

```
    return;
  }
  cout<<"Queue elements are: ";
  while (temp != NULL) {
    cout<<temp->data<<" ";
    temp = temp->next;
  }
  cout<<endl;
}
int main() {
  int ch;
  cout<<"1) Insert element to queue"<<endl;
  cout<<"2) Delete element from queue"<<endl;
  cout<<"3) Display all the elements of queue"<<endl;
  cout<<"4) Exit"<<endl;
  do {
    cout<<"Enter your choice : "<<endl;
    cin>>ch;
    switch (ch) {
      case 1: Insert();
      break;
      case 2: Delete();
      break;
      case 3: Display();
      break;
      case 4: cout<<"Exit"<<endl;
      break;
      default: cout<<"Invalid choice"<<endl;
    }
  } while(ch!=4);
  return 0;
}
```

```
"C:\Users\SAKSHI\Documents\Sakshi 1905391.exe"
1) Insert element to queue
2) Delete element from queue
3) Display all the elements of queue
4) Exit
Enter your choice :
1
Insert the element in queue :
11
Enter your choice :
2
Element deleted from queue is : 11
Enter your choice :
1
Insert the element in queue :
23
Enter your choice :
3
Queue elements are: 23
Enter your choice :
4
Exit

Process returned 0 (0x0)   execution time : 45.615 s
Press any key to continue.
```

EXPERIMENT 13

BUBBLE SORT

```
#include<iostream>

using namespace std;

void swapping(int &a, int &b) {     //swap the content of a and b

  int temp;

  temp = a;

  a = b;

  b = temp;

}

void display(int *array, int size) {

  for(int i = 0; i<size; i++)

    cout << array[i] << " ";
```

```
  cout << endl;
}
void bubbleSort(int *array, int size) {
  for(int i = 0; i<size; i++) {
    int swaps = 0;        //flag to detect any swap is there or not
    for(int j = 0; j<size-i-1; j++) {
      if(array[j] > array[j+1]) {      //when the current item is bigger than next
        swapping(array[j], array[j+1]);
        swaps = 1;    //set swap flag
      }
    }
    if(!swaps)
      break;      // No swap in this pass, so array is sorted
  }
}
int main() {
  int n;
  cout << "Enter the number of elements: ";
  cin >> n;
  int arr[n];     //create an array with given number of elements
  cout << "Enter elements:" << endl;
  for(int i = 0; i<n; i++) {
    cin >> arr[i];
  }
  cout << "Array before Sorting: ";
  display(arr, n);
```

```
  bubbleSort(arr, n);
  cout << "Array after Sorting: ";
  display(arr, n);
}
```

C:\Users\SAKSHI\Documents\Sakshi1921091.exe

```
Enter the number of elements: 5
Enter elements:
12
15
67
34
11
Array before Sorting: 12 15 67 34 11
Array after Sorting: 11 12 15 34 67

--------------------------------
Process exited after 24.69 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 14

SELECTION SORT

```
#include<iostream>
using namespace std;
void swapping(int &a, int &b) {        //swap the content of a and b
  int temp;
  temp = a;
  a = b;
  b = temp;
}
void display(int *array, int size) {
  for(int i = 0; i<size; i++)
    cout << array[i] << " ";
```

```
  cout << endl;
}
void selectionSort(int *array, int size) {
  int i, j, imin;
  for(i = 0; i<size-1; i++) {
    imin = i;   //get index of minimum data
    for(j = i+1; j<size; j++)
      if(array[j] < array[imin])
        imin = j;
      //placing in correct position
      swap(array[i], array[imin]);
  }
}
int main() {
  int n;
  cout << "Enter the number of elements: ";
  cin >> n;
  int arr[n];          //create an array with given number of elements
  cout << "Enter elements:" << endl;
  for(int i = 0; i<n; i++) {
    cin >> arr[i];
  }
  cout << "Array before Sorting: ";
  display(arr, n);
  selectionSort(arr, n);
  cout << "Array after Sorting: ";
```

```
  display(arr, n);
}
```

C:\Users\SAKSHI\Documents\Sakshi1921091.exe

```
Enter the number of elements: 5
Enter elements:
87
90
56
34
11
Array before Sorting: 87 90 56 34 11
Array after Sorting: 11 34 56 87 90

--------------------------------
Process exited after 17.79 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 15

