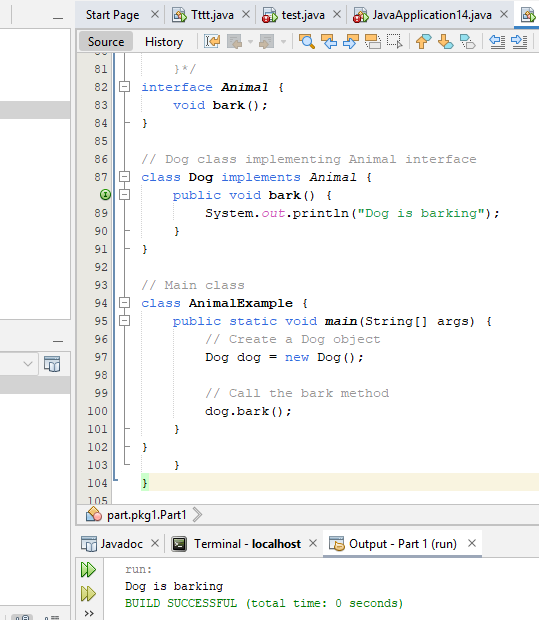
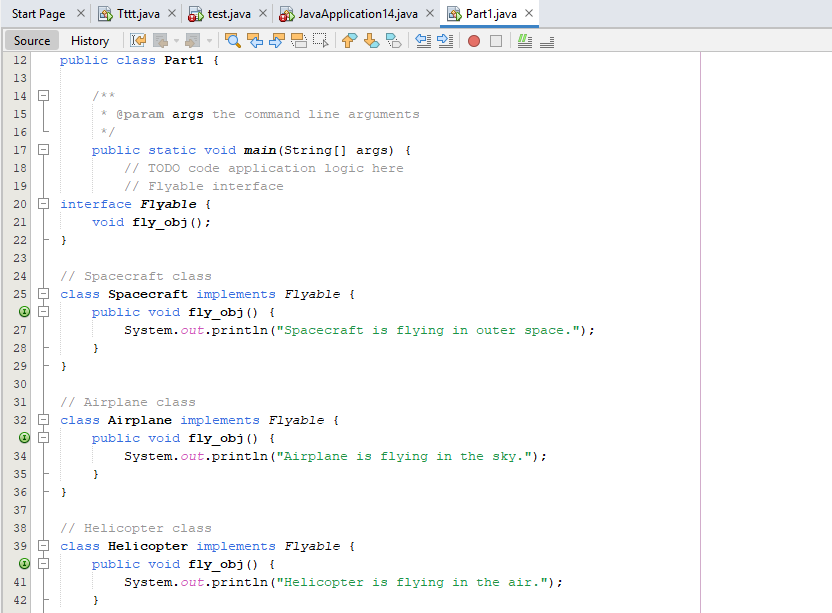
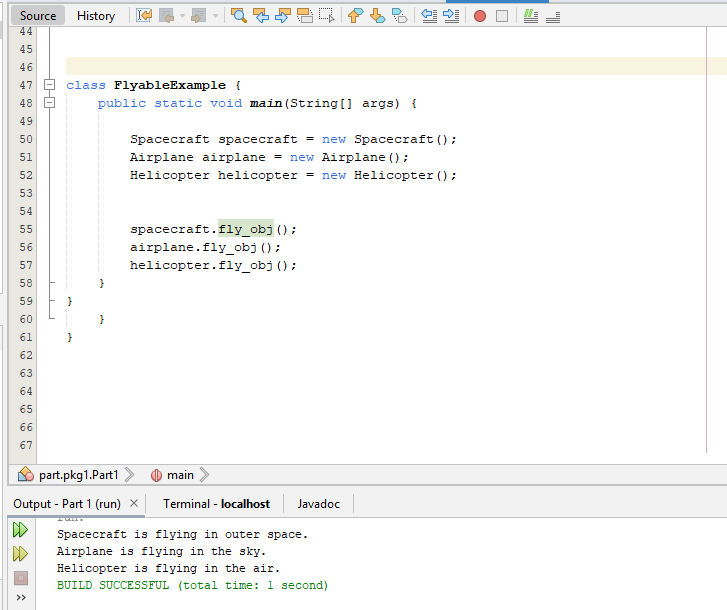
