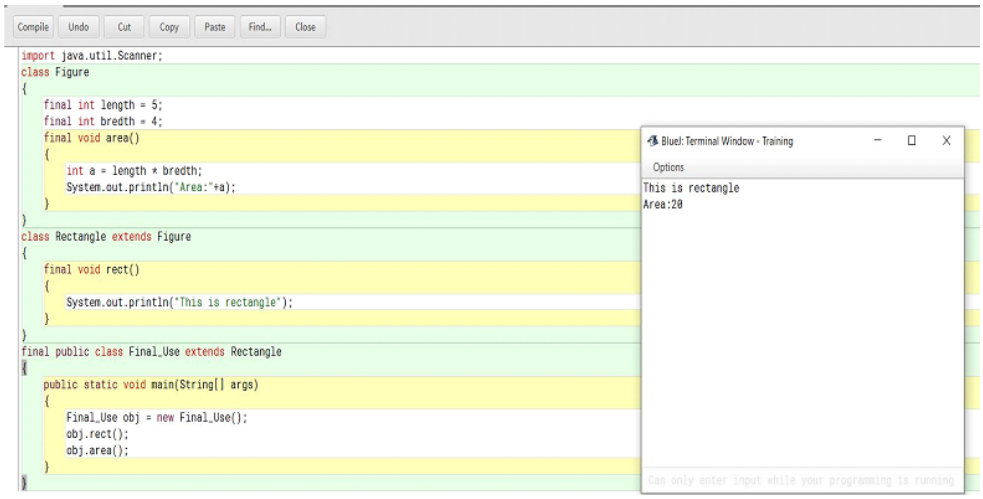
Guneet Kohli 1805172

PROGRAM-11

Program to show the use of 'final' keywords

```
import java.util.Scanner;
class Figure{
   final int length = 5;
   final int bredth = 4;
  final void area()
   { int a = length * bredth;
   System.out.println("Area:"+a); } }
class Rectangle extends Figure
{ final void rect()
   { System.out.println("This is rectangle") ; }
}
final public class Final_Use extends Rectangle
{ public static void main(String[] args)
   {Final_Use obj = new Final_Use();
      obj.rect();
      obj.area(); }
}
```

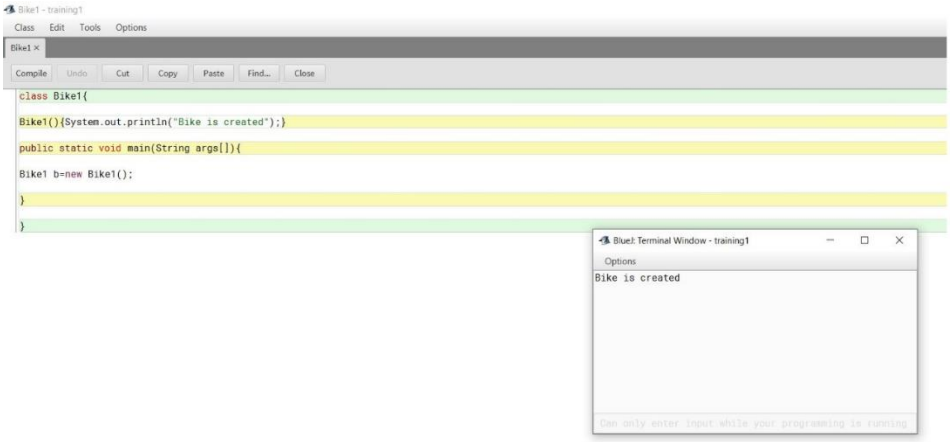


PROGRAM-12

Program to illustrate the concept of default and parameterized constructors.

-Default constructor:

```
class Bike1{
//creating a default constructor
Bike1(){System.out.println("Bike is created");}
//main method
public static void main(String args[]){
//calling a default constructor
Bike1 b=new Bike1();
}
}
```



-Parameterized constructor:

```
 class Student4{
 int id;
  String name;
```

Guneet Kohli 1805172

```
  //creating a parameterized constructor
    Student4(int i,String n){
id = i;
name = n;
}
 //method to display the values
void display(){System.out.println(id+" "+name);}


public static void main(String args[]){
     //creating objects and passing values
Student4 s1 = new Student4(111,"Karan");
Student4 s2 = new Student4(222,"Aryan");
 //calling method to display the values of object
 s1.display();
      s2.display();
         }
     }
```

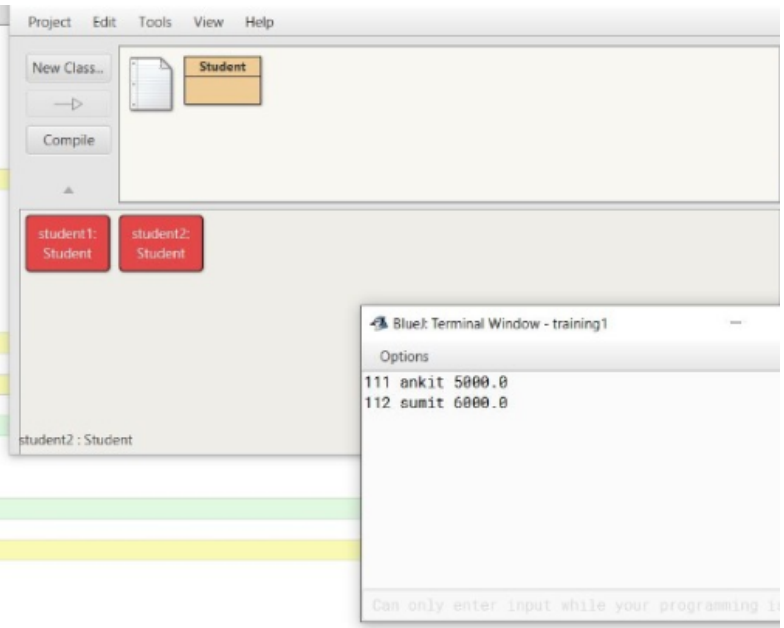
PROGRAM-13

Program to illustrate the concept of ‘this’ pointer.

```
class Student{
int rollno;
String name;
float fee;
Student(int rollno,String name,float fee){
this.rollno=rollno;
this.name=name;
this.fee=fee;
}
void display(){System.out.println(rollno+" "+name+" "+fee);}
}

class TestThis2{
public static void main(String args[]){
Student s1=new Student(111,"ankit",5000f);
Student s2=new Student(112,"sumit",6000f);
s1.display();
s2.display();
}
}
```

Guneet Kohli 1805172

```
int rollno;
String name;
float fee;
Student(int rollno,String name,float fee){
this.rollno=rollno;
this.name=name;
this.fee=fee;
}
void display(){System.out.println(rollno+" "+name+" "+fee);}
}

class TestThis2{
public static void main(String args[]){
Student s1=new Student(111,"ankit",5000f);
Student s2=new Student(112,"sumit",6000f);
s1.display();
s2.display();
}
}
```

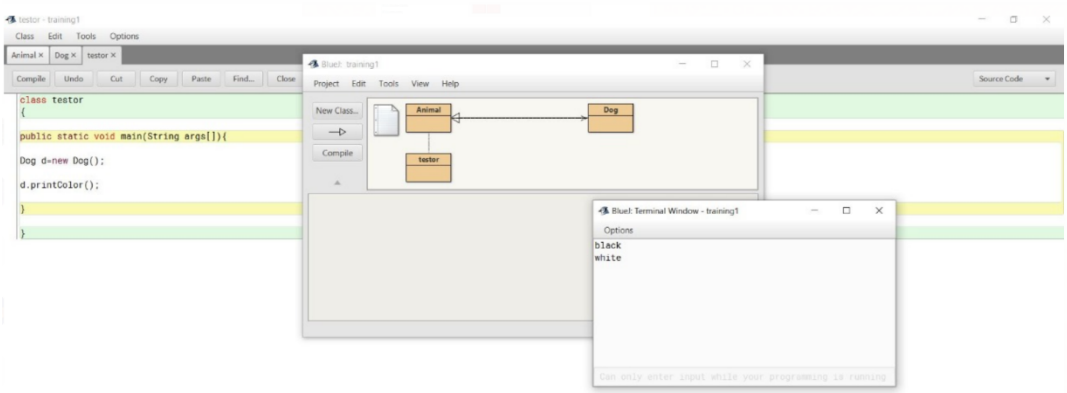


Guneet Kohli 1805172

PROGRAM-14

Program to illustrate the use ‘super’ keyword in inheritance.

```
class Animal{
String color="white";
}
class Dog extends Animal{
String color="black";
void printColor(){
System.out.println(color);//prints color of Dog class
System.out.println(super.color);//prints color of Animal class
}
}
class TestSuper1{
public static void main(String args[]){
Dog d=new Dog();
d.printColor();
}
}
```

