

---

- **Assignment Name: Demonstration of Array**

```
#include<iostream.h>

#include<conio.h>

#include<process.h>

class demo

{

int a[10],i,j,n,item,k;

public:

void get();

void insert();

void del();

void dis();

};

void demo::get()

{

cout<<"\nEnter n";

cin>>n;

cout<<"\nEnter Array Element:";

for(i=0;i<n;i++)

cin>>a[i];

}
```

```
void demo::insert()
{
    cout<<"\nEnter Position:";
    cin>>k;
    cout<<"\nEnter Item:";
    cin>>item;

    j=n;
    while(j>=k)
    {
        a[j+1]=a[j];

        j--;
    }

    a[k]=item;
    n++;
}

void demo::del()
{
    cout<<"\nEnter Position:";
    cin>>k;

    j=k;
    while(j<=n-1)
    {
```

```

a[j]=a[j+1];

j++;

}

n--;

}

void demo::dis()

{

cout<<"\n Elements are\n";

for(i=0;i<n;i++)

cout<<a[i]<<"\t";

}

void main()

{

clrscr();

2

demo d;

int ch;

d.get();

cout<<"\n1. Insert 2.Del 3.Dis 4. Exit\n";

while(ch!=4)

{

cout<<"\n Enter choice";

```

```

cin>>ch;
switch(ch)
{
case 1: d.insert(); break;
case 2: d.del(); break;
case 3: d.dis(); break;
case 4: exit(0);
}
}
getch();
}

```

- **Demonstration of Matrix**

```

// prac2
#include<iostream.h>
#include<conio.h>
class matrix
{
int a[5][5],b[5][5],c[5][5],d[5][5],e[5][5],f[5][5];
int p,q,i,j,k,n,m;
public:
void get();
void add();

```

```
void sub();

void trans();

};

void matrix::get()
{
    cout<<"\nEnter Number of Row & Column :\t";
    cin>>n>>m;

    cout<<"\nEnter the first Matrix:\n";
    for(i=0;i<n;i++)
    {
        for(j=0;j<m;j++)
            cin>>a[i][j];
    }

    cout<<"\nEnter Number of Row & Column :\t";
    cin>>p>>q;

    cout<<"\nEnter the first Matrix:\n";
    for(i=0;i<p;i++)
    {
        for(j=0;j<q;j++)
            cin>>b[i][j];
    }
}
```

```
void matrix::add()
{
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
c[i][j]=a[i][j]+b[i][j];
}
}
cout<<"\nThe addition of two matrix is :\n";
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
cout<<c[i][j]<<"\t";
cout<<"\n";
}
}
}
```

```
void matrix::sub()
{
for(i=0;i<n;i++)
{
```

```
for(j=0;j<m;j++)
{
d[i][j]=a[i][j]-b[i][j];
}
}

cout<<"\nThe Substraction of two matrix is :\n";
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
    cout<<d[i][j]<<"\t";

    cout<<"\n";
}
}

void matrix::trans()
{
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
e[i][j]=a[j][i];
}
}
}
```

```

cout<<"\nThe Transpose of first matrix is :\n";
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
cout<<e[i][j]<<"\t";
cout<<"\n";
}
}

```

```

void main()
{
clrscr();
matrix m;
m.get();
m.add();
m.sub();
m.trans();
getch();
}

```

- **Implement Stack for Integer**

```

#include<iostream.h>

#include<conio.h>

```



```
#include<process.h>

class stack

{
int s[10],n,top,ele,i;

public:

stack()

{
top=-1;
}

void push();

void dis();

int pop();

int peep();

void change();

};

void stack::push()

{
if(top>=2)

cout<<"\nStack is overflow:";

else

{

cout<<"\nEnter element:";
```

```
cin>>ele;

top++;

s[top]=ele;

}

}

void stack::dis()

{

cout<<"\nElements in stack are:\n";

for(i=top;i>=0;i--)

cout<<s[i]<<"\t";

}

int stack::pop()

{

if(top== -1)

{

cout<<"\nUnderflow";

return 0;

}

else

return (s[top--]);

}

int stack::peek()
```

```

{
cout<<"\nEnter position:";
cin>>i;
if((top-i+1)<0)
{
cout<<"\nUnderflow";
return 0;
7
}
else
return (s[top-i+1]);
}
void stack::change()
{
cout<<"\nEnter position ";
cin>>i;
if((top-i+1)<0)
{
cout<<"\nUnderflow";
}
else
{

