8.1) Develop a C++ program to illustrate object as a class member.

Program :

#include<iostream>

using namespace std;

class DOB

{

private:

int dd,mm,yy;

public:

void show()

{

cout<<"enter day ,month and year:";

cin>>dd>>mm>>yy;

cout<<dd<<"-"<<mm<<"-"<<yy;

}

};

class student

{

private:

string name;

int pinno;

DOB d;

public:

void display()

{

cout<<" enter name :";

cin>>name;

cout<<" enter pinno:";

cin>>pinno;

cout<<" name "<<name<<endl;

cout<<" pinno"<<pinno<<endl;

d.show();

}

};

int main()

{

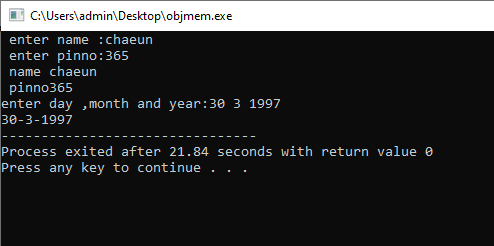
student s;

s.display();

return 0;

}

Output:



9.2) Develop a C++ program to illustrate runtime polymorphism

Program:

#include<iostream>

using namespace std;

class Base

{

public:

virtual void show()

{

cout<<" show() of base class"<<endl;

}

};

class derived : public Base

{

public:

void show()

{

cout<<"show() of derived class "<<endl;

}

};

int main()

{

Base b,\*bptr;

bptr=&b;

bptr->show();

derived d,\* dptr;

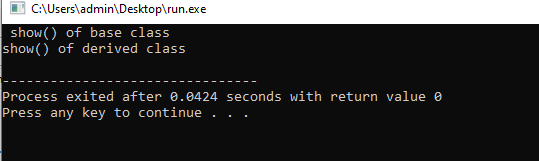
bptr=&d;

bptr->show();

return 0;

}

Output:



9.3) Develop a C++ program to illustrate pure virtual function and calculate the area of different shapes by using abstract class.

Program:

#include<iostream>

using namespace std;

class shapes

{

public:

virtual void area()=0;

};

class square: public shapes

{

public:

int a;

square()

{

cout<<" enter a:";

cin>>a;

}

void area()

{

cout<<" area of square"<<a\*a<<endl;

}

};

class circle:public shapes

{

public:

int r;

circle()

{

cout<<" enter r:";

cin>>r;

}

void area()

{

cout<<" area of circle:"<<3.14\*r\*r<<endl;

}

};

class rectangle: public shapes

{

public:

int l,b;

rectangle()

{

cout<<" enter length and breadth:";

cin>>l>>b;

}

void area()

{

cout<<" area of rectangle:"<<l\*b<<endl;

}

};

int main()

{

square s,\*sptr;

sptr=&s;

sptr->area() ;

circle c,\*cptr;

cptr=&c;

cptr->area();

rectangle r,\*rptr;

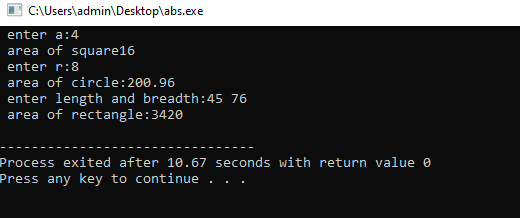
rptr=&r;

rptr->area();

return 0;

}

output:



8.2) Develop a C++ program to illustrate pointer to a class

Program:

#include<iostream>

using namespace std;

class Base

{

public:

void show()

{

cout<<" show() of base class"<<endl;

}

};

class derived : public Base

{

public:

void show()

{

cout<<"show() of derived class "<<endl;

}

};

int main()

{

Base b,\*bptr;

bptr=&b;

bptr->show();

derived d,\* dptr;

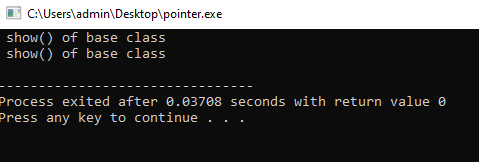
dptr=&d;

dptr->show();

return 0;

}

Output:



9.1) Develop a C++ program to illustrate virtual functions

Program: