SREE NARAYANA GURU COLLEGE

An Autonomous College, Affiliated to Bharathiar University
Accredited with "A" grade by NAAC (3rd cycle) & an ISO 9001 : 2015 Certified Institution
Approved by Govt. of Tamil Nadu, & Recognized by UGC
A Premier Post Graduate and Research Co- Educational Institution
K.G. Chavadi, Coimbatore - 641105

I M.Sc. Computer Science

Practical I: ALGORITHM AND OOPS LAB (13P)

PRACTICAL RECORD

2025-2026

DEPARTMENT OF COMPUTER SCIENCE
SREE NARAYANA GURU COLLEGE
K.G. CHAVADI, COIMBATORE – 641 105

SREE NARAYANA GURU COLLEGE
K.G. CHAVADI, COIMBATORE – 641 105

CERTIFICATE

REGISTER NUMBER

This is to certify that the bonafide record work done by _____of I M.Sc. Computer Science in the Computer Lab of the College during the year 2025-2026.

Staff In-charge **Head of the Department**

Submitted for the End Semester Examination held on ..... at Sree Narayana Guru College, Coimbatore – 641105.

Internal Examiner **External Examiner**

INDEX

ALGORITHM AND OOPS LAB

S.No	Date	Program Name	Pg No	Signature
1		**Towers Of Hanoi**		
2		**Binary Search Tree Using Traversals**		
3		**Stack Operations**		
4		**Circular Queue**		
5		**Quick Sort**		
6		**Heap Sort**		
7		**Knapsack Problem**		
8		**Divide And Conquer**		
9		**Eight Queens**		
10		**Virtual Function**		
11		**Parameterized Constructor**		
12		**Friend Function**		
13		**Function Overloading**		
14		**Single Inheritance**		
15		**Employee Details Using Files**		

Program:1

Date: **TOWERS OF HANOI**

SOURCE CODE:

```
#include<iostream.h>
#include<conio.h>
void towers(int, char, char,char);
int main ()
{
clrscr();
int num;
cout<<"Enter the number of disks:";
cin>>num;
cout<<"\n The sequence of moves involved in tower of Hanoi is:";
towers(num,"A","C","B");
getch();
return 0;
}
void towers(int num,char frompeg,char topeg,char auxpeg) {
if(num==1)
{
cout<<"\n Move disk1 frompeg"<<frompeg<<"topeg"<<topeg;
return;
}
towers (num-1,frompeg,auxpeg,topeg);
cout<<"\n Move disk"<<num<<"frompeg"<<frompeg<<"topeg"<<topeg;
towers (num-1, auxpeg,topeg, frompeg);
}
```

OUTPUT:

Enter the number of disks: 3

The sequence of moves involved in tower of Hanoi is:

Move disk1 frompegA topegC

Move disk2 frompegA topegB

Move disk1 frompegC topegB

Move disk3 frompegA topegC

Move disk1 frompegB topegA

Move disk2 frompegB topegC

Move disk1 frompegA

topegC

Program:2

Date:

BINARY SEARCH TREE USING TRAVERSE

SOURCE CODE:

```
#include<iostream.h>
#include<conio.h>
struct tree
{
tree *l,*r;
int data;
int value;
}*root=NULL,*p=NULL
; void create(tree *root)
{
int value;
char ch;
if(root==NULL)
{
root=new tree;
cout<<"\n Enter the value of root
node:"; cin>>root->data;
root-
>r=NULL;
root-
>l=NULL;
}
do
{
p=root;
cout<<"\n Enter the value of node:";
```

```
cin>>value;
while(root)
{
if(value<p->data)
{
if(p->l==NULL)
{
p->l=new
tree; p=p->l;
p-
>data=value;
p->l=NULL;
p->r=NULL;
cout<<"\n Value entered in  left";
break;
}
else if(p->l!=NULL)
{
p=p->l;
}
}
if(value>p->data)
{
if(p->r==NULL)
{
p->r=new
tree; p=p->r;
p->data=value;
```

```
p->l=NULL;
p->r=NULL;
cout<<"\n Value entered in right";
break;
}
else if(p->r!=NULL)
{
p=p->r;
}
}
}
cout<<"\n Do you want to continue";
cin>>ch;
}
while(ch=="y"||ch=="Y");
}
void inorder(tree *p)
{
if(p!=NULL)
{
inorder(p->l);
cout<<p-
>data<<endl;
inorder(p->r);
}
}

void preorder(tree *p)
```

```
        {
        if(p!=NULL)
        {
        cout<<p-
        >data<<endl;
        preorder(p->l);
        preorder(p->r);
}
        }
        void postorder(tree *p)
        {
        if(p!=NULL)
        {
        postorder(p->l);
        postorder(p->r);
        cout<<p-
        >data<<endl;
        }
        }
        void main()
        {
        clrscr();
        create(root);
        cout<<"\n Printing traversal in inorder";
        inorder(root);
        cout<<"\n Printing traversal in
preorder"; preorder(root);
        cout<<"\n Printing traversal in postorder";
        postorder(root);
```

```
getch();
}
```

OUTPUT:

```
Enter the value of root node:  10
Enter the value of node: 5 Value
entered in left
Do you want to continue: y
Enter the value of node: 20
Value entered in right
Do you want to continue: n
Printing traversal in inorder:
5
10
20
Printing traversal in preorder: 10
5
20
Printing traversal in postorder: 5
20
10
```

Program:3
Date:

STACK OPERATIONS

SOURCE CODE:

```
#include<iostream.h>
   #include<conio.h>
   #include<process.h>
   class stack
   {
   private:
   struct node
   {
   int info;
   node *next;
   }*top,*temp,*ptr,*x;
   public:
   stack()
   {
   top=temp=ptr=NULL;
   }
   void
   push(int);
   void pop();
   void peep();
   };
   void stack :: push(int n)
   {
   ptr=new
   node; ptr-
   >info=n;
   ptr->next=NULL;
```

```
if(ptr==NULL)
{
cout<<"\n Cannot create any new
node"; getch();
}
if(top==NULL)
top=ptr;
else
{
temp=top;
top=ptr;
ptr->next=temp;
}
}
void stack :: pop()
{
if (top==NULL)
cout<<"\n Stack empty";
else
{
ptr=top;
top=top-
>next;
cout<<"\n Deleted element is"<<ptr- >info;
delete ptr;
}
}
void stack :: peep()
```

```
{
x=top;

while(x!=NULL)
{
cout<<x-
>info<<"\n"; x=x-
>next;
}
}
void main()
{
stack obj;
char
choice; int
ch,data; do
{
cout<<"\n Stack
Operations";
cout<<"\n**********"
; cout<<"\n 1.Push";
cout<<"\n 2.Pop";
cout<<"\n 3.Peep";
cout<<"\n 4.Exit";
cout<<"\n Which operation do you want to perform?";
cin>>ch;
switch(ch)
{
```

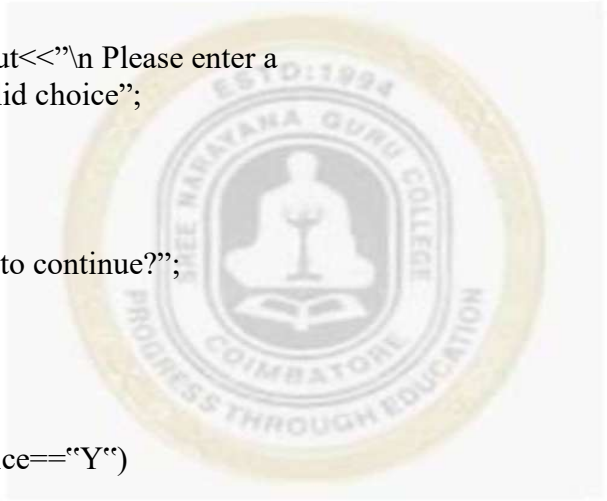
```
        case 1:
                cout<<"\n Enter the data you want to push";
                cin>>data;
                obj.push(data);
                break
 case 2:                        }
                                obj.pop();
                                break;

case 3:
                                obj.peep();
                                break;

case 4:
                                exit(0);
                                break;

default
:                               cout<<"\n Please enter a
                                valid choice";


        cout<<"\n Do you want to continue?";
        cin>>choice;
        }
        while(choice=="y"||choice=="Y")
        ; };
```



