#include<iostream>

using namespace std;

class student

{

protected:

string branch;

string name;

char gender;

void getst()

{

cout<<"enter name ";

cin>>name;

cout<<"enter branch ";

cin>>branch;

cout<<" enter gender:";

cin>>gender;

}

};

class marks : public student

{

protected:

int m1,m2,m3,m4,m5,m6;

int pinno;

public:

void getm()

{

getst();

cout<<"enter marks:";

cin>>m1>>m2>>m3>>m4>>m5>>m6;

cout<<" enter pinno:";

cin>>pinno;

}

};

class project

{

protected:

int ep,ip;

void getp()

{

cout<<" enter ip &ep :";

cin>>ip>>ep;

}

};

class percentage : public marks ,public project

{

private:

float per;

public:

void getper()

{

getm();

getp();

per=(float)( m1+m2+m3+m4+m5+m6+ip+ep)/8;

}

void show()

{

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

cout<<" branch is "<< branch<<endl;

cout<<" gender is "<<gender<<endl;

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

cout<<"percentege is"<<per<<endl;

}

};

int main()

{

percentage p;

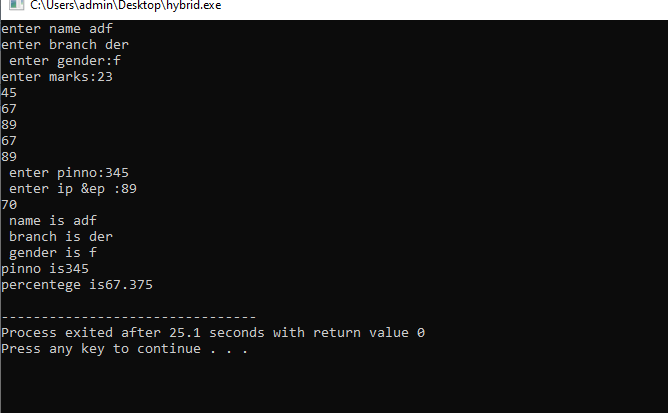
p.getper();

p.show();

return 0;

}

Output:



7.2) Develop a C++ program in C++ to illustrate the order of execution of constructors and destructors in inheritance.

#include<iostream>

using namespace std;

class A

{

protected:

A()

{

cout<<" a constructor"<<endl;

}

~A()

{

cout<<" a destructor"<<endl;

}

};

class B: public A

{

public:

B()

{

cout<<" b constructor"<<endl;

}

~B()

{

cout<<" b destructor"<<endl;

}

};

class C: public A

{

public:

C()

{

cout<<" c constructor"<<endl;

}

~C()

{

cout<<"c destructor"<<endl;

}

};

class D : public B ,public C

{

public:

D()

{

cout<< " d constructor"<<endl;

}

~D()

{

cout<<" d destructor"<<endl;

}

};

int main()

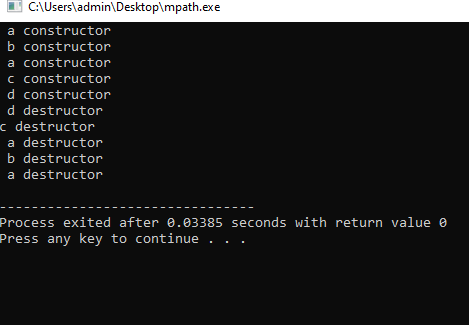
{

D d;

return 0;

}

Output:



8.3) Develop a C++ program to illustrate Virtual Base Class.

#include<iostream>

using namespace std;

class A

{

public:

void showA()

{

cout<<" method of class A"<<endl;

}

};

class B: virtual public A

{

public:

void showB()

{

cout<<"method of class B"<<endl;

}

};

class C :virtual public A

{

public:

void showC()

{

cout<<" method of class C"<<endl;

}

};

class D:public B,public C

{

public:

void showD()

{

cout<<"method of class D"<<endl;

}

};

int main()

{

D d;

d.showA();

d.showB();

d.showC();

d.showD();

return 0;

}

Output:

