LAB 1 AFTER MIDS

TASK REPORT

Task 1 :

When we compile the code without using virtual keyword ,on the console screen the output of the function in base class is “Base class “ and output of function of derived class is also” Base class”,where as when we add the keyword virtual before the testfunction() in the derived class the output is” Base class“for class Base function and ,”Derived Class” for class Derived Function .

Code After Changes :

#include<iostream >

using namespace std;

class Base {

public :

virtual void testfunction ();

};

class Derived:public Base {

public:

void testfunction ();

};

void Base :: testfunction ()

{

cout <<"Base Class "<<endl;

}

void Derived :: testfunction ()

{

cout<<"Derived Class "<<endl;

}

int main (void)

{

Base \*ptr = new Base;

ptr -> testfunction ();

delete ptr ;

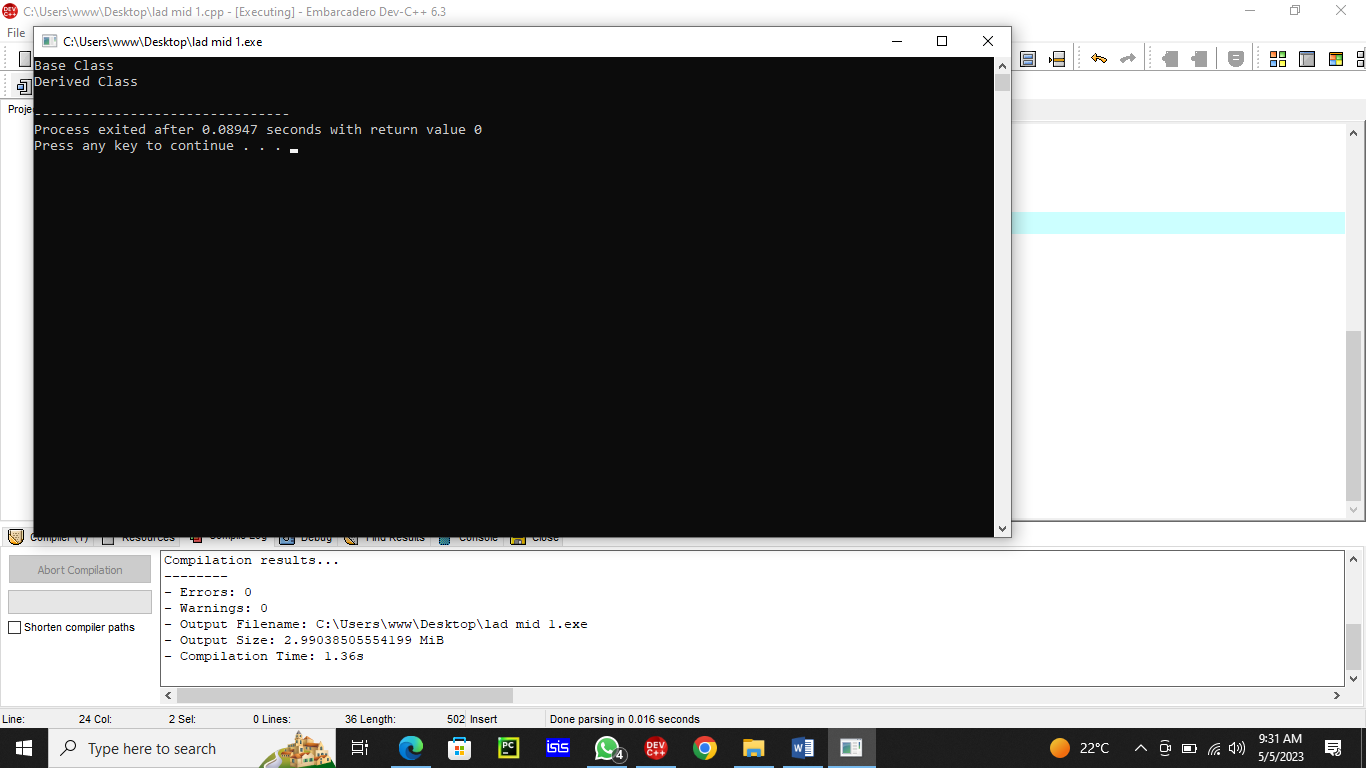
ptr= new Derived ;

ptr->testfunction() ;

delete ptr ;

return 0;

}



Task 2:

. Error generated as inheritance of functions shuld be done in class Dog from class Mammal.

.After proper inheritance we get the Mammal class output not the dog class for that we will make another pointer.

. now when we execute we get the correct output .

Code :

#include <iostream >

using namespace std;

class Mammal {

public:

Mammal(void );

~Mammal(void );

void Move() const;

void Speak() const;

protected :

int itsAge ;

};

class Dog :public Mammal{

public:

virtual void Move() const;

virtual void Speak() const;

};

Mammal::Mammal(void):itsAge(1)

{

cout<<"Mammal constructor..."<<endl;

}

Mammal ::~Mammal(void)

{

cout <<"Mammal destructor ..."<<endl;

}

void Mammal ::Move ()const

{

cout<<"Mammal moves a step! "<<endl;

}

void Mammal ::Speak ()const

{

cout <<"What does a mammal speak ? Mammilian!"<<endl;

}

// Dog

void Dog ::Move ()const

{

cout<<"Dog moves a step! "<<endl;

}

void Dog ::Speak ()const

{

cout <<"What does a Dog speak ? Mammilian!"<<endl;

}

int main ()

{

Mammal \*pDog=new Dog;

pDog -> Move ();

pDog->Speak();

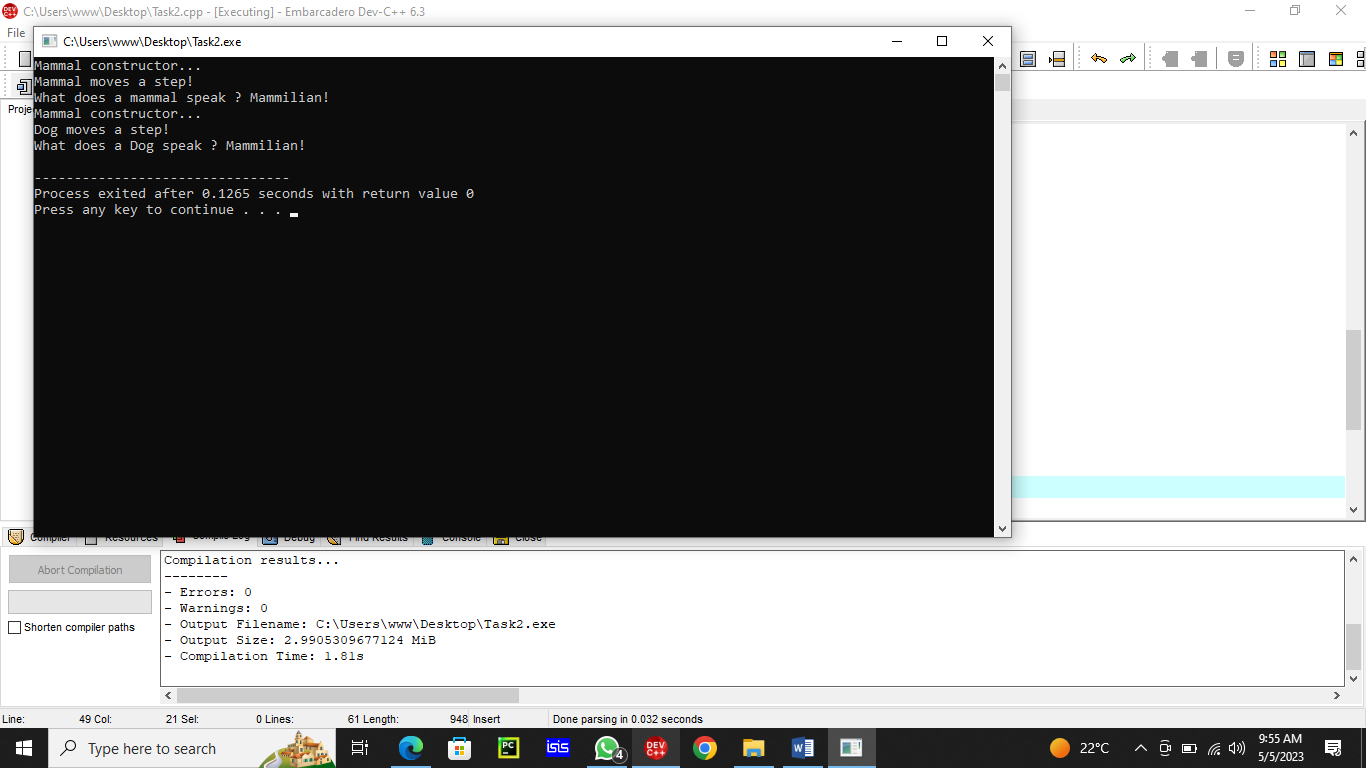
Dog \*pDog2 =new Dog;

pDog2->Move ();

pDog2->Speak();

return 0 ;