```

```

int n;

cout<<"\nEnter element:";

cin>>n;

s[top-i+1]=n;

}

}

void main()

{

clrscr();

stack s;

int ch;

cout<<"\n1. Push 2.Display 3.Pop 4.Peep 5.Change 6.Exit\n";

while(ch!=6)

{

cout<<"\nEnter ch :";

cin>>ch;

switch(ch)

{

case 1: s.push(); break;

case 2: s.dis(); break;

case 3: int n=s.pop();

if(n>0)

```

```

cout<<"\nPop ele is "<<n;
break;
case 4: int m=s.peep();
if(m>0)
cout<<"\nPeep ele is "<<m;
break;
case 5: s.change(); break;
case 6: exit(0);
}
}
getch();
}

```

- **Implement linear queue for integer**

```

#include<iostream.h>
#include<conio.h>
#include<process.h>
class queue
{
int f,r,q[10],n,i;
public:
queue()
{

```

```
f=r=0;

}

void insert();

void del();

void dis();

};

void queue::insert()

{

if(r==3)

cout<<"\nOverflow";

else

{

cout<<"\nEnter n";

cin>>n;

if(f==0)

    f=1;

    r++;

    q[r]=n;

}

}

void queue::del()

{
```

```
if(f==0)
{
cout<<"\nUnderflow";
return;
}
else
{
int n;
n=q[f];
if(f==r)
f=r=0;
else
f++;
cout<<"\nDeleted element is "<<n;
}
}
void queue::dis()
{
if(f==0)
cout<<"\nUnderflow";
else
{
```

```

cout<<"\nElements in queue are:";

10

for(i=f;i<=r;i++)

    cout<<q[i]<<"\t";

}

}

void main()

{

clrscr();

queue q;

int ch;

cout<<"\n 1.insert 2.display 3.delete 4. exit \n";

while(ch!=4)

{

cout<<"\nEnter ch:";

cin>>ch;

switch(ch)

{

case 1: q.insert(); break;

case 2: q.dis(); break;

case 3: q.del(); break;

case 4:exit(0);

```



```
}
```

```
}
```

```
getch();
```

```
}
```

- **Perform Insert, Display, delete, search, sum operation on LL**

```
// Link List insert at begening
```

```
#include<iostream.h>
```

```
#include<conio.h>
```

```
#include<process.h>
```

```
class linklist
```

```
{
```

```
int ele,data;
```

```
linklist *start,*link,*move,*next;
```

```
public:
```

```
linklist()
```

```
{
```

```
start=NULL;
```

```
}
```

```
void insertbeg();
```

```
void search();
```

```
void del();
```

```
void show();
```

```
void count();

};

void linklist::insertbeg()

{

cout<<"enter elements"<<endl;

cin>>ele;

linklist *node;

node=new linklist();

node->data=ele;

node->link=NULL;

if(start==NULL)

{

start=node;

}

else

{

node->link=start;

start=node;

}

}

void linklist::show()

{
```

```
if(start==NULL)

{

cout<<"node is empty"<<endl;

}

else

{

move=start;

while(move!=NULL)

{

cout<<move->data;

move=move->link;

}

}

}

void linklist::del()

{

if(start==NULL)

{

cout<<"Linklist is empty"<<endl;;

}

else

{
```

```
        linklist *temp;

        temp=start;

        ele=temp->data;

        start=temp->link;

        delete temp;

    }

}

void linklist::search()

{

    int s;

    cout<<"enter element to search"<<endl;

    cin>>s;

    if(start==NULL)

    {

        cout<<"node is empty"<<endl;

    }

    else

    {

        move=start;

        while(move!=NULL)

        {

            if(move->data==s)
```

```
{  
    cout<<"data found"<<endl;  
    move=move->next;  
}  
else  
{  
    cout<<"data not found"<<endl;  
    move=move->next;  
}  
}  
}  
}  
  
void linklist::count()  
{  
    int c;  
    if(start==NULL)  
    {  
        cout<<"Empty";  
    }  
    else  
    {  
        linklist *temp;
```

```

        temp=start;

        c=0;

        while(temp!=NULL)
        {
                c=c+temp->data;

                temp=temp->link;

        }

        cout<<"Sum"<<c;

}

void main()

{

clrscr();

linklist l;

int ch;

cout<<"1.insertbeg 2.show 3.Delete 4.search 5.Count 6.exit"<<endl;

while(ch!=6)

{

cout<<"\nenter your choise"<<endl;

cin>>ch;

switch(ch)

{

