**QUE1.**

**Code:**

*//You are provided with two libraries. Both libraries have a class called LibraryClass and both have a function called show(). Your task is to utilize both of these classes in a main program without any naming conflicts.*

#include <iostream>

using **namespace** std;

**namespace** libraryX

{

**class** LibraryClass

  {

**public:**

**void** show()

   {

     cout<< "show() FNX : LIBRARY X" <<endl;

    }

  };

};

**namespace** libraryY

{

**class** LibraryClass

  {

**public:**

**void** show()

   {

      cout<< "show() FNX : LIBRARY Y" <<endl;

    }

  };

};

**int** main()

{

  libraryX::LibraryClass X;

  libraryY::LibraryClass Y;

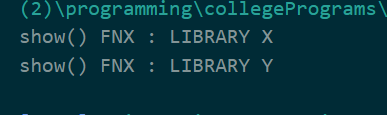
  X.show();

  Y.show();

        return 0;

}

**O/P:**

****

**QUE2.**

**Code:**

*//You are developing software for a robotic company. The company manufactures ground robots and drones. Each type of robot has different components. Your task is to define nested namespaces that categorize these robots and their components.*

#include <iostream>

using **namespace** std;

**namespace** ground\_robots

{

**void** fnx()

  {

    cout<< "Inside  robots" <<endl;

  }

**namespace** robots\_components

  {

**class** part

    {

**public:**

**void** fnx()

      {

        cout<< "Inside the parts of robots" <<endl;

      }

          };

  }

}

**namespace** drones

{

**void** fnx()

  {

    cout<< "Inside drones" <<endl;

  }

**namespace** drones\_components

  {

**class** part

          {

**public:**

**void** fnx()

      {

        cout<< "Inside the parts of drones" <<endl;

      }

    };

  }

}

**int** main()

{

  ground\_robots::robots\_components::part obj;

  obj.fnx();

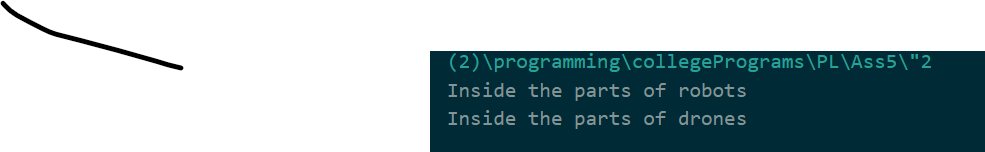
  drones::drones\_components::part obj1;

  obj1.fnx();

  return 0;

}

**O/P:**

****

**QUE3.**

**Code:**

*//You have to design a basic system where different animals make sounds. The base class Animal will have a function makeSound(), which should be overridden by derived classes to make specific sounds.*

#include<iostream>

using **namespace** std;

**class** Animal

{

**public:**

**virtual** **void** makeSound(){

      cout<<"WE ARE ABLE TO CREATE VARIOUS SOUND OF ANIMAL"<<endl;

    }

};

**class** CAT:**public** Animal{

**public:**

**void** makeSound() **override**{

      cout<<"CAT SOUND IS MEOW"<<endl;

   }

};

**class** HORSE:**public** Animal{

**public:**

**void** makeSound() **override**{

     cout<<"HORSE SOUND IS KNEE"<<endl;

   }

};

**class** GORILLA:**public** Animal{

**public:**

**void** makeSound() **override**{

     cout<<"GORILLA SOUND IS GRUNT"<<endl;

   }

};

**int** main(){

 Animal\* gorilla\_s=new GORILLA();

 gorilla\_s->makeSound();

 delete gorilla\_s;

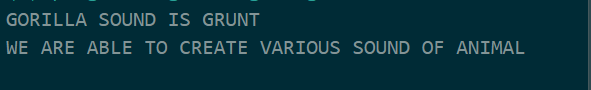
 Animal x;

 x.makeSound();

 return 0;

}

**O/P:**



**QUE4.**

**Code:**

*//. You are to design a system where different shapes calculate their areas. The base class Shape will have a function area(), which should be overridden by derived classes to calculate specific areas.*

#include <iostream>

#include <cmath>

using **namespace** std;

**class** shape

{

**public:**

**virtual** **void** area(**int** data)

    {

        cout<< "We can calculate Area of any shape" <<endl;

    }

};

**class** circle : **public** shape

{

**public:**

**void** area(**int** data) **override**

    {

        cout<<endl;

        cout<< "CIRCLE AREA : pi\*radius\*radius = " <<round(data\*data\*M\_PI)<<endl;

        cout<<endl;

    }

};

**class** square : **public** shape

{

**public:**

**void** area(**int** data) **override**

    {

        cout<<endl;

        cout<< "SQUARE AREA : side \* side = "<<round(data\*data)<<endl;

        cout<<endl;

    }

};

**int** main()

{

  shape\*circle\_a=new circle();

  circle\_a->area(5);

  delete circle\_a;

  shape \*square1=new square();

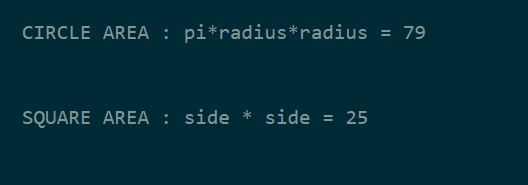
  square1->area(5);

  delete square1;

    return 0;

}

**O/P:**

****

**QUE5.**

**Code:**

#include "book\_management.h"

#include "member\_management.h"

#include "staff\_management.h"

**namespace** LibrarySystem {

**namespace** BookManagement {

    }

}

**namespace** LibrarySystem {

**namespace** MemberManagement {

    }

}

**namespace** LibrarySystem {

**namespace** StaffManagement {

    }

}

**int** main() {

    return 0;

}

**O/P:**