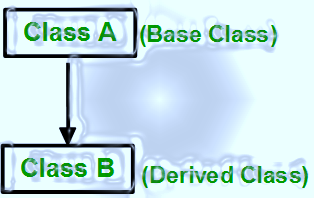
**SDF**

**Tutorial 12**

1. List and define the types of inheritance supported in C++.

**Answer:**

1. **Single Inheritance**: In single inheritance, a class is allowed to inherit from only one class. i.e. one sub class is inherited by one base class only.

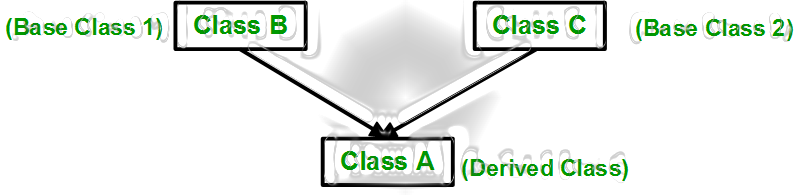
**Syntax**:

class subclass\_name : access\_mode base\_class

{

//body of subclass

};

1. **Multiple Inheritance:** Multiple Inheritance is a feature of C++ where a class can inherit from more than one classes. i.e one **sub class** is inherited from more than one **base classes**.

**Syntax**:

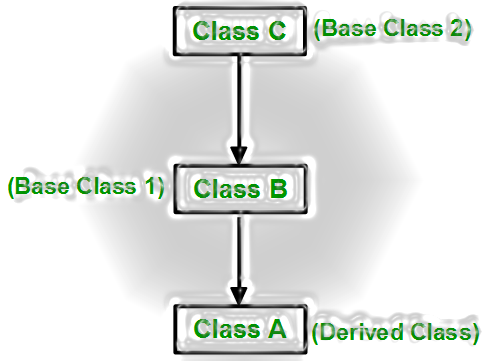
class subclass\_name : access\_mode base\_class1, access\_mode base\_class2, ....

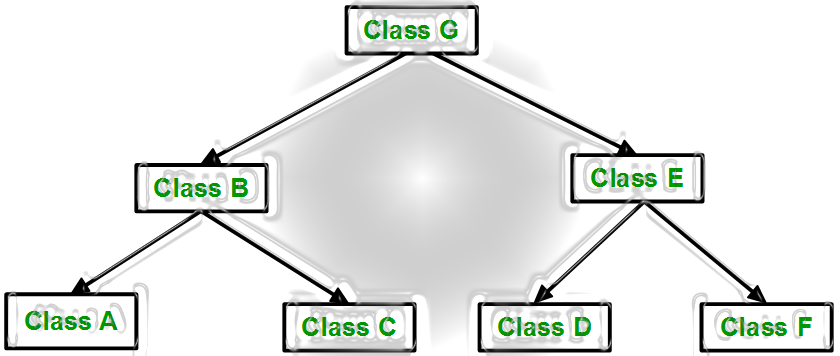
{

//body of subclass

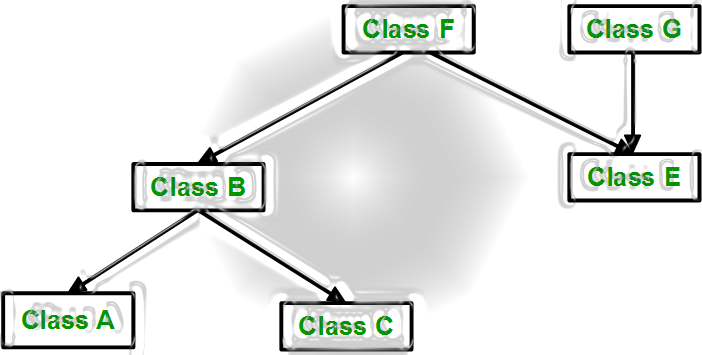
};

Here, the number of base classes will be separated by a comma (‘, ‘) and access mode for every base class must be specified.