```

```

case 1:l.insertbeg());break;
case 2:l.show());break;
case 3:l.del());break;
case 4:l.search());break;
case 5:l.count());break;
case 6:exit(0);
default:cout<<"you entered wrong choise"<<endl;
}
}
getch();
}

```

- **Perform Deletion in LL according to position & information**

```

#include<iostream.h>
#include<conio.h>
#include<process.h>
class node
{
int info,item;
node *link;
public:
void insert();
void dis();

```

```
void del_info();  
void del_pos();  
  
};  
  
node *move,*start,*temp;  
void node::insert()  
{  
    cout<<"\nEnter the item:";  
    cin>>item;  
    node *node1=new node;  
    node1->link=NULL;  
    node1->info=item;  
    if(start==NULL)  
        start=node1;  
    else  
    {  
        move=start;  
        while(move->link!=NULL)  
            move=move->link;  
        move->link=node1;  
    }  
}  
  
void node::dis()
```



```

{
node *x;
x=start;
while(x!=NULL)
{
cout<<"\t"<<x->info;
x=x->link;
}
}

void node::del_pos()
{
int pos,f=0,c=0;
node *p;
cout<<"\nEnter Position:";
cin>>pos;
temp=start;
if(start==NULL)
cout<<"\nLL is empty\n";
if(pos==1)
{
start=start->link;

```

```
f=1;

}

while(temp!=NULL)

{

c++;

p=temp;

temp=temp->link;

if(c==pos-1)

{

f=1;

p->link=temp->link;

}

}

if(f==0)

cout<<"\n node is not found";

}

void node::del_info()

{

int pos,f=0;

node *p;

cout<<"\nEnter the element:";

cin>>item;
```

```
temp=start;
if(start==NULL)
cout<<"\nLL is Empty:";
if(start->info==item)
{
start=start->link;
f=1;
}
while(temp!=NULL)
{
p=temp;
temp=temp->link;
if(temp->info==item)
{
f=1;
p->link=temp->link;
}
}
if(f==0)
cout<<"\n node is not found";
}
void main()
```

```

{
clrscr();

node n;

int ch;

cout<<"\n1.Insert 2.Display 3.Del_position 4.Del_information 5.exit:\n";

while(ch!=5)

{

cout<<"\nEnter choice";

cin>>ch;

switch(ch)

{

case 1: n.insert(); break;

case 2: n.dis(); break;

case 3: n.del_pos(); break;

case 4: n.del_info(); break;

16

case 5: exit(0);

}

}

getch();

}

```

- Implement Doubly Link List

```
#include<iostream.h>

#include<conio.h>

#include<process.h>

class node

{

int info,c,j;

node *left,*right;

public:

void insert();

void display();

void del();

};

node *start=NULL,*temp=NULL,*move=NULL, *temp1=NULL;

void node::insert()

{

int item;

node *p=new node;

cout<<"\nEnter element:";

cin>>item;

p->info=item;

p->left=NULL;

p->right=NULL;
```

```
if(start==NULL)
{
start=p;
return;
}
else
{
temp=start;
while(temp->right!=NULL)
temp=temp->right;
temp->right=p;
p->left=start;
}
}

void node::display()
{
move=start;
if(move==NULL)
{
cout<<"\n LL Empty:";
return;
}
```

```

else
{
cout<<"\n node in DLL are :";
while(move!=NULL)
{
cout<<move->info<<"\t";
move=move->right;
}
}
18
}

void node::del()
{
if(start==NULL)
{
cout<<"\n LL Empty:";
return;
}

temp=start;
start=temp->right;
start->left=NULL;
temp->right=NULL;