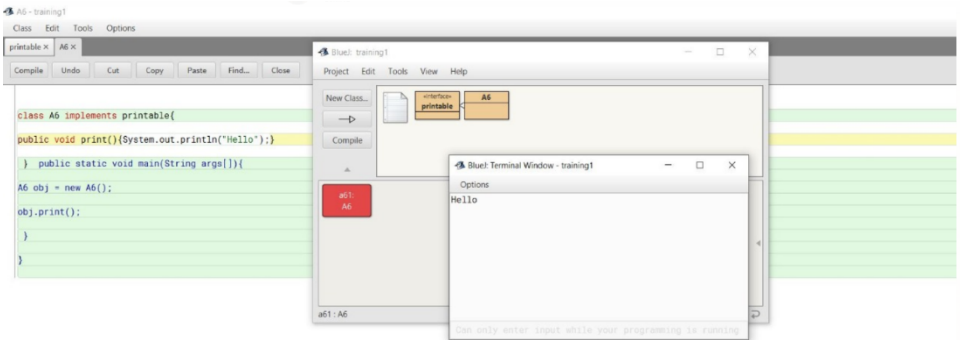


Guneet Kohli 1805172

PROGRAM-15

Program to create an interface and implement it using class.

```
interface printable{
void print();
}
class A6 implements printable{
public void print(){System.out.println("Hello");}

public static void main(String args[]){
A6 obj = new A6();
obj.print();
 }
}
```