1. **Multilevel Inheritance**: In this type of inheritance, a derived class is created from another derived class.

**4.Hierarchical Inheritance**: In this type of inheritance, more than one sub class is inherited from a single base class. i.e. more than one derived class is created from a single base class.

**5.Hybrid (Virtual) Inheritance**: Hybrid Inheritance is implemented by combining more than one type of inheritance. For example: Combining Hierarchical inheritance and Multiple Inheritance.  
Below image shows the combination of hierarchical and multiple inheritance:

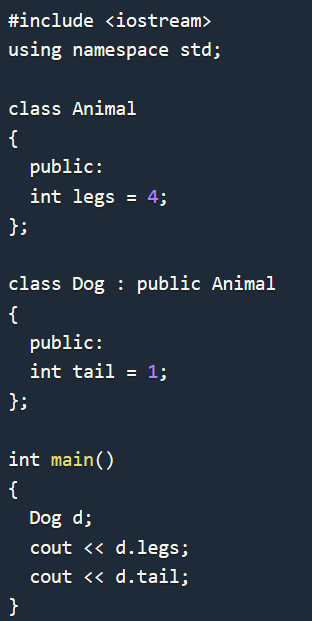


**Question 2: For below code snippet, the public and protected members of Superclass becomes \_\_\_\_\_\_\_\_ members of Sub class.**

**class subclass : protected Superclass**

**Answer: Protected**

**Question 3:** What will be the output of following code ?



**Answer:** 41

**Question 4:** Do base class and its object have any knowledge about any classes derived from base class?

**Answer: No! Base class and its object do not have any knowledge about any classes derived from base class.**

**Question 5:** Whenever you create derived class object, first the base class default constructor is executed and then the derived class constructor?

**Answer**: Yes! Whenever we create derived class object , first the base class default constructor is executed and then the derived class constructor.

**Question 6:**Write a C++ program to display the skills of a person according to his/her profession using inheritance.

**Code:**

#include<iostream>

using namespace std;

class profession

{

public:

profession()

{

cout<<"His/Her Profession:";

}

};

class Artist : public profession

{

public:

Artist():profession()

{

cout<<"Artist\nSkills : Persistence,Patience,Passion,A sense of adventure and Discipline.";

}

};

class Dancer : public profession

{

public:

Dancer():profession()

{

cout<<"Dancer\nSkills : Goal-directed actions that are observable as small units of engagement in daily life occupations";

}

};

class Engineer : public profession

{

public:

Engineer():profession()

{

cout<<"Enginner\nSkills : Critical thinking,communication,project and time management";

}

};

class Doctor : public profession

{

public:

Doctor():profession()

{

cout<<"Doctor\nSkills : Compassion,Understanding,Empathy,Honesty,Competence,Commitment,Humanity and Courage";

}

};

int main()

{

int ch;

cout<<"Choices of the professions:-"<<endl;

cout<<"1.Engineer"<<endl;

cout<<"2.Doctor"<<endl;

cout<<"3.Artist"<<endl;

cout<<"4.Dancer"<<endl;

cout<<"5.Exit";

while(1)

{

cout<<endl<<endl<<"Enter your choice:";

cin>>ch;

if(ch==1)

Engineer e;

else if(ch==2)

Doctor d;

else if(ch==3)

Artist a;

else if(ch==4)

Dancer d;

else if(ch==5)