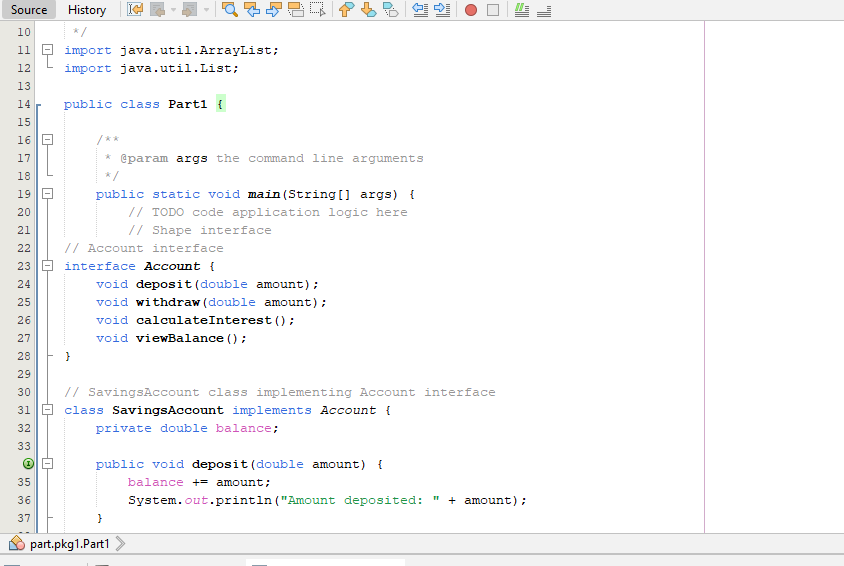
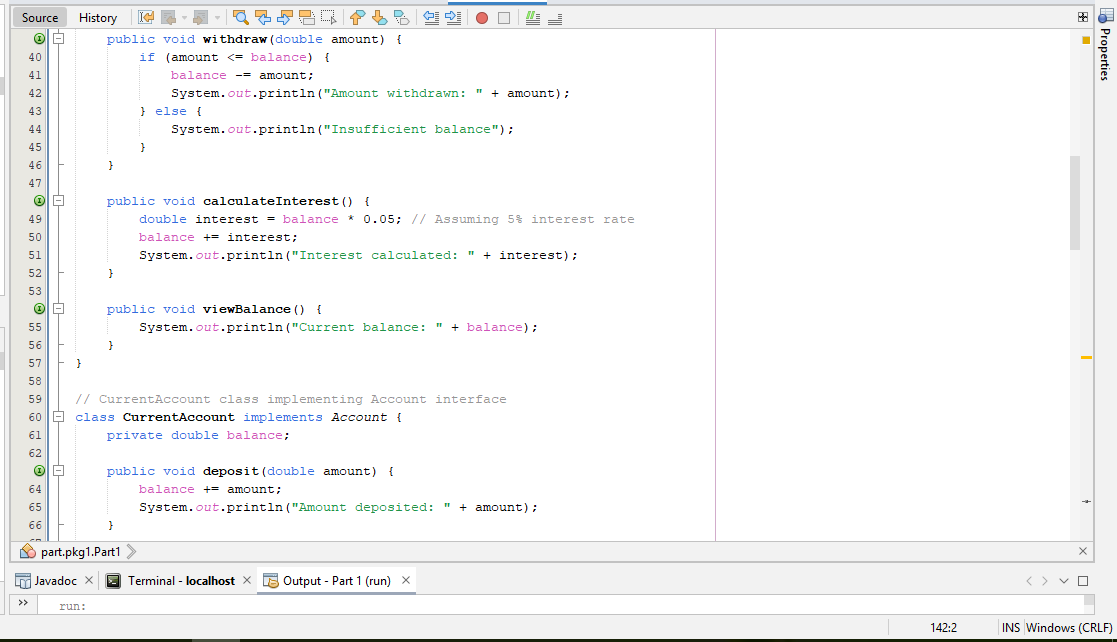