OUTPUT:

Stack Operations

****

1. Push

2. Pop

3. Peep

4. Exit

Which operation do you want to perform?

1 Enter the data you want to push 3 Do

you want to continue? y

Stack Operations

****

1. Push

2. Pop

3. Peep

4. Exit

Which operation do you want to perform?

1 Enter the data you want to push 6 Do

you want to continue? y

Stack Operations

****

1. Push

2. Pop

3. Peep

4. Exit

Which operation do you want to perform? 1

Enter the data you want to push

9 Do you want to continue? y

Stack Operations

****

1. Push

2. Pop

3. Peep

4. Exit

Which operation do you want to perform?

3 9

6

3

Do you want to continue? y

Stack Operations

****

1. Push

2. Pop

3. Peep

4. Exit

Which operation do you want to perform?

2 Deleted element is 9

Do you want to continue? No

Program:4

Date:

CIRCULAR QUEUE

SOURCE CODE:

```
#include<iostream.h>
#include<conio.h>
#include<process.h>
class circular_queue
{
private:
int *cqueue_arr;
int front,rear;
public:
circular_queue()
{
cqueue_arr=new
int[MAX]; rear=front=-1;
}
void insert(int item)
{
if((front==0 && rear==MAX-1)||(front==rear+1))
{
cout<<"Queue overflow \n";
return;
}
if(front==-1)
{
front=0;
rear=0;
}
else
```

```
{
if(rear==MAX-1)
rear=0;
else
rear=rear+1;
}
cqueue_arr[rear]=item;
}
void del()
{
if(front==-1)
{
cout<<"Queue underflow \n";
return;
}
cout<<"Element deleted from queue is:"<<cqueue_arr[front]<<"\n";
if(front==rear)
{
front=-
1; rear=-
1;
}
else
{
if(front==MAX-1)
front=0;

else
```

```
        front=front+1;
}
        }
        void display()
        };
        void circular_queue :: display()
        {
        int
        front_pos=front,rear_pos=rear;
        if(front==-1)
        {
        cout<<"Queue is empty \n";
        return;
        }
        cout<<"Queue elements:";
        if(front_pos<=rear_pos)
        {
        while(front_pos<=rear_pos)
        {
        cout<<cqueue_arr[front_pos]<<" ";
        front_pos++;
        }
        }
        else
        {
        while(front_pos<=MAX-1)
        {
        cout<<cqueue_arr[front_pos]<<" ";
```

```
front_pos++;
}

front_pos;
while(front_pos<=rear_pos)
{
cout<<cqueue_arr[front_pos]<<"
"; }
cout<<endl;
}
}
int main()
{
clrscr();
int choice,item;
circular_queue
cq; do
{
cout<<"\n 1.Insert";
cout<<"\n 2. Delete";
cout<<"\n
3.Display";
cout<<"\n 4.Exit";
cout<<"\n Enter your choice";
cin>>choice;
switch(choice)
{

case1:
```

```
        cout<<"\n Input the elements for insertion in queue:";
        cin>>item;
        cq.insert(item);
                case4:cq.del();
                break;
case2:
                break;
                cq.disply();
                break;
case3:
                exit(0);
                break;
default:
        cout<<"\n Wrong
choice"; }
}
while(choice!=4);
getch();
return 0;
}
```

OUTPUT:

1. Insert

2. Delete

3. Display

4. Exit

Enter your choice: 1

Input the elements for insertion in queue: 2

1. Insert

2. Delete

3. Display

4. Exit

Enter your choice: 1

Input the elements for insertion in queue: 4

1. Insert
2. Delete
3. Display
4. Exit

Enter your choice: 3

Queue elements: 2 4

1. Insert
2. Delete
3. Display
4. Exit

Enter your choice: 2

Element deleted from queue is: 2

1. Insert
2. Delete
3. Display
4. Exit

Enter your choice: 4

Program: 5
Date:

QUICK SORT

SOURCE CODE:

```
#include<iostream.h>
#include<conio.h>
int part(int low, int high, int *a)
{
int
i,h=high,l=low,p,t;
p=a[low];
while(low<high)
{
while(a[l]<p)
{
l++;
}
while(a[h]>p)
{
h--;
}
if(l<h)
{
t=a[l];
a[l]=a[h];
a[h]=t;
}
else
{
```

```
t=p;

p=a[l];
a[l]=t;

break;

}

}

return h;

}

void quick(int l,int h,int *a)

{

int index,i;

if(l<h)

{

index=part(l,h,a);

quick(l,index-1,a);

quick(index+1,h,a)

;

}

}

int main()

{

clrscr();

int a[100],n,l,h,i;

cout<<"\n Enter the number of elements:";

cin>>n;

cout<<"\n Enter the elements:";

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

cin>>a[i];

cout<<"\n Initial Array : \n";
```

```
for(i=0;i<n;i++)
{
cout<<a[i]<<"\t";

}
h=n-1;
l=0;
quick(l,h,a);
cout<<"\n After Sorting: \n";
for(i=0;i<n;i++)
{
cout<<a[i];
}
getch();
return 0;
}
```

OUTPUT

Enter the number of elements: 5

Enter the elements: 10 1 9 3 5

Initial Array:

10 1 9 3 5

After Sorting:

1 3 5 9 10

Program:6
Date:

HEAP SORT

SOURCE CODE:

```
#include<iostream.h>
#include<conio.h>
void heapify(int arr[],int n,int i)
{
int largest=i;
int l=2*i+1;
int r=2*i+2;
if(l<n && arr[l]>arr[largest])
largest=l;
if(r<n && arr[r]>arr[largest])
largest=r;
if(largest!=i)
{
int swap=arr[i];
arr[i]=arr[largest];
arr[largest]=swap;
}
}
void heapsort(int arr[],int n)
{
for(int i=n/2-1;i>=0;i--)
heapify(arr,n,i);
for(int i=n-1;i>=0;i--)
{
int swap=arr[0];
```

```
        arr[0]=arr[i];
        arr[i]=swap;
        heapify(arr,i,0);
        }
        }
        void printarray(int arr[],int n)
        {
        for(int
        i=0;i<n;i++)
        cout<<arr[i]<<" ";
        cout<<"\n";
        }
        void main()
        {
        clrscr();
        int arr[30],n;
        cout<<"\n Enter the number of elements:";
        cin>>n;
        cout<<"\n Enter the
        elements:"; for(int
        i=0;i<n;i++) cin>>arr[i];
        cout<<"\n Initial Array: \n";
        printarray(arr,n);
        heapsort(arr,n);
        cout<<"\n Sorted array is: \n";
        printarray(arr,n);
        getch();
}
```

OUTPUT:

Enter the number of elements: 5

Enter the elements: 10 2 11 6 22

Initial Array:

10 2 11 6 22

Sorted array is:

2 6 10 11 22

Program:7

Date: **KNAPSACK PROBLEM**

SOURCE CODE

```
#include<iostream.h>
#include<conio.h>
void knapsack(int n,float weight[],float profit[],float capacity)
{
float x[20],tp=0;
int i,j,u;
u=capacity;
for(i=0;i<n;i++)
x[i]=0.0;
for(i=0;i<n;i++)
{
if(weight[i]>u
) break;
else
{
x[i]=1.0;
tp=tp+profit[i]
; u=u
weight[i];
}
}
if(i<n)
x[i]=u/weight[i];
tp=tp+(x[i]*profit[i]);
cout<<"\n The result vector
is:"; for(i=0;i<n;i++)
```

```
cout<<"\t"<<x[i];
cout<<"\n Maximum profit is"<<tp;

}
void main()
{
clrscr();
float weight[20],profit[20],capacity;
int num,i.,j;
float ratio[20],temp;
cout<<"\n Enter the number of
objects:"; cin>>num;
cout<<"\n Enter the weights and profits of each object:";
for(i=0;i<num;i++)
{
cin>>weight[i]>>profit[i];
}
cout<<"\n Enter the capacity of
knapsack:"; cin>>capacity;
for(i=0;i<num;i++)
{
ratio[i]=profit[i]/weight[i];
}
for(i=0;i<num;i++)
{
for(j=i+1;j<num;j++)
{
if(ratio[i]<ratio[j])
```

```
{
temp=ratio[j];
ratio[j]=ratio[i];
ratio[i]=temp;
temp=weight[j];
weight[j]=weight[i];
weight[i]=temp;
temp=profit[j];
profit[j]=profit[i];
profit[i]=temp;
}
}
}
knapsack(num,weight,profit,capacity);
getch();
}
```

OUTPUT:

Enter the number of objects: 3

Enter the weights and profit of each object:

1 2

3

4

5

6

Enter the capacity of knapsack: 3 The

result vector is: 1.000000 0.666667

0.000000 Maximu profit is: 4.666667

Program:8

Date: **DIVIDE AND CONQUER**

SOURCE CODE

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr();
int n, a [30], i, j,top,mid,bottom,item;
cout<<"Enter how many elements you want: ";
cin>>n;
cout<<"\n Enter the "<<n<<" elements in ascending order:";
for(i=0;i<n;i++)
{
cin>>a[i];
}
cout<<"\n Enter the item to search:";
cin>>item;
bottom=1;
top=n;
do
{
mid=(bottom+top)/2
; if(item<a[mid])
top=mid-1;
else
if(item>a[mid]

)
bottom=mid+1
```

```
;
}
while(item!=a[mid] && bottom<=top);

if(item==a[mid])
{
cout<<"\n Binary search successful";
cout<<"\n"<<item<<" found at
position"<<mid+1; }
else
{
cout<<"\n Search failed, not found";
}
getch();
}
```

OUTPUT:

Enter how many elements you want: 5

Enter the elements in ascending order: 11 22 33 44 55

Enter the item to search: 44

Binary search successful

44 found at position 4

Program:9

Date:

EIGHT QUEENS

SOURCE CODE;

```
#include<iostream.h
> #include<conio.h>
#include<math.h>
char a[10][10];
int n;
void printmatrix ()
{
int i, j;
cout<<"\n";
for(i=0;i<n;i++)
{
for(j=0;j<n;j++)
cout<<a[i][j];
cout<<"\n \n";
}
}
int getmarkedcol(int row)
{
int i;
for(i=0;i<n;i++)
if(a[row][i]=="Q")
{
return(i);
}
}
int feasible(int row,int col)
```

```
{
int i,tcol;
for(i=0;i<n;i++)
{
tcol=getmarkedcol(i);
if(col==tcol||abs(row-i)==abs(col
tcol)) return 0;
}
return 1;
}
void nqueen(int row)
{
int i,j;
if(row<n)
{
for(i=0;i<n;i++)
{
if(feasible(row,i))
{
a[row][i]="Q";
nqueen(row+1)
; a[row][i]=".";
}
}
}
else
{
cout<<"\n The solution is:";
```