```

```
cout<<"\n deleted element is"<<temp->info;
}
void main()
{
clrscr();
node n;
int ch;
cout<<"\n1. Insert 2. Display 3.Delete 4. Exit";
while(ch!=4)
{
cout<<"\nEnter choice";
cin>>ch;
switch(ch)
{
case 1: n.insert(); break;
case 2: n.display(); break;
case 3: n.del(); break;
case 4: exit(0);
}
}
getch();
}
```



- **Perform Bubble Sort**

```
#include<iostream.h>
```

```
#include<conio.h>
```

```
class arr
```

```
{
```

```
    int a[10],n,m;
```

```
    public:
```

```
        void get();
```

```
        void bsort();
```

```
        void disp();
```

```
};
```

```
void arr::get()
```

```
{
```

```
    cout<<"Enter Size of array"<<endl;
```

```
    cin>>n;
```

```
    cout<<"Enter elements "<<endl;
```

```
    for(int i=0;i<n;i++)
```

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
    cout<<"Elements are "<<endl;
```

```
    for(i=0;i<n;i++)
```

```

        {
            cout<<a[i]<<endl;
        }
    }
void arr::bsort()
{
    for(int i=0;i<n;i++)
    {
        for(int j=0;j<n-1;j++)
        {
            if(a[j]>a[j+1])
            {
                m=a[j+1];
                a[j+1]=a[j];
                a[j]=m;
            }
        }
    }
}
void arr::disp()
{
    cout<<"Sorted elements are"<<endl;

```

```

        for(int i=0;i<n;i++)
        {
            cout<<a[i]<<endl;
        }
    }

```

```

void main()

```

```

{
    clrscr();

    arr o;

    o.get();

    o.bsort();

    o.disp();

    getch();
}

```

- **Perform Selection Sort**

```

#include<iostream.h>

```

```

#include<conio.h>

```

```

class arr

```

```

{
    int a[10],n,m;

    public:

    void get();

```

```
void ssort();

void disp();

};

void arr::get()
{
    cout<<"Enter Size of array"<<endl;
    cin>>n;
    cout<<"Enter elements "<<endl;
    for(int i=0;i<n;i++)
    {
        cin>>a[i];
    }
    cout<<"Elements are "<<endl;
    for(i=0;i<n;i++)
    {
        cout<<a[i]<<endl;
    }
}

void arr::ssort()
{
    for(int j=0;j<n;j++)
    {
```

```

    int min=j;
    for(int k=j+1;k<n;k++)
    {
        if(a[k]<a[min])
        {
            min=k;
        }
    }
    if(min!=j)
    {
        m=a[j];
        a[j]=a[min];
        a[min]=m;
    }
}

void arr::disp()
{
    cout<<"Sorted elements are"<<endl;
    for(int i=0;i<n;i++)
    {
        cout<<a[i]<<endl;
    }
}

```

```

        }
    }
void main()
{
    clrscr();
    arr o;
    o.get();
    o.ssort();
    o.disp();
    getch();
}

```

- **Implement Insertion Sort**

```

#include<iostream.h>

#include<conio.h>

class arr
{
    int a[10],n;
    public: void get();
    void isort();
    void disp();
};

void arr::get()

```

```

{
    cout<<"Enter Size of array"<<endl;
    cin>>n;
    cout<<"Enter elements "<<endl;
    for(int i=0;i<n;i++)
    {
        cin>>a[i];
    }
    cout<<"Elements are "<<endl;
    for(i=0;i<n;i++)
    {
        cout<<a[i]<<endl;
    }
}

```

```

void arr::isort()

```

```

{
    for(int j=1;j<=n;j++)
    {
        int t=a[j];
        int k=j-1;
        while(k>=0 && a[k]>t)
        {

```

```

        a[k+1]=a[k];

        k--;

    }

    a[k+1]=t;

}

}

void arr::disp()