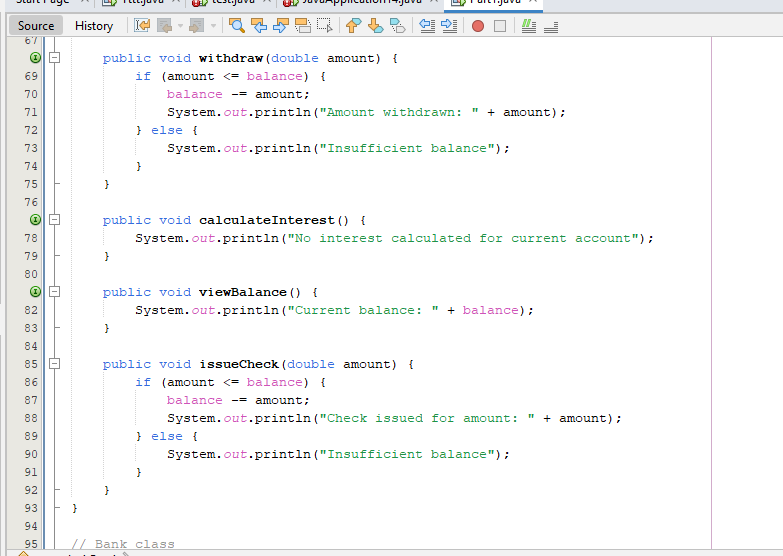
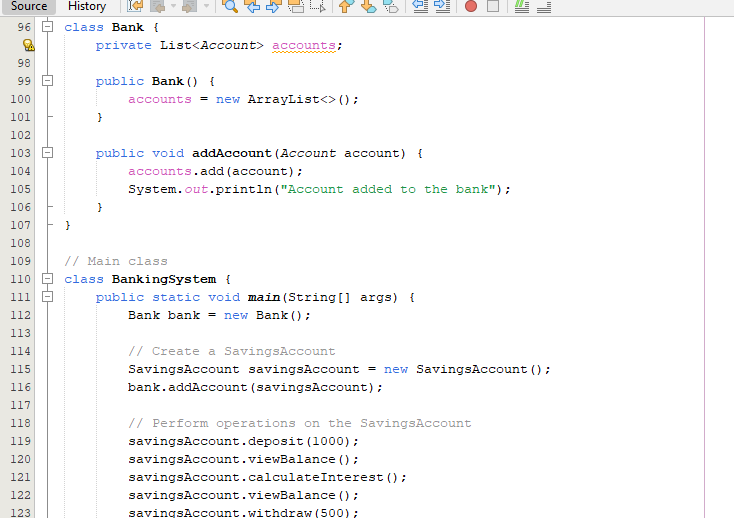
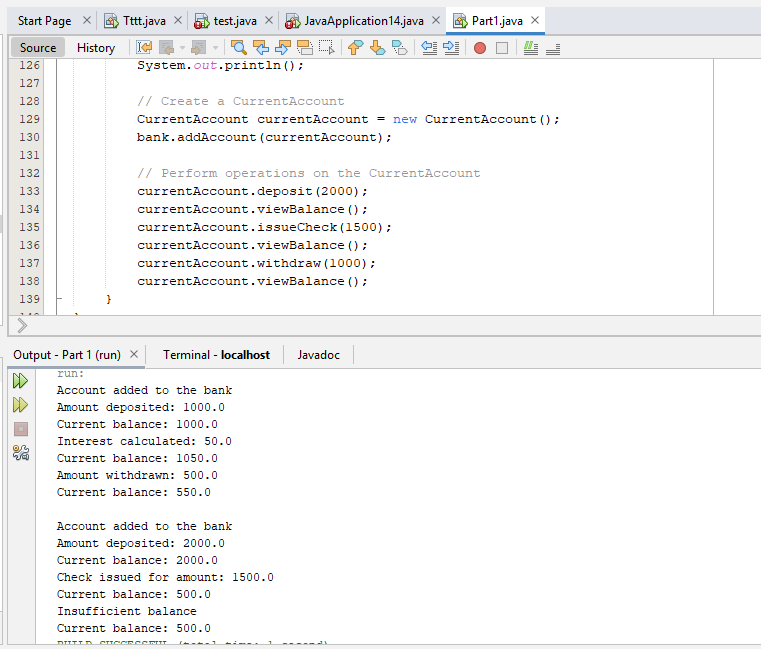