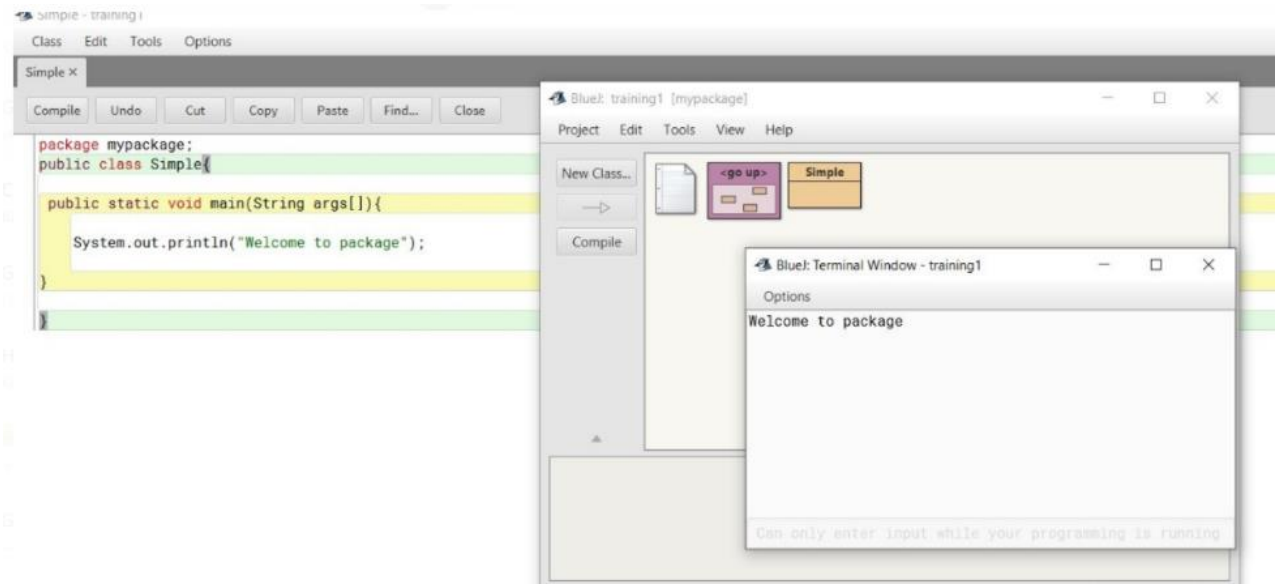


Guneet Kohli 1805172

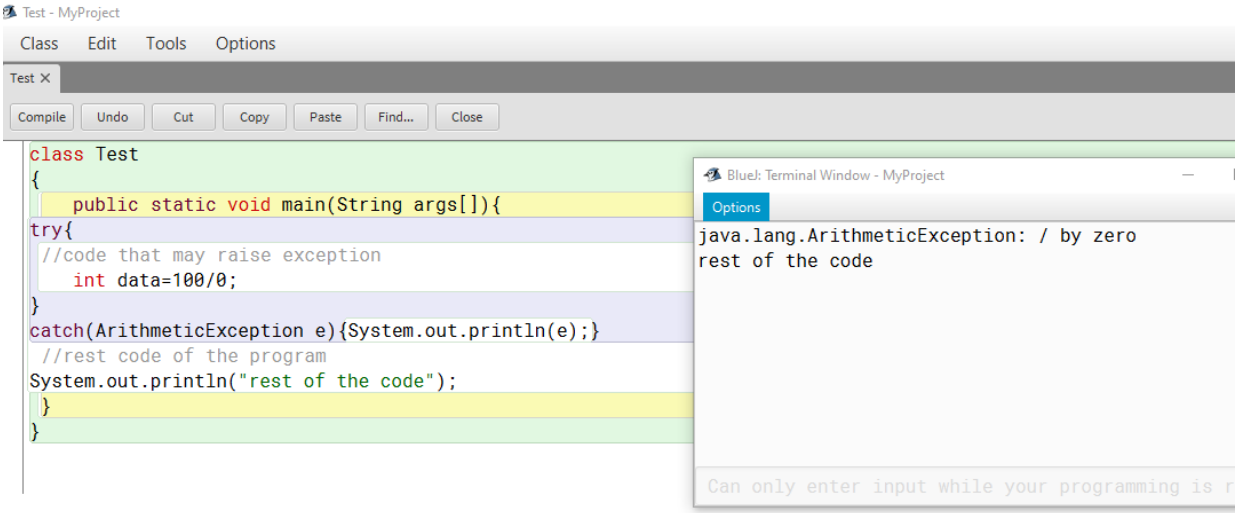
PROGRAM-16

Program to illustrate the concept of packages.

```
package mypack;

public class Simple{

 public static void main(String args[]){

   System.out.println("Welcome to package");

}

}
```



Guneet Kohli 1805172



PROGRAM-17

Program to demonstrate the concept of exception handling.

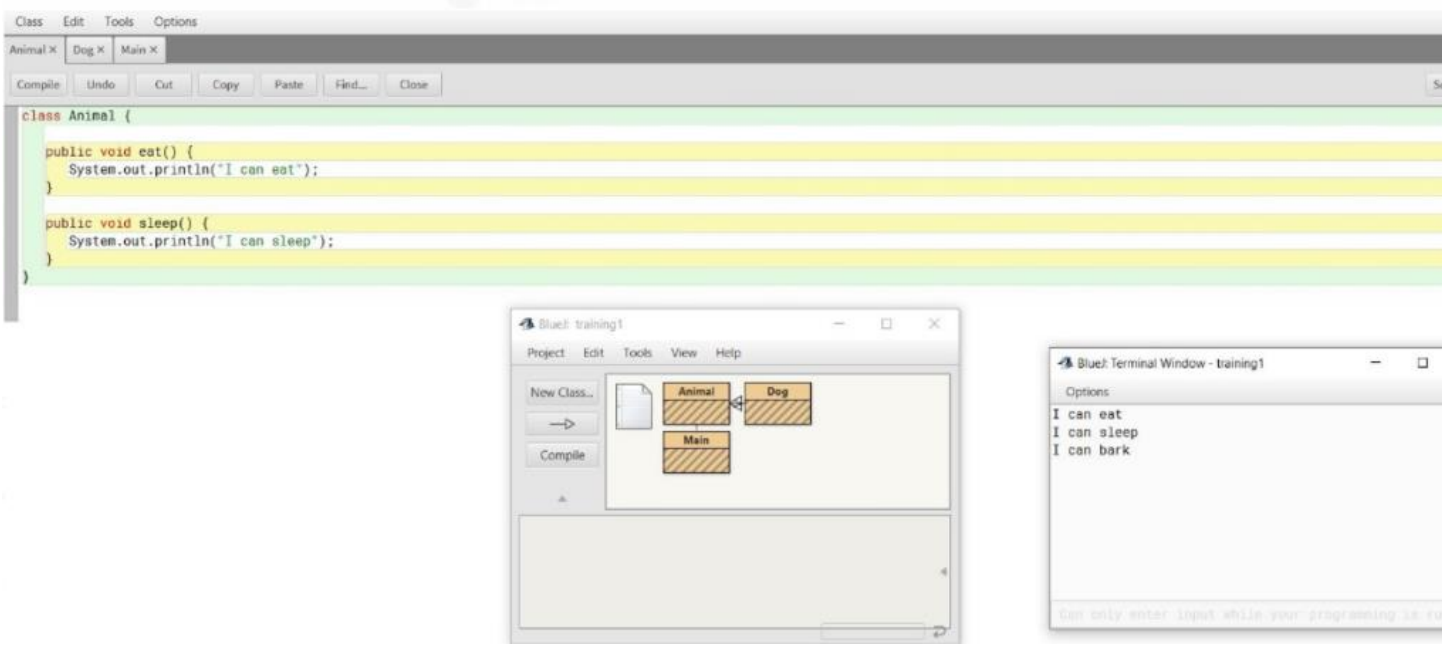
```
Public class Test{
Public static void main(String args[]){
try{
 //code that may raise exception
Int data=100/0;
}catch(ArithmeticException e){System.out.println(e);}
 //rest code of the program
System.out.println("rest of the code...");
 }
}
```

