Institute of Computer Technology

B. Tech Computer Science and Engineering

Subject: ESFP-II (2CSE203)

**PRACTICAL-8**

**AIM: - To learn about virtual function & Friend Function in C++.**

***Exercise:***

**1. Crate an animal class and also create another 2 child classes of animal class called cat & dog. All the defined classes has one common method named gettype(). Print the animal type as per class name using virtual functions.**

***CODE:***

#include <iostream>

using namespace std;

class Animal

{

public:

virtual void gettype()

{

cout<<"This is ANIMALS class."<<endl;

}

};

class Cat:public Animal

{

public:

void gettype() override

{

cout<<"I am Cat."<<endl;

}

};

class Dog:public Animal

{

public:

void gettype() override

{

cout<<"I am Dog."<<endl;

}

};

int main()

{

Animal A;

A.gettype();

Cat C;

C.gettype();

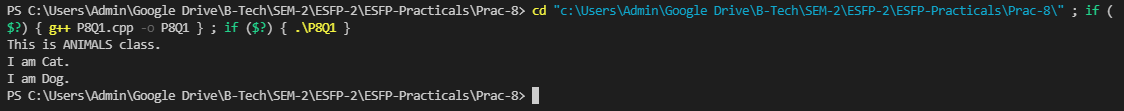
Dog D;

D.gettype();

return 0;

}

***OUTPUT:***

******

**2. Create a program to calculate the area of a square and a circle using Abstract class & Pure virtual Function.**

***CODE:***

#include <iostream>

using namespace std;

class Shape

{

public:

virtual void CalcArea()=0;

};

class SquareCircle:public Shape

{

public:

int l,r;

void CalcArea() override

{

float C\_Area,S\_Area;

cout<<"\nEnter side of Square: ";

cin>>l;

cout<<"\nEnter radius of Circle: ";

cin>>r;

C\_Area=3.14\*r\*r;

S\_Area=l\*l;

cout<<"\nArea of Circle: "<<C\_Area;

cout<<"\nArea of Square: "<<S\_Area;

}

};

int main()

{

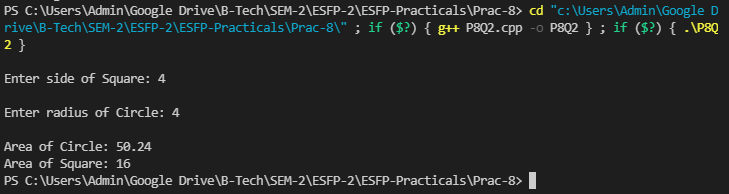
SquareCircle SC;

SC.CalcArea();

return 0;

}

***OUTPUT:***

******

**3. Create 2 classes A and B with common private data member function. How can we call that Private data member function of both classes from outside the class?**

***CODE:***

#include <iostream>

using namespace std;

class A {

private:

void FunA()

{

cout<<"\nPrivate Member of class A.";

}

friend void getData();

};

class B {

private:

void FunB()

{

cout<<"\nPrivate Member of class B.";

}

friend void getData();

};

void getData() {

A Aobj;

Aobj.FunA();

B Bobj;

Bobj.FunB();

}

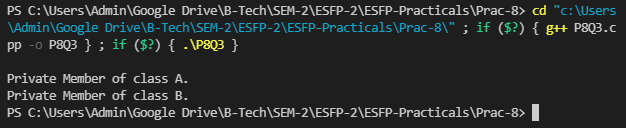
int main() {

getData();

return 0;

}

***OUTPUT:***

****

**Post Practical Work**

**1. Demonstrate the use of virtual destructor using appropriate C++ code.**

***CODE:***

#include<iostream>

using namespace std;

class Yash

{

public:

Yash()

{

cout << "\nConstructor Called";

}

virtual ~Yash()

{

cout << "\nDestructor Called";

}

};

class Derived: public Yash

{

public:

Derived()

{

cout << "\nDerived Constructor called." ;

}

~Derived()

{

cout << "\nDerived Destructor called.";

}

};

int main()

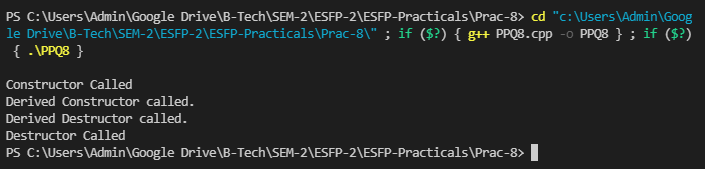
{

Yash \*p = new Derived;

delete p;

}

***OUTPUT:***

******

**2. What is the output of following C++ program?**

**class Base{**

**public:**

**void f(){**

**cout<<"Base::f()"<<endl;**

**}**

**};**

**class Derived:public Base{**

**public:**

**void f(){**

**cout<<"Derived::f()"<<endl; } }; int main(){ Base \*d = new Derived(); d->f();**

**return 0;**

**}**

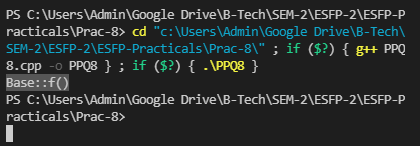
**A. Base::f()**

B. Derived::f()

C. Base::f() Derived::f()

D. Compiler error

***OUTPUT:***

******

**3. What is output of the following C++ program?**

**class A{**

**public:**

**void f(){**

**cout<<"A::f()"<<endl;**

**}**

**};**

**class B:public A{**

**public:**

**void fb(){**

**cout<<"A::fb()"<<endl;**

**}**

**};**

**class C:public A{**

**public:**

**void fc(){**

**cout<<"A::fc()"<<endl;**

**}**

**};**

**class D: public B,public C{**

**public:**

**void fd(){**

**cout<<"A::fd()"<<endl;**

**}**

**};**

**int main(){**

**D obj;**

**obj.f();**

**return 0;**

**}**

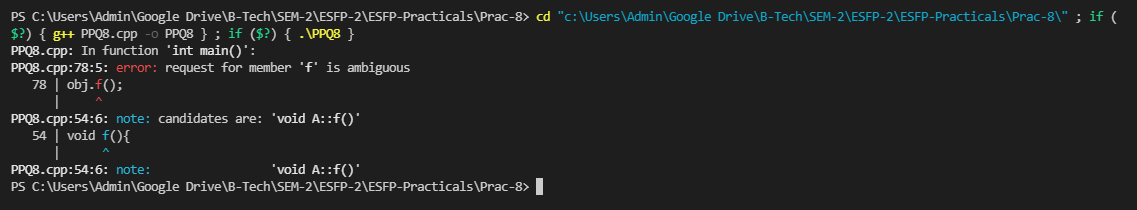
A. A::f()

B. A::f() A::f()

C. A::f() A::f()

**D. Compiler error**

***OUTPUT:***

******