}



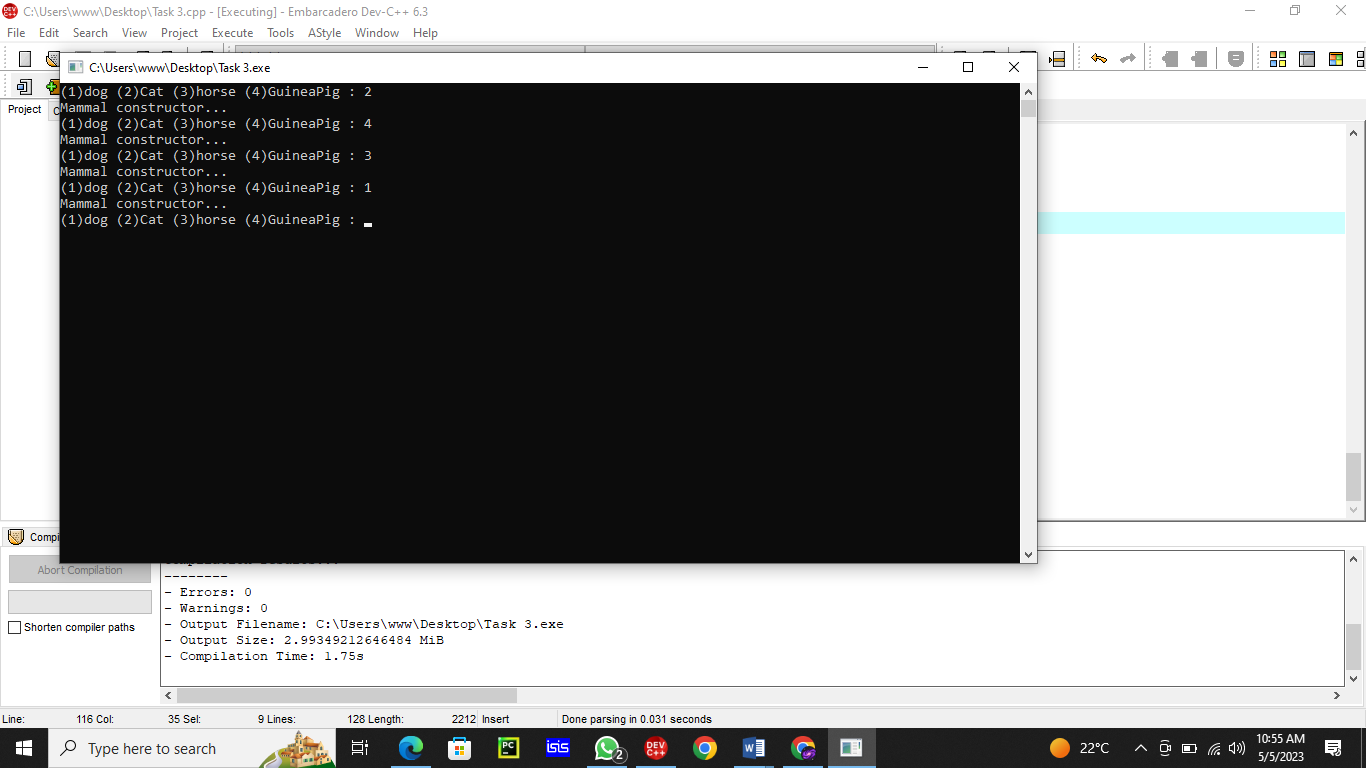
.Task 3

.Cat ,Horse , GuineaPig classes were added and so they were inheriting mammal class properties .

And so there functions were defined .

There pointers were made and so virtual functions were called .

The output calls the mammal constructor



Some questions that you should start to understand:

Are inherited members and functions passed along to subsequent generations?

Yes

If Dog derives from Mammal, and Mammal derives from Animal, does Dog inherit Animal's functions and data?

Ans Yes

Q. If, in the example above, Mammal overrides a function in Animal, which does Dog get, the original or the overridden function?

Yes ,mammal has the overridden function

Q. Can a derived class make a public base function private?

Yes

Q. Why not make all class functions virtual?

Yes

if one function is virtual all others are virtual too .

Q. If a function (SomeFunc()) is virtual in a base class and is also overloaded, so as to take either an integer or two integers, and the derived class overrides the form taking one integer, what is called when a pointer to a derived object calls the two-integer form?

Ans function requires only one int