**PRACTICAL NO: 01**

**Aim:** Program to demonstate use of data member & member function

**Code:**

#include<iostream>

class circle

{

float r,a;

public:

void read()

{

std::cout<<"Enter radius:";

std::cin>>r;

}

void compute()

{

a=3.14\*r\*r;

}

void display()

{

std::cout<<"Area="<<a;

}

};

int main()

{

circle c;

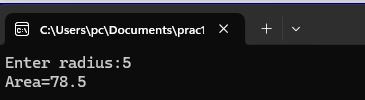
c.read();

c.compute();

c.display();

}

**Output:-**

****

**PRACTICAL NO: 02**

**Aim:** Program based on branching and looping statements using classes

**INTERNAL MEMBER FUNCTION:**

**Code:-**

#include<iostream>

class series

{

int n,i,sum;

public:

void read()

{

std::cout<<"Enter the value of n:";

std::cin>>n;

}

void compute()

{

for(i=1, sum=0;i<=n;i++)

{

sum=sum+i\*i;

}

}

void display()

{

std::cout<<"Value of the series="<<sum;

}

};

int main()

{

series s;

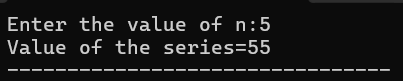
s.read();

s.compute();

s.display();

}

**Output:-**

****

**PRACTICAL NO: 03**

**Aim:** Program to demonstate one and two dimentional aarays using classes

**Array of pointer to object**

**Code:-**

#include<iostream.h>

#include<conio.h>

class circle

{

float r,a;

public:

void read();

void compute();

void display();

};

Inline void circle::read()

{

cout<<"Enter radius:";

cin>>r;

}

inline void circle::compute()

{

a=3.14\*r\*r;

}

inline void circle::display()

{

cout<<"Area="<<a<<endl;

}

void main()

{

clrscr();

circle \*p[50];

circle \*c;

int n,i;

cout<<"Enter the number of objects:":

cin>>n;

for(i=0;i<=n-1;i++)

{

c= new circle;

c->read0;

c->compute();

p[i]=c;

}

for(i=0;i<=n-1;i++)

{

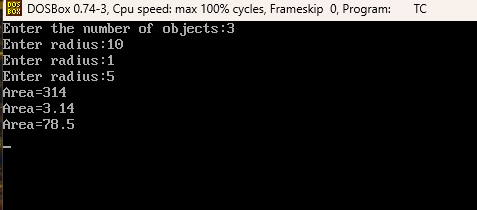
p[i]->display();

}

getch();

}

**Output:-**

****

**PRACTICAL NO : 04**

**Aim:** Program to use scope resolution operator. Display the various values of the same variables declared at different scope levels.

**Inline Functions:**

**Code:-**

#include<iostrem.h>

#include<conio.h>

Class circle

{

Float r,a;

Public:

Void read();

Void compute();

Void display();

};

Inline void circle::read()

{

Cout<<”Enter radius:”;

Cin>>r;

}

Inline void circle::compute()

{

A=3.14\*r\*r;

}

Inline void circle::display()

{

Cout<<”Area=”<<a;

}

Void main()

{

Clrscr();

Circle c;

c.read();

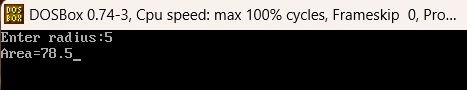
c.compute();

c.display();

getch();

}

**Output:**

****

**PRACTICAL NO: 05**

**Aim:** Program to demonstate various types of constuctors and destuctors.

**A)**

#include <iostream>

class circle

{

float r,a;

public:

circle()

{

std::cout<<"Enter the value of radius:";

std::cin>>r;

}

void compute();

void display();

};

inline void circle::compute()

{

a=3.14\*r\*r;

}

inline void circle::display()

{

std::cout<<"Area="<<a;

}

int main()

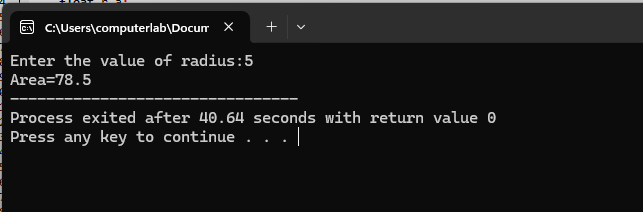
{

circle c;

c.compute();

c.display();

}



**B)**

**Code:**

#include<iostream>

#include<conio.h>

class circle

{

float r,a;

public:

circle(float x)

{

r=x;

}

void compute();

void display();

};

inline void circle::compute()

{

a=3.14\*r\*r;

}

inline void circle::display()

{

std::cout<<"Area="<<a;

}

int main()

{

float p;

std::cout<<"Enter the radius of the circle";

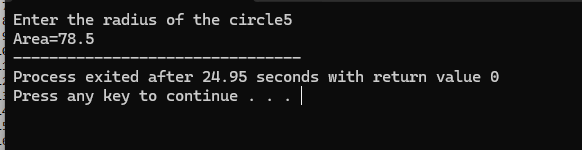
std::cin>>p;

circle c(p);

c.compute();

c.display();