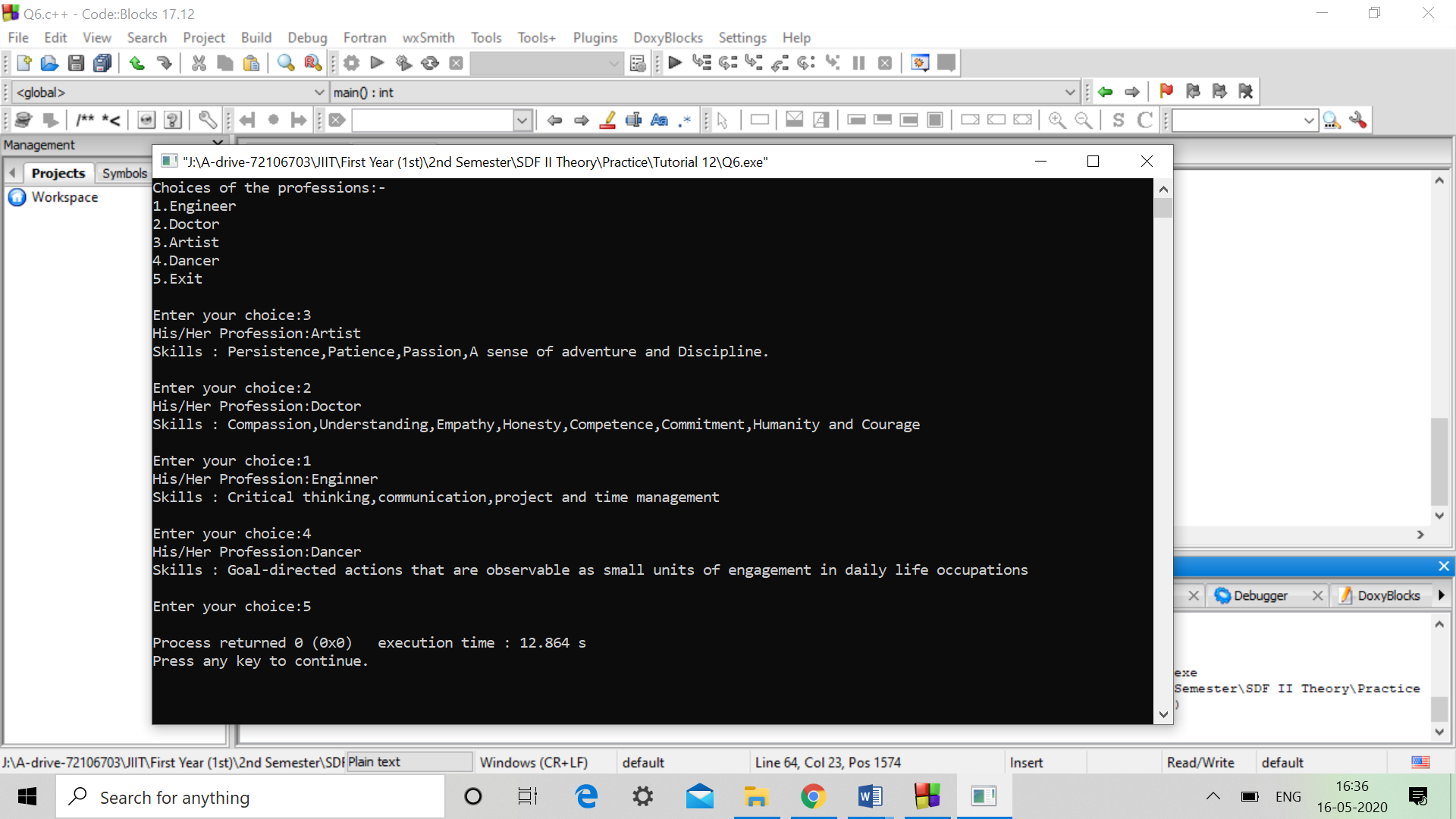
break;

}

return 0;

}

**Output:**

****

# Question 7:Write a C++ program to read and print employee information using multiple inheritance.

**Code:**

#include<iostream>

#include<cstdio>

using namespace std;

class a1

{

private:

string name,address;

protected:

void get()

{

cout<<"Enter Name: "<<endl;

fflush(stdin);

getline(cin,name);

cout<<"Enter Address: "<<endl;

fflush(stdin);

getline(cin,address);

}

void show()

{

cout<<"Name :"<<name<<endl;

cout<<"Address : "<<address<<endl;

}

};

class a2

{

private:

string occ;

int salary;

protected:

void get()

{

cout<<"Enter Occupation:"<<endl;

fflush(stdin);

getline(cin,occ);

cout<<"Enter salary: "<<endl;

fflush(stdin);

cin>>salary;

}

void show()

{

cout<<"Occupation :"<<occ<<endl;

cout<<"Salary: "<<salary<<endl;

}

};

class b:public a1,public a2

{

public:

b()

{

a1::get();

a2::get();

}

void showdata()

{

a1::show();

a2::show();

}

};

int main()

{

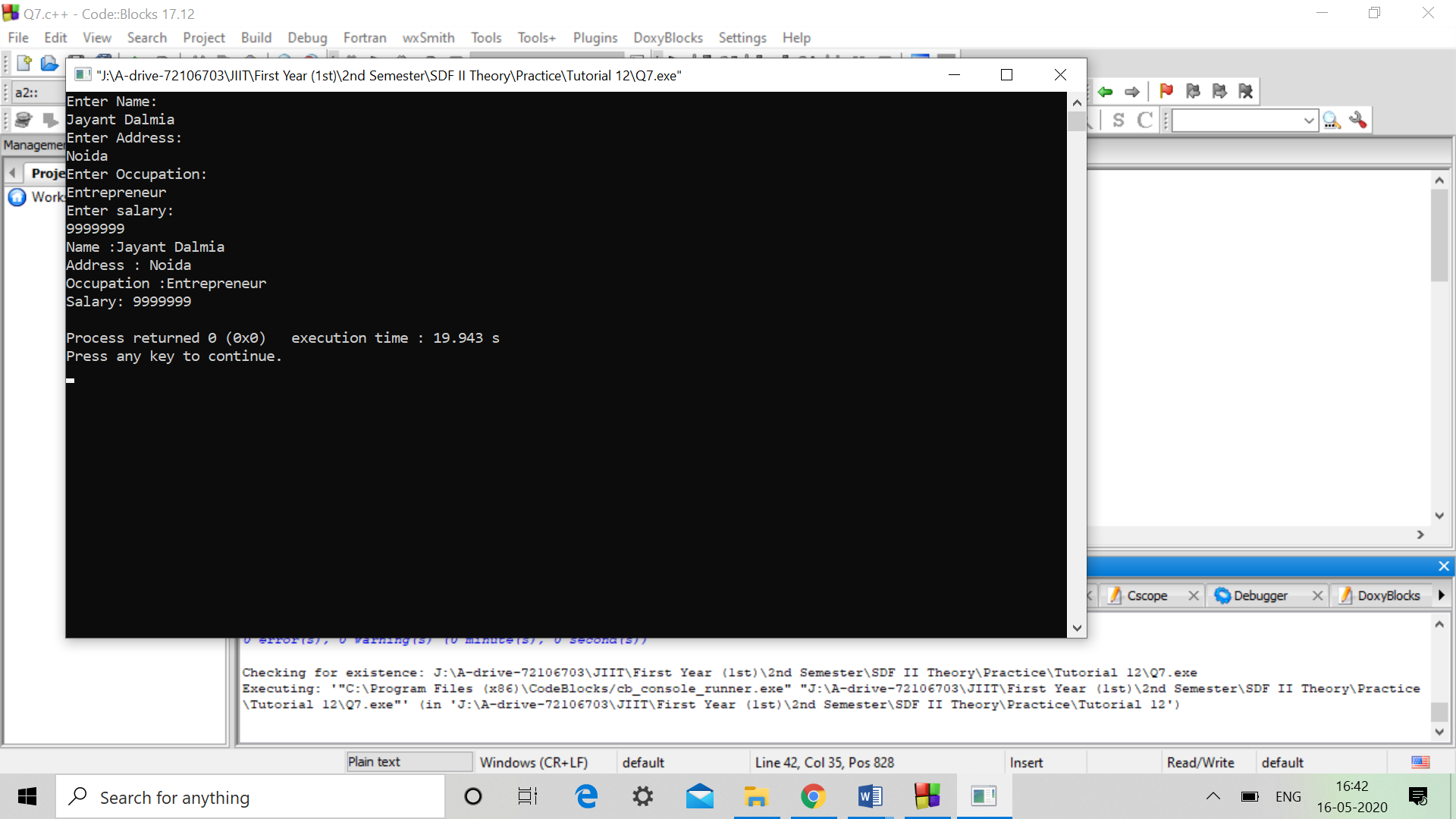
b emp;