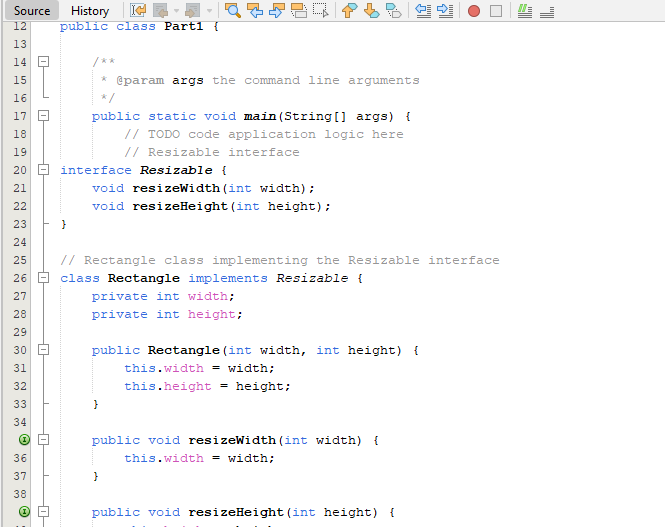
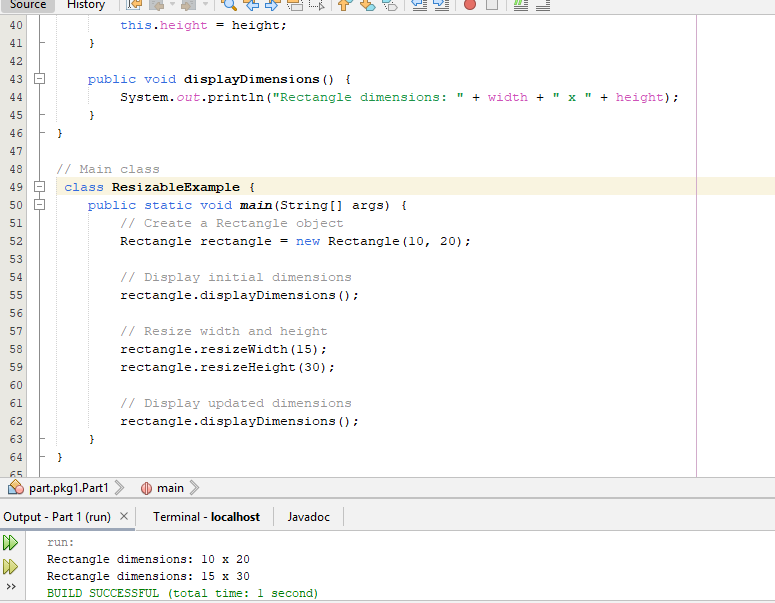
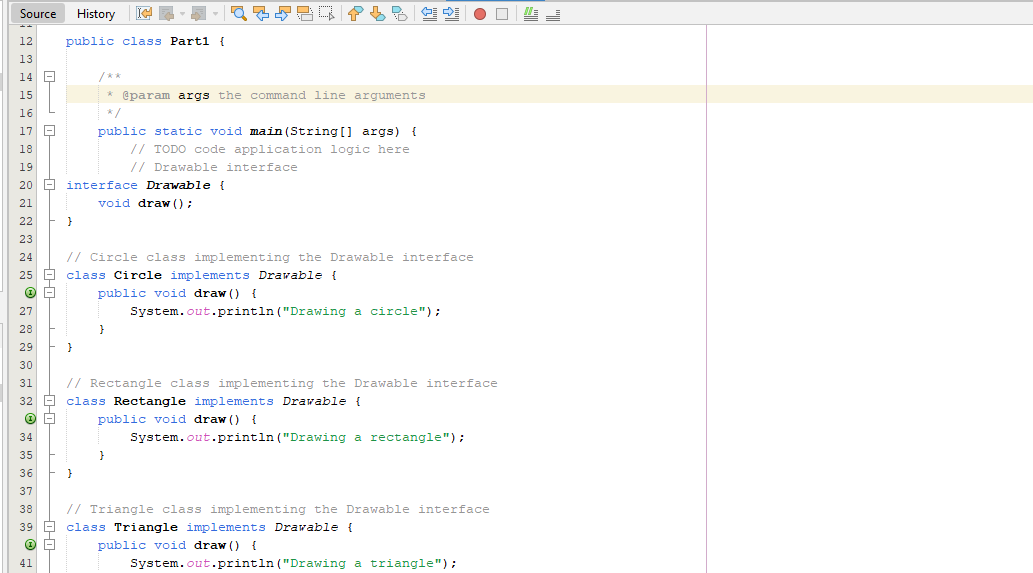
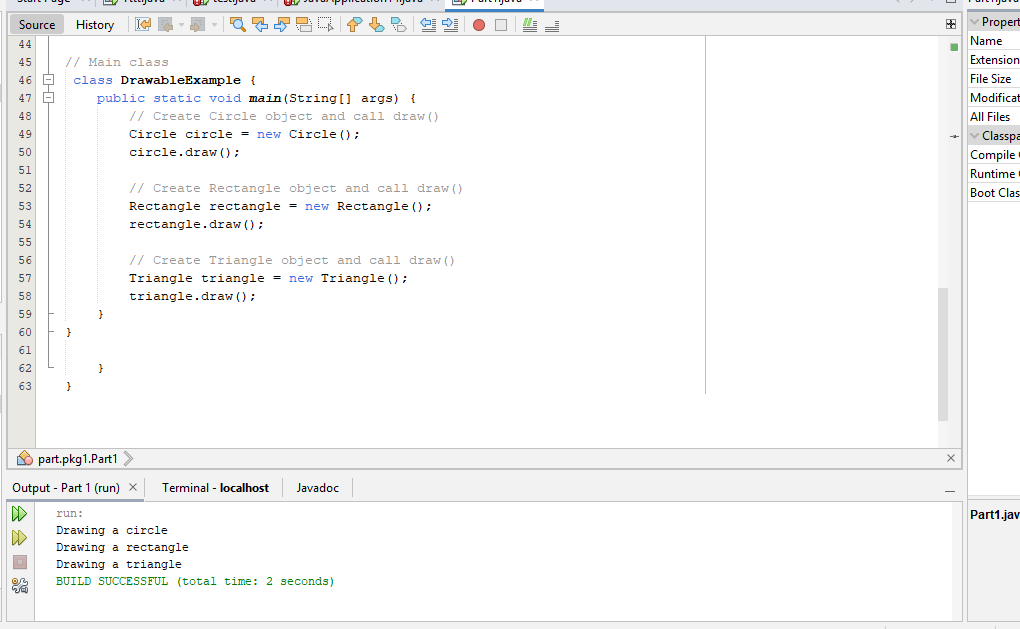
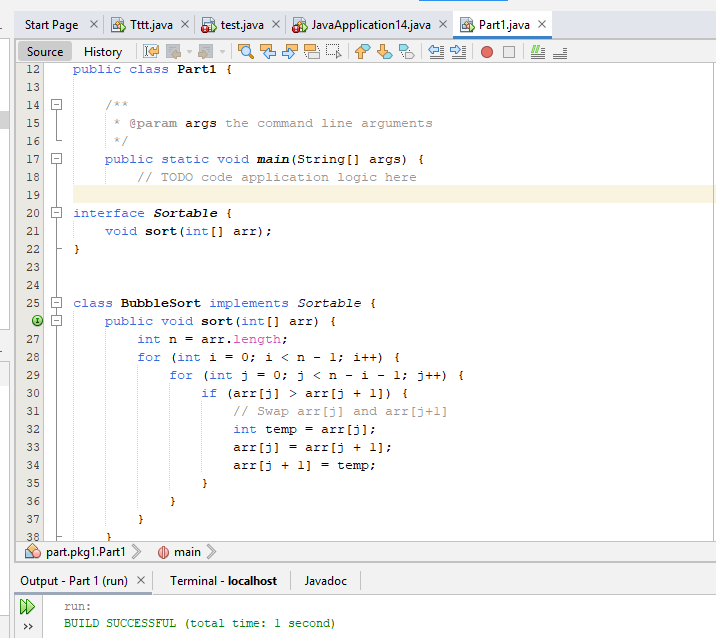
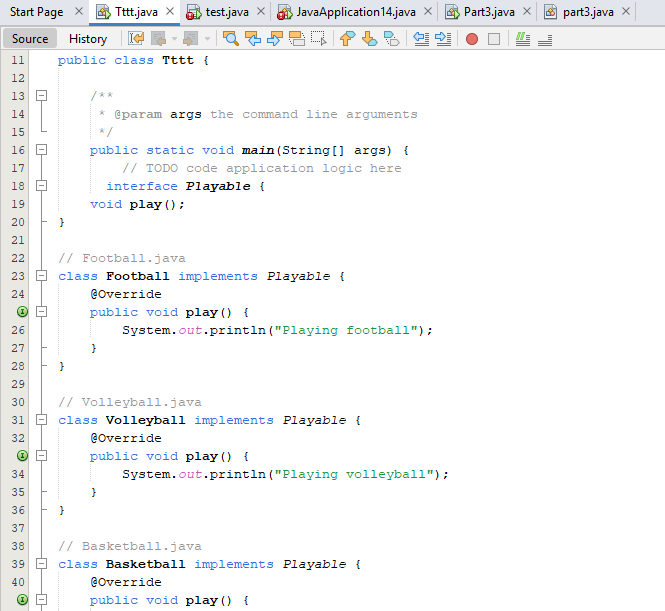
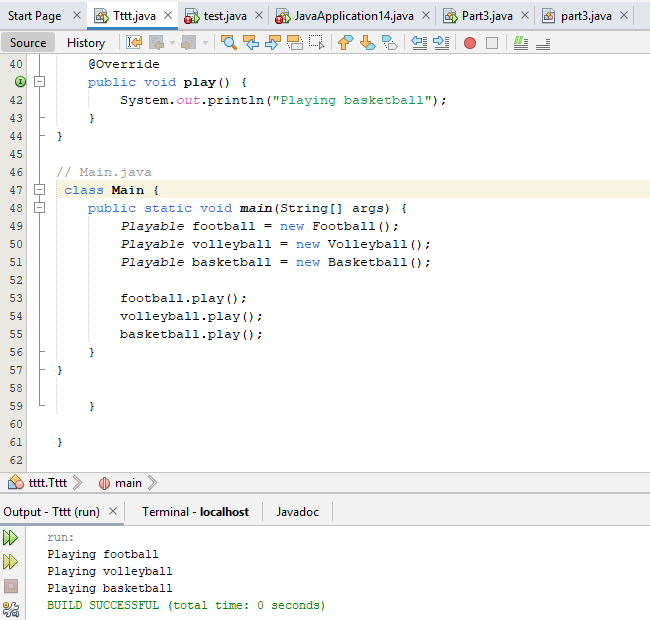
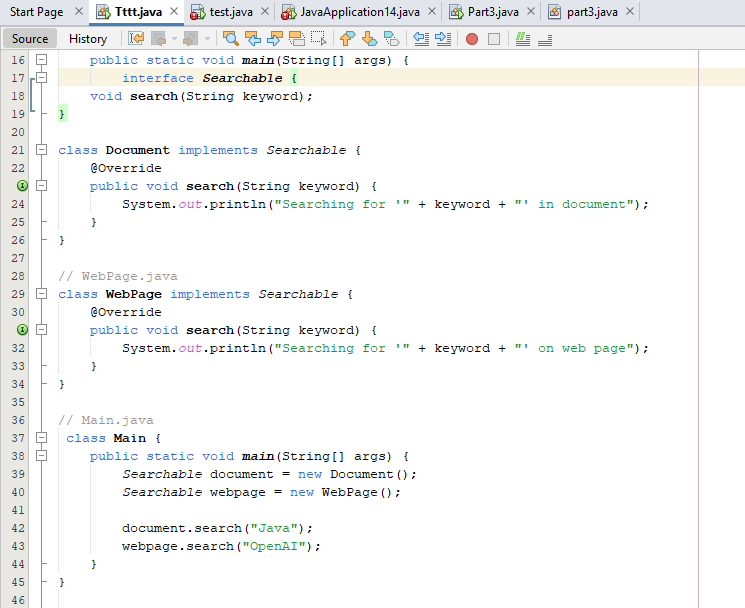
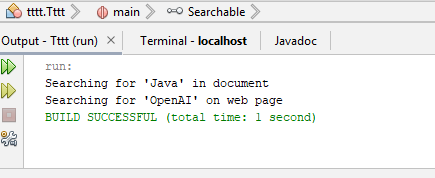
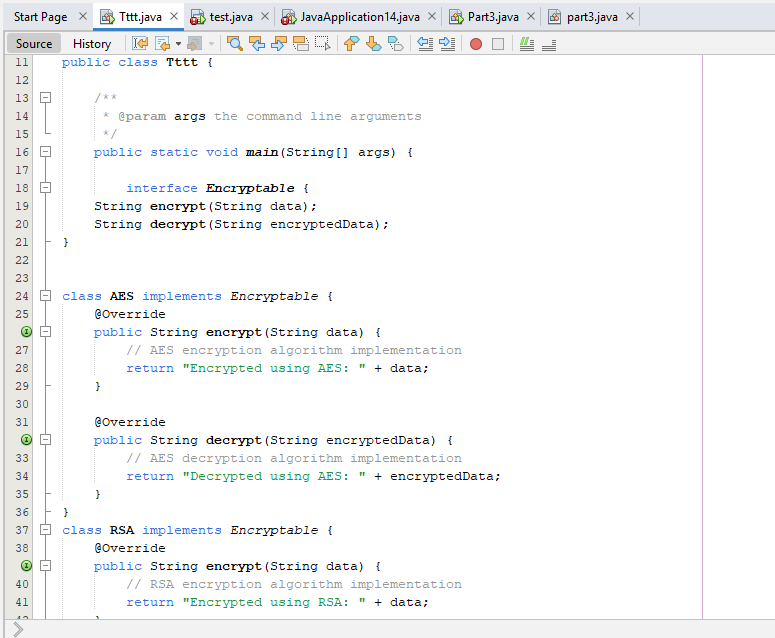