```
printmatrix();
}
}
void main()
{
clrscr();
int i,j;
cout<<"\n Enter the number of queens:";
cin>>n;
for(i=0;i<n;i++
)
for(j=0;j<n;j++
) a[i][j]=".";
nqueen(0);
getch();
}
```

OUTPUT:

Enter the number of queens: 4

The solution is:

. Q . .

. . . Q

Q . . .

. . Q .

The solution is:

. . Q .

Q . . .

. . . Q

. Q . .

Program:10

Date:

VIRTUAL FUNCTION

SOURCE CODE:

```
#include<iostream.h>
#include<conio.h>
class shape
{
public:
virtual void get(){}
virtual void area(){}
};
class square:public shape
{
public:
float a;
void get()
{
cout<<"\n enter the size of a square :";
cin>>a;
}
void area()
{
float res;
res=a*a;
cout<<"\n area of square:"<<res;
}
};
class triangle:public shape
{
public:
float h,b;
void get()
{
cout<<"\n enter the height and base of a triangle";
cin>>h>>b;
}
void area()
{
float res;
res=0.5*h*b;
cout<<"\n area of triangle:"<<res;
}
```

```
        };
class rectangle:public shape
        {
        public:
        float l,b;
        void get()
        {
        cout<<"\n enter the length and breadth of rectangle:";
        cin>>l>>b;
        }
        void area()
        {
        float res;
        res=l*b;
        cout<<"\n area of rectangle:"<<res;
        } };
        void main()
        {
        shape A;
        shape
        *A1;
        A1=&A;
        A1->get();
        A1->area();
        square B;
        square
        *B1;
        B1=&B;
        B1->get();
        B1->area();
        triangle C;
        triangle *C1;
        C1=&C;
        C1->get();
        C1->area();
        rectangle D;
        rectangle
        *D1; D1=&D;
        D1->get();
        D1->area();
        getch(); }
```

OUTPUT:

Enter the size of a square:2

Area of square:4

Enter the height and base of a triangle:3 6

Area of triangle:9

Enter the length and breadth of a triangle:5 4

Area of a rectangle:20

Program:11

Date:

PARAMETERIZED CONSTRUCTOR

SOURCE CODE:

```
#include<iostream.h>
#include<conio.h>
class fibonacci
{
private:
int i,k,next;
public:
fibonacci(int
n)
{
k=n;
}
void calc();
};
void fibonacci::calc()
{
int f1=0,f2=1;
cout<<f1<<"\n";
cout<<f2<<"\n";
for(i=3;i<=k;i++)
{
next=f1+f2;
f1=f2;
f2=next;
cout<<next<<”\n”;
}
}
void main()
{
clrscr();
int p;
cout<<"\n enter number \n";
cin>>p;
fibonacci para(p);
para.calc();
getch();
}
```

OUTPUT:

Enter

number:50 1

1

2

3

Program:12

Date:

FRIEND FUNCTION

SOURCE CODE:

```
#include<iostream.h>
#include<conio.h>
class C;
class B;
class A
{
private:
int x;
public:
void getx()
{
cout<<"\n ENTER VALUE:";
cin>>x;
}
friend int largest(A a,B b,C c)
};
class B
{
private:
int y;
public:
void gety()
{
cout<<"\n ENTER VALUE:";
cin>>y;
}
friend int largest(A a,B b,C c)
};
class C
{
private:
int z;
public:
void getz()
{
cout<<"\n ENTER VALUE:";
cin>>z;
}
}

friend int largest(A a,B b,C c)
```

```
};
int largest(A a,B b,C c)
{
if((a.x>b.y)&&(a.x>c.z))
{
return a.x;
}
else if(b.y>c.z)
{
return b.y;
}
else
{
return c.z;
}
}
void main()
{
clrscr();
A obj1;
obj1.getx();
B obj2;
obj2.gety();
C obj3;
obj3.getz();
cout<<"\n THE LARGEST NUMBER
IS "<<largest(obj1,obj2,obj3); getch(); }
```

OUTPUT:

ENTER VALUE:4

ENTER VALUE:5

ENTER VALUE:7

THE LARGEST NUMBER IS:7

Program:13

Date:

FUNCTION OVERLOADING

SOURCE CODE

```
 #include<iostream.h
> #include<conio.h>
#include<math.h>
#define pi 3.14
float res;
float area(float r)
 {
res=pi*r*r;
cout<<"\n area of circle="<<res;
return 0;
 }
void area(float l,float b)
 {
float res;
res=l*b;
cout<<"\n area of rectangle"<<res;
 }
void area(float a,float h,float ba)
 {
float res;
res=a*h*ba;
cout<<" \n area of triangle="<<rea;
 }
void main()
 {
int ch;
cout<<"\n enter your choice:";
cin>>ch;
switch(ch)
 {
case 1:
float r;
cout<<"\n enter radius of
circle:"; cin>>r;
area(r);
break;
case 2:
float l,b;
cout<<"\n enter the length and breath of
rectangle:";  cin>>l>>b;
area(l,b);
break;
case 3:
```

```
float h,ba;

cout<<"\n enter height and base of triangle:";


cin>>h>>ba;
area(0.5*h,ba)
; break;
case 4:
break;
}
getch();
}
```

OUTPUT:

Enter your choice:1

Enter radius of circle: 4

Area of circle=50.24

Enter your choice:2

Enter length and breath of rectangle:

2 4 Area of rectangle=8

Enter your choice:3

Enter height and base of

triangle: 3 4 Area of triangle

=6

.

Program no:14

Date:

SINGLE INHERITANCE

SOURCE CODE:

```
#include<iostream.h>
#include<conio.h>
class employee
{
private:
int empno;
char name[20],designation[20];
public:
double salary;
void getdata()
{
cout<<"\n enter employee number";
cin>>empno;
cout<<"\n enter employee name";
cin>>name;
cout<<"\n enter employee designation";
cin>>designation;
cout<<"\n enter employee salary";
cin>>salary;
}
void display()
{
cout<<"\n Pay Slip:";
cout<<"\n employee number:"<<empno;
cout<<"\n employee name:"<<name;
cout<<"\n employee
designation:"<<designation; cout<<"\n
employee salary:"<<salary;
}
};
class allowance:public employee
{
private:
int hra,pf;
public:
void cal()
{
hra=salary*12/100;
pf=salary*8/100;
}
void display1()
{
cout<<"\n provident fund:"<<pf;
cout<<"\n house rent allowance:"<<hra;
```

```
}


};
void main()
{
allowance
Empdata;
Empdata.getdata
();
Empdata.cal();
Empdata.display
();
Empdata.display
1();  getch();
}
```

OUTPUT:

Enter employee number 101

Enter employee name aki

Enter employee designation

GM Enter employee salary

30000 Pay slip:

Employee number:101

Employee name:aki

Employee

designation:GM

Employee salary:30000

Provident fund:2400

House rent

allowance:3600

Program:15

Date:

EMPLOYEE DETAILS USING FILES

SOURCE CODE:

```
#include<iostream.h>
#include<fstream.h>
#include<conio.h>
void main()
{
char data[100];
int line;
ofstream
outfile;
outfile.open("emp.txt");
cout<<"Enter the name of
employee:"; cin.getline(data,10);
outfile<<data<<endl;
cout<<"Enter the
id:";
cin.getline(data,10);
outfile<<data<<endl;
cout<<"Department:"
; cin.getline(data,10);
outfile<<data<<endl;
cout<<"Salary:";
cin.getline(data,10);
outfile<<data<<endl;
outfile.close();
ifstream infile;
infile.open("emp.txt")
;
```

```
        cout<<"\n Reading
        from file\n";
        infile>>data;
        cout<<data<<endl;
        infile>>data;
 cout<<data<<endl;
infile>>data;
cout<<data<<endl;
infile>>data;
cout<<data<<endl;
infile.close();
getch();
        }
```

OUTPUT:

Enter the name of employee:

Raju Enter the id: 1

Department:

ProductionSalary: 20000

Reading from file

Raju

1

Production

2000