emp.showdata();

return 0;

}

**Output:**

****

# Question 8: Write a C++ program to calculate cube, root and square of a number using hierarchical inheritance.

**Code:**

#include<iostream>

#include<math.h>

using namespace std;

class operation

{

protected:

int n;

public:

void setdata()

{

cout<<"\nEnter the value:";

cin>>n;

}

};

class cube:public operation

{

public:

void getdata()

{

cout<<"Cube:"<<pow(n,3)<<endl;

}

};

class root:public operation

{

public:

void getdata()

{

cout<<"Root:"<<pow(n,0.5)<<endl;

}

};

class square:public operation

{

public:

void getdata()

{

cout<<"Square:"<<pow(n,2)<<endl;

}

};

int main()

{

int ch;

cout<<"Choices:-\n";

cout<<"1.Cube\n";

cout<<"2.Root\n";

cout<<"3.Square\n";

cout<<"4.Exit\n";

while(1)

{

cout<<"\nEnter Your choice:";

cin>>ch;

if(ch==1)

{

cube c;

c.setdata();

c.getdata();

}

else if(ch==2)

{

root r;

r.setdata();

r.getdata();

}

else if(ch==3)

{

square s;

s.setdata();

s.getdata();

}

else if(ch==4)

break;

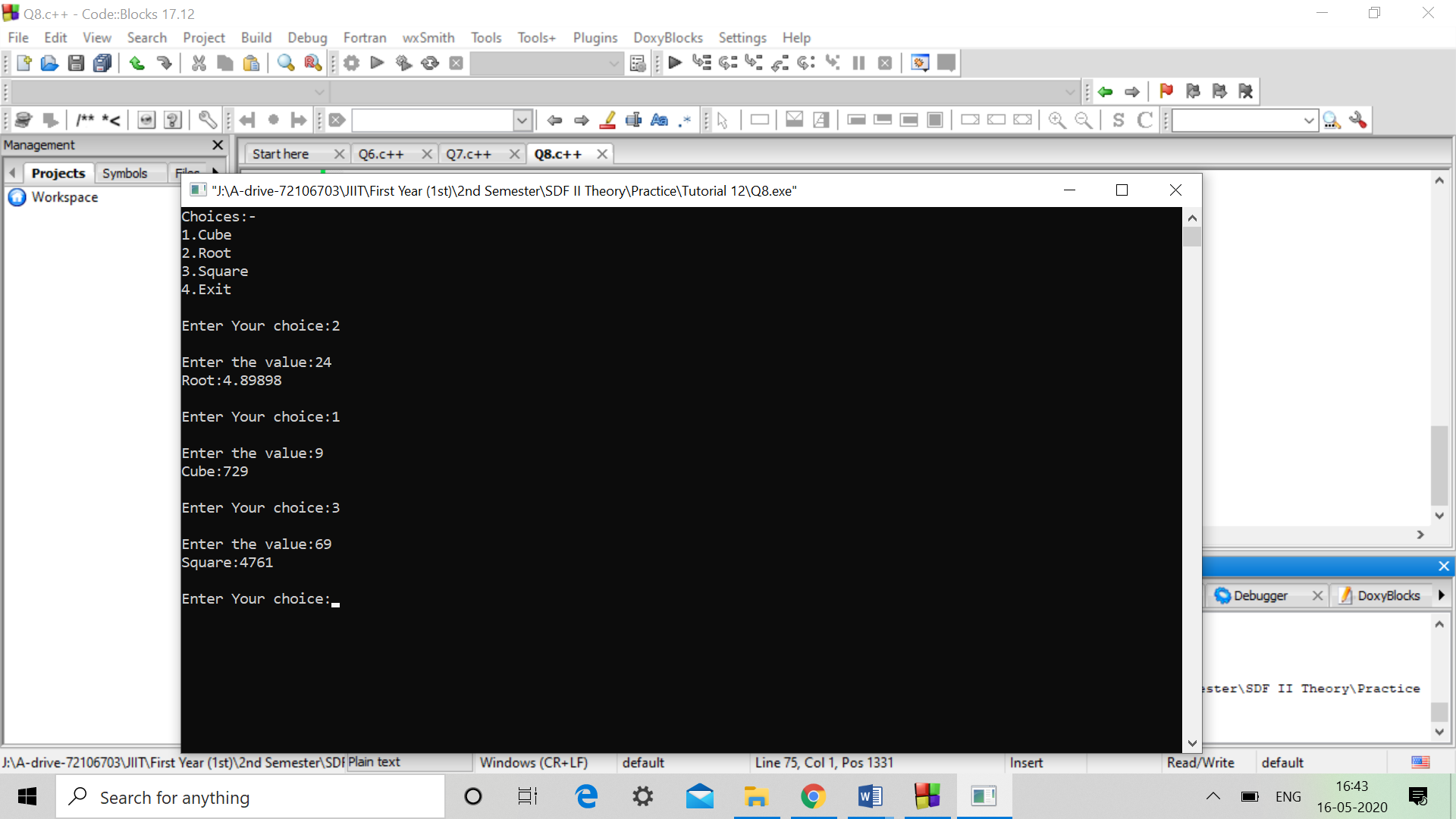
}

cout<<"Exit!!!";

return 0;

}

**Output:**

****

**Question 9:** Create two classes named Mammals and MarineAnimals. Create another class named   
BlueWhale which inherits both the above classes. Now, create a function in each of   