{

    cout<<"Sorted elements are"<<endl;

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

    {

        cout<<a[i]<<endl;

    }

}

void main()

{

    clrscr();

    arr o;

    o.get();

    o.isort();

    o.disp();

    getch();

```



```
}
```

- **Implement Linear and Binary Search**

```
#include<iostream.h>
```

```
#include<conio.h>
```

```
#include<process.h>
```

```
class demo
```

```
{
```

```
int a[10],i,j,n,f,temp,ele,demo,mid,low,high;
```

```
public:
```

```
void get();
```

```
void sort();
```

```
void linear();
```

```
void binary();
```

```
void dis();
```

```
};
```

```
void demo::get()
```

```
{
```

```
cout<<"\n Enter n:";
```

```
cin>>n;
```

```
cout<<"\nEnter array Elements:";
```

```
for(i=1;i<=n;i++)
```

```
    cin>>a[i];
}

void demo::linear()
{
    int ele;

    cout<<"\nEnter the element to be search";
    cin>>ele;

    for(i=1;i<=n;i++)
    {
        if(a[i]==ele)
        {
            cout<<"\nSuccessful search";
            cout<<"\nElement is found at position "<<i;
            return;
        }
    }

    if(i>n)
    {
        cout<<"\nUnsuccessful search:";
        cout<<"\nElement is not found ";
    }
}
```

```
void demo::sort()
{
for(i=1;i<=n;i++)
{
    for(j=1;j<=n-1;j++)
    {
        if(a[j]<a[j+1])
        {
temp=a[j];
a[j]=a[j+1];
26
a[j+1]=temp;
        }
    }
}
}
```

```
void demo::binary()
{
cout<<"\nEnter element to be search ";
cin>>ele;
f=0;
low=1;
```

```
high=n;
while(low<=high)
{
mid=(low+high)/2;
if(a[mid]==ele)
{
f=1;
cout<<"\nElement is found at :"<<mid;
return;
}
else if(a[mid]<ele)
low=mid+1;
else if(a[mid]>ele)
high=mid-1;
}
if(f==0)
cout<<"\n Element is not found:";
}
void demo::dis()
{
cout<<"\n Element are \n";
for(i=1;i<=n;i++)
```

```
cout<<a[i]<<"\t";  
}  
void main()  
{  
clrscr();  
demo d;  
int ch;  
d.get();  
d.dis();  
cout<<"\n 1:Linear 2:Binary 3:exit\n";  
while(ch!=3)  
{  
cout<<"\nEnter Choice:";  
cin>>ch;  
switch(ch)  
{  
case 1: d.linear(); break;  
case 2: d.sort();  
d.dis();  
d.binary(); break;  
case 3: exit(0); break;  
}
```

```
}
```

```
getch();
```

```
}
```

- **Implement Tower of Hanoi**

```
#include<iostream.h>
```

```
#include<conio.h>
```

```
class demo
```

```
{
```

```
int n;
```

```
public:
```

```
void tower(int,char,char,char);
```

```
void get();
```

```
};
```

```
void demo::get()
```

```
{
```

```
cout<<"\nEnter the number of disk: ";
```

```
cin>>n;
```

```
tower(n,'A','B','C');
```

```
}
```

```
void demo::tower(int n,char beg,char aux,char end)
```

```
{
```

```
if(n!=0)
```

```

{
tower(n-1,beg,end,aux);
cout<<"\n Move disk "<<n<<" from "<<beg<<" to "<<end<<"\n";
tower(n-1,aux,beg,end);
}
}

void main()
{
clrscr();
demo d;
d.get();
getch();
}

```

- **Finding Factorial of Number**

```

#include<iostream.h>

#include<conio.h>

class factorial
{
double n,f;
public:
void get();
double fact(double);

```

```

};

void factorial::get()

{
    cout<<"\nEnter n";

    cin>>n;

    f=fact(n);

    cout<<"\n Factorial of "<<n<<" is "<< f;

}

double factorial::fact(double n)

{
    if(n==0)

    return 1;

    else

    return(n*fact(n-1));

}

void main()

{

    clrscr();

    factorial f;

    f.get();

    getch();

}

```