}

**Output:** 

**C)**

**Code:**

#include<iostream>

#include<conio.h>

class test

{

int\*p;

public:

test()

{

p=new int;

}

void read()

{

std::cout<<"Enter a number";

std::cin>>\*p;

}

void display()

{

std::cout<<"Value="<<\*p<<std::endl;

}

~test()

{

delete p;

std::cout<<"Destroyed";

}

};

int main()

{

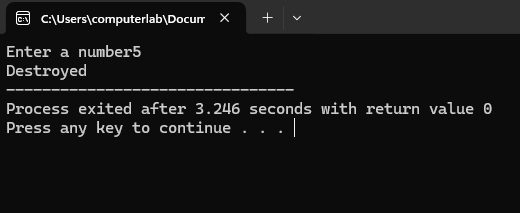
test t;

t.read();

getch();

}

**Output:**

****

**PRACTICAL NO: 06**

**Aim:** Program to demonstate use of public , protected & privete scope **specifiers**

**Code:**

#include<iostream>

#include<conio.h>

class Data

{

protected:

int a,b;

public:

void read()

{

std::cout<<"Enter two numbers";

std::cin>>a>>b;

}

};

class Sum:public Data

{

private:

int sum;

public:

void add()

{

sum=a+b;

}

void display()

{

std::cout<<"The sum is"<<sum;

}

};

int main()

{

Sum s;

s.read();

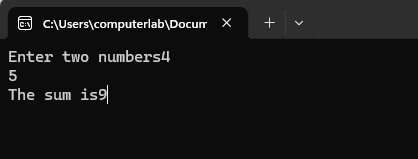
s.add();

s.display();

getch();

}

**Output:**

****

**PRACTICAL NO:7**

**Aim: Program to demonsate single and multilevel inheritance.**

**(A)**

**Code:**

#include<iostream>

#include<conio.h>

class Data

{

protected:

int r;

public:

void read()

{

std::cout<<"Enter the radius";

std::cin>>r;

}

};

class Area: public Data

{

private:

float result;

public:

void compute()

{

result=3.14\*r\*r;

}

void display()

{

std::cout<<"The area of the circle is"<<result;

}

};

int main()

{

Area a ;

a.read();

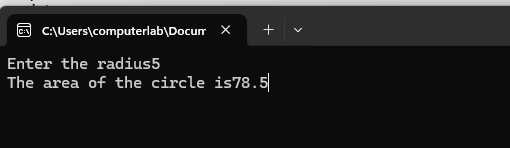
a.compute();

a.display();

getch();

}

**Output:**

****

**(B)**

**Code:**

#include<iostream>

#include<conio.h>

using namespace std;

class Data

{

protected:

int p,c,m;

public:

void read()

{

cout<<"Enter the marks obtained in Physics,Chemistry and Maths ";

cin>>p>>c>>m;

}

};

class Sum:public Data

{

protected:

int total;

public:

void sum()

{

total=p+c+m;

}

};

class Percent:public Sum

{

private:

float percent;

public:

void calculate()

{

percent=total/300.0\*100;

}

void display()

{

cout<<"The percentage is "<<percent;

}

};

int main()

{

Percent a;

a.read();

a.sum();

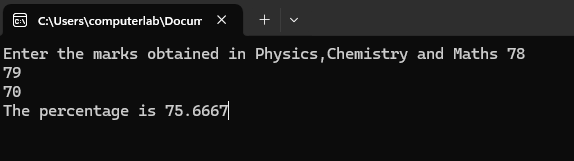
a.calculate();

a.display();

getch();

}

**Output:**

****

**PRACTICAL NO : 08**

**Aim:-** Programs to demonstrate multiple inheritance and hierarchical inheritance.

**Code:-**

#include<iostream.h>

#include<conio.h>

#include<stdio.h>

class Staff

{

protected:

char name[20];

int code:

};

class Teacher: public Staff

{

private:

char subject[20];

int experience:

public:

void read()

{

cout<<"Enter name, code, subject and experience of the teacher:":

gets(name);

cin>>code;

gets(subject);

cin>>experience;

}

void display()

{

cout<<"Teacher

Details:\nName:"<<name<<"\nCode:"<<code<<"\nSubject:"<<subject<<"\nExperience:"<<experience;

}

};

class Officer: public Staff

{

private:

char dept[20];

int grade;

public:

void read()

{

cout<<"Enter name, code, department and grade of the officer:";

gets(name);

cin>>code;

gets(dept):

cin>>grade:

}

void display()

{

cout<<"Officer

Details:\nName:"<<name<<"\nCode:"<<code<<"\nDepartment:"<<dept<<"\nGrade:"<<grade;

}

};

class Typist: public Staff

{

protected:

int speed,experience;

};

class Regular: public Typist

{

private:

int salary;

public:

void read()

{

cout<<"Enter name, code, speed, experience and salary of the regular typist:";

gets(name);

cin>>code>>speed>>experience>>salary;

}

void display()

{

cout<<"Regular Typist

Details:\nName:"<<name<<"\nCode:"<<code<<"\nSpeed:"<<speed<<"\nExperience:"<<experience<<"\nSalary:"<<salary:

}

};

class Casual: public Typist

{

private:

int dailywages;

public:

void read()

{

cout<<"Enter name, code, speed, experience and dailywages of the Casual typist:";

gets(name);

cin>>code>>speed>>experience>>dailywages;

}

void display()

{

cout<<"Casual Typist

Details:\nName:"<<name<<"\nCode:"<<code<<"\nSpeed:"<<speed<<"\nExperience:"<<experience<<<"\nDaily Wages:"<<dailywages;

}

};

void main()

{

clrscr();

int choice;

cout<<"1. Teacher\n2. Officer\n3. Regular Typist\n4. Casual Typist\nEnter the choice, whose details you want to enter:";

cin>>choice;

switch(choice)

{

case 1: Teacher t

tread();

t.display();

break;

case 2:Officer o;

o.read();

o.display():

break;

case 3: Regular r;

r.read();

r.display();

break;

case 4:Casual c;

c.read);

c.display();

break;

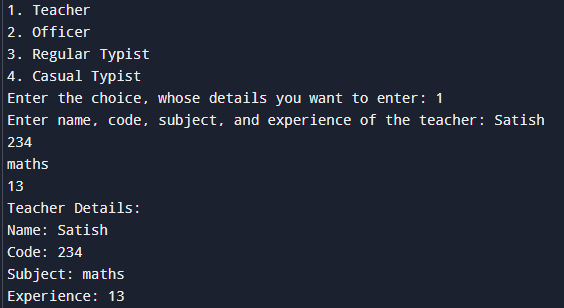
default:cout<<"Invalid choice";

}

getch();

}

**Output:-**



**PRACTICAL NO : 09**

**Aim:-**Programs to demonstrate inheritance and derived class constructors.

**Code:-**

#include<iostream.h>

#include<conio.h>

#include<stdio.h>

class Student

{

protected:

char name[20];

int roll no:

};

class Test: public Student

{

protected:

int marks:

public:

void read()

{

cout<<"Enter name, roll number and marks obtained";

gets(name);

cin>>roll\_no>>marks;

}

};

class Sports

{

protected:

int score;

public:

void accept()

{

cout<<"1. Student has won in national sports event\n2. Student has not won in any national sports event\nEnter your choice:";

cin>>score;

}

};

class Result: public Test, publie Sports

{

int total:

public:

void calculate()

{

if(score==1)

total-marks+15;

else

total=marks;

}

void display()

{

cout<<"The total is "<<total;

}

};

void main(

{

clrscr;

Result r;

r.read();

r.accept();

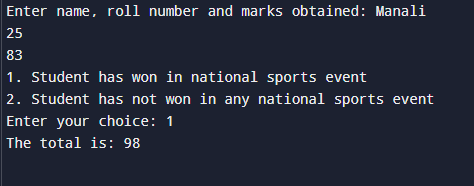
r.calculate();

r.display();

getch();

}

**Output:-**



**PRACTICAL NO : 10**

**Aim:-**Programs to demonstrate Friend function, inline function, this pointer.

**A)**

**Code:-**

#include<iostream.h>

#include<conio.h>

float add(float a, int b)

{

float c;

c=a+b;

return c;

}

int add(int a, int b)

{

int c;

c = a + b;

return c;

}

float add(float a, float b)

{

float c;

c = a + b;

return c;

}

void main()

{

clrscr;

int x,a=5,b=6;

float y, p = 3.5 q = 6.6;

x = add(a, b);

cout<<"Sum="<<x<<endl;

y = add(p, q);

cout<<"Sum="<<y<<endl;

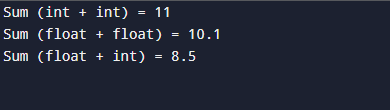
y = add(p, a);

cout<<"Sum="<<y<<endl;

getch();

}

**Output:-**



**B)**

**Code:-**

#include<iostream.h>

#include<conio.h>

class Base

{

protected:

int a,b;

public:

void read()

{

cout<<"Enter two values:";

cin>>a>>b;

}

void display()

{

cout<<"The values are:"<<a<<"\n"<<b;

}

};

class Sub: public Base

{

protected:

int c,d;

public:

void read()

{

cout<<"Enter 4 values:";

cin>>a>>b>>c>>d;

}

void display

{

cout<<"The values are:"<<a<<"\n"<<b<<"\n"<<c<<"\n"<< d;

}

};

void main()

{

clrscr();

Sub s;

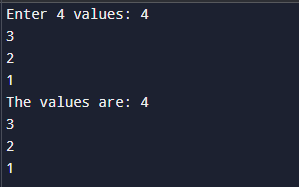
s.read();

s.display();

getch();

}

**Output:-**



**C)**

**Code:-**