INSERTION SORT

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

void display(int *array, int size) {
  for(int i = 0; i<size; i++)
    cout << array[i] << " ";
  cout << endl;
}
```

```
void insertionSort(int *array, int size) {
  int key, j;

  for(int i = 1; i<size; i++) {
    key = array[i];//take value
    j = i;

    while(j > 0 && array[j-1]>key) {
      array[j] = array[j-1];
      j--;
    }

    array[j] = key;//insert in right place
  }
}

int main() {
  int n;
  cout << "Enter the number of elements: ";
  cin >> n;
  int arr[n]; //create an array with given number of elements
  cout << "Enter elements:" << endl;

  for(int i = 0; i<n; i++) {
    cin >> arr[i];
  }
```

```
  cout << "Array before Sorting: ";
  display(arr, n);
  insertionSort(arr, n);
  cout << "Array after Sorting: ";
  display(arr, n);
}
```

C:\Users\SAKSHI\Documents\Sakshi 1905391.exe

```
Enter the number of elements: 5
Enter elements:
11
27
4
54
76
Array before Sorting: 11 27 4 54 76
Array after Sorting: 4 11 27 54 76

--------------------------------
Process exited after 18.82 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 16

QUICKSORT

```
#include <iostream>
using namespace std;
// Swap two elements - Utility function
void swap(int* a, int* b)
{
   int t = *a;
   *a = *b;
   *b = t;
}
```

```
// partition the array using last element as pivot
int partition (int arr[], int low, int high)
{
   int pivot = arr[high];    // pivot
   int i = (low - 1);

   for (int j = low; j <= high- 1; j++)
   {
      //if current element is smaller than pivot, increment the low element
      //swap elements at i and j
      if (arr[j] <= pivot)
      {
         i++;    // increment index of smaller element
         swap(&arr[i], &arr[j]);
      }
   }
   swap(&arr[i + 1], &arr[high]);
   return (i + 1);
}

//quicksort algorithm
void quickSort(int arr[], int low, int high)
{
   if (low < high)
   {
      //partition the array
```

```
        int pivot = partition(arr, low, high);

        //sort the sub arrays independently
        quickSort(arr, low, pivot - 1);
        quickSort(arr, pivot + 1, high);
    }
}

void displayArray(int arr[], int size)
{
    int i;
    for (i=0; i < size; i++)
        cout<<arr[i]<<"\t";

}

int main()
{
    int arr[] = {12,23,3,43,51,35,19,45};
    int n = sizeof(arr)/sizeof(arr[0]);
    cout<<"Input array"<<endl;
    displayArray(arr,n);
    cout<<endl;
    quickSort(arr, 0, n-1);
    cout<<"Array sorted with quick sort"<<endl;
    displayArray(arr,n);
```

```
    return 0;
}
```

C:\Users\SAKSHI\Documents\Sakshi 1905391.exe

```
Input array
12      23      3       43      51      35      19      45
Array sorted with quick sort
3       12      19      23      35      43      45      51
--------------------------------
Process exited after 2.639 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 17

PREORDER, INORDER, POSTORDER TRAVERSALS

```
// C program for different tree traversals
#include <iostream>
using namespace std;

/* A binary tree node has data, pointer to left child
and a pointer to right child */
struct Node
{
    int data;
    struct Node* left, *right;
    Node(int data)
    {
        this->data = data;
        left = right = NULL;
    }
};
```

```
/* Given a binary tree, print its nodes according to the
"bottom-up" postorder traversal. */
void printPostorder(struct Node* node)
{
   if (node == NULL)
      return;

   // first recur on left subtree
   printPostorder(node->left);

   // then recur on right subtree
   printPostorder(node->right);

   // now deal with the node
   cout << node->data << " ";
}

/* Given a binary tree, print its nodes in inorder*/
void printInorder(struct Node* node)
{
   if (node == NULL)
      return;

   /* first recur on left child */
   printInorder(node->left);
```

```
    /* then print the data of node */
    cout << node->data << " ";

    /* now recur on right child */
    printInorder(node->right);
}

/* Given a binary tree, print its nodes in preorder*/
void printPreorder(struct Node* node)
{
    if (node == NULL)
        return;

    /* first print data of node */
    cout << node->data << " ";

    /* then recur on left sutree */
    printPreorder(node->left);