these classes which prints "I am mammal", "I am a marine animal" and "I belong to   
both the categories: Mammals as well as Marine Animals" respectively. Now, create   
an object for each of the above class and try calling  
1 - function of Mammals by the object of Mammal  
2 - function of MarineAnimal by the object of MarineAnimal  
3 - function of BlueWhale by the object of BlueWhale  
4 - function of each of its parent by the object of BlueWhale

**Code:**

#include<iostream>

using namespace std;

class Mammals

{

public:

void disp1()

{

cout<<"I am mammal\n";

}

};

class MarineAnimal

{

public:

void disp2()

{

cout<<"I am a marine animal\n";

}

};

class BlueWhale:public Mammals,public MarineAnimal

{

public:

void disp3()

{

cout<<"I belong to both the categories: Mammals as well as Marine Animals\n";

}

};

int main()

{

Mammals Mammal;

MarineAnimal MarineAnimal;

BlueWhale BlueWhale;

cout<<"Calling function of Mammals by the object of Mammal:-\n";

Mammal.disp1();

cout<<"\nCalling function of MarineAnimal by the object of MarineAnimal:-\n";

MarineAnimal.disp2();

cout<<"\nCalling function of BlueWhale by the object of BlueWhale:-\n";

BlueWhale.disp3();

cout<<"\nCalling function of each of its parent by the object of BlueWhale:-\n";