PROGRAM-18

Program to demonstrate the concept of inheritance.

```
class Animal {
   public void eat() {
System.out.println("I can eat");
}
public void sleep() {
      System.out.println("I can sleep");
 }
}
class Dog extends Animal {
public void bark() {
 System.out.println("I can bark");
}
}
class Main {
 public static void main(String[] args) {
Dog dog1 = new Dog();
      dog1.eat();
dog1.sleep();
      dog1.bark();
}}
```

Guneet Kohli 1805172

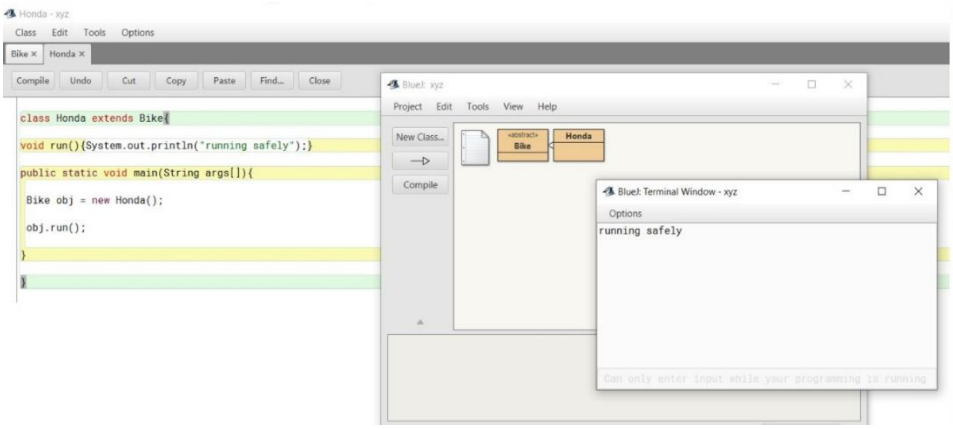


Guneet Kohli 1805172

PROGRAM-19

Program to create an abstract class.

```
abstract class Bike{
abstract void run();
}
class Honda4 extends Bike{
void run(){System.out.println("running safely");}
public static void main(String args[]){
 Bike obj = new Honda4();
 obj.run();
}
}
```



Guneet Kohli 1805172

PROGRAM-20

Program to demonstrate the concept of method overloading.

```
class Multiply {
   void mul(int a, int b) {
      System.out.println("Sum of two=" + (a * b));
}
void mul(int a, int b, int c) {
      System.out.println("Sum of three=" + (a * b * c));
   }
}
public class Polymorphism {
   public static void main(String args[]) {
      Multiply m = new Multiply();
      m.mul(6, 10);
      m.mul(10, 6, 5);
}
}
```