    /* now recur on right subtree */
    printPreorder(node->right);
}

/* Driver program to test above functions*/
int main()
```

```
{
    struct Node *root = new Node(1);
    root->left          = new Node(2);
    root->right        = new Node(3);
    root->left->left     = new Node(4);
    root->left->right = new Node(5);

    cout << "\n Preorder traversal of binary tree is \n";
    printPreorder(root);

    cout << "\n Inorder traversal of binary tree is \n";
    printInorder(root);

    cout << "\n Postorder traversal of binary tree is  \n";
    printPostorder(root);

    return 0;
}
```

C:\Users\SAKSHI\Documents\Sakshi 1905391.exe

```
 Preorder traversal of binary tree is
1 2 4 5 3
 Inorder traversal of binary tree is
4 2 5 1 3
 Postorder traversal of binary tree is
4 5 2 3 1
--------------------------------
Process exited after 2.626 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 18

BREADTH FIRST SEARCH (BFS)

```
#include<iostream>
#include<queue>
#define NODE 6
using namespace std;
typedef struct node{
  int val;
  int state; //status
}node;
int graph[NODE][NODE] = {
  {0, 1, 1, 1, 0, 0},
  {1, 0, 0, 1, 1, 0},
  {1, 0, 0, 1, 0, 1},
  {1, 1, 1, 0, 1, 1},
  {0, 1, 0, 1, 0, 1},
  {0, 0, 1, 1, 1, 0}
};
void bfs(node *vert, node s){
  node u;
  int i, j;
  queue<node> que;
  for(i = 0; i<NODE; i++){
```

```
    vert[i].state = 0; //not visited
  }
  vert[s.val].state = 1;//visited
  que.push(s); //insert starting node
  while(!que.empty()){
    u = que.front(); //delete from queue and print
    que.pop();
    cout << char(u.val+'A') << " ";
    for(i = 0; i<NODE; i++){
      if(graph[i][u.val]){
        //when the node is non-visited
        if(vert[i].state == 0){
          vert[i].state = 1;
          que.push(vert[i]);
        }
      }
    }
    u.state = 2;//completed for node u
  }
}
int main(){
  node vertices[NODE];
  node start;
  char s;
  for(int i = 0; i<NODE; i++){
    vertices[i].val = i;
```

```
  }
  s = 'B';//starting vertex B
  start.val = s-'A';
  cout << "BFS Traversal: ";
  bfs(vertices, start);
  cout << endl;
}
```

C:\Users\SAKSHI\Documents\Sakshi 1905391.exe

```
BFS Traversal: B A D E C F

--------------------------------
Process exited after 2.751 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 19

DEPTH FIRST SEARCH (DFS)

```
#include<iostream>
#include<stack>
using namespace std;
#define NODE 6

typedef struct node {
  int val;
  int state; //status
}node;

int graph[NODE][NODE] = {
  {0, 1, 1, 1, 0, 0},
```

```
  {1, 0, 0, 1, 1, 0},
  {1, 0, 0, 1, 0, 1},
  {1, 1, 1, 0, 1, 1},
  {0, 1, 0, 1, 0, 1},
  {0, 0, 1, 1, 1, 0}
};

void dfs(node *vertex, node start) {
  node u;
  stack<node> myStack;

  for(int i = 0; i<NODE; i++) {
    vertex[i].state = 0;   //not visited
  }

  myStack.push(start);
  while(!myStack.empty()) {
    //pop and print node
    u = myStack.top();
    myStack.pop();
    cout << char(u.val+'A') << " ";

    if(u.state != 1) {
      //update vertex status to visited
      u.state = 1;
      vertex[u.val].state = 1;
```

```
        for(int i = 0; i<NODE; i++) {
          if(graph[i][u.val]) {
            if(vertex[i].state == 0) {
              myStack.push(vertex[i]);
              vertex[i].state = 1;
            }
          }
        }
      }
    }
}

int main() {
  node vertices[NODE];
  node start;
  char s;

  for(int i = 0; i<NODE; i++) {
    vertices[i].val = i;
  }

  s = 'C';   //starting vertex C
  start.val = s-'A';
  cout << "DFS Traversal: ";
  dfs(vertices, start);
```

```
  cout << endl;
}
```

C:\Users\SAKSHI\Documents\Sakshi 1905391.exe

```
DFS Traversal: C F E B D A

--------------------------------
Process exited after 2.7 seconds with return value 0
Press any key to continue . . .
```