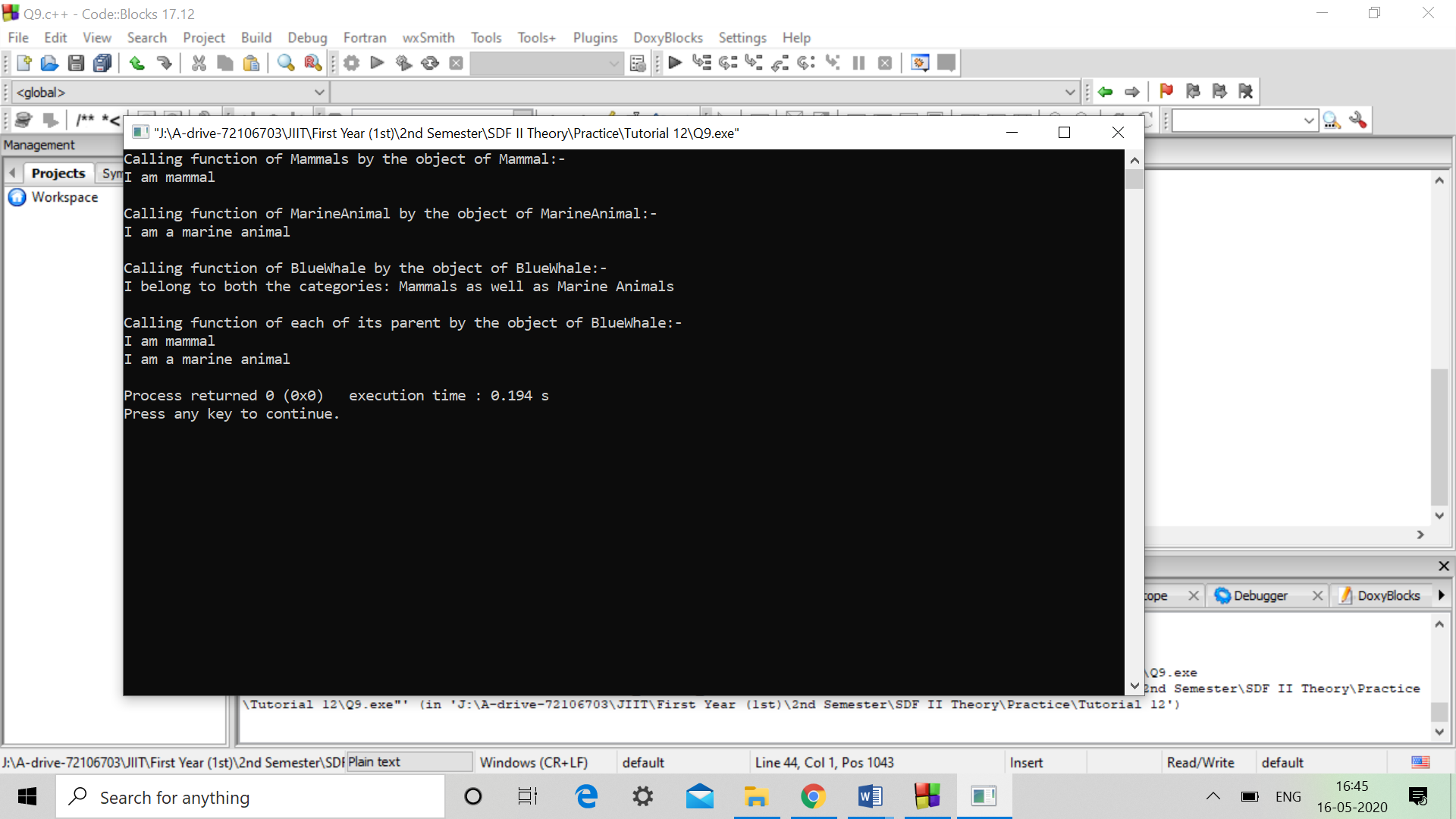
BlueWhale.disp1();

BlueWhale.disp2();

return 0;

}

**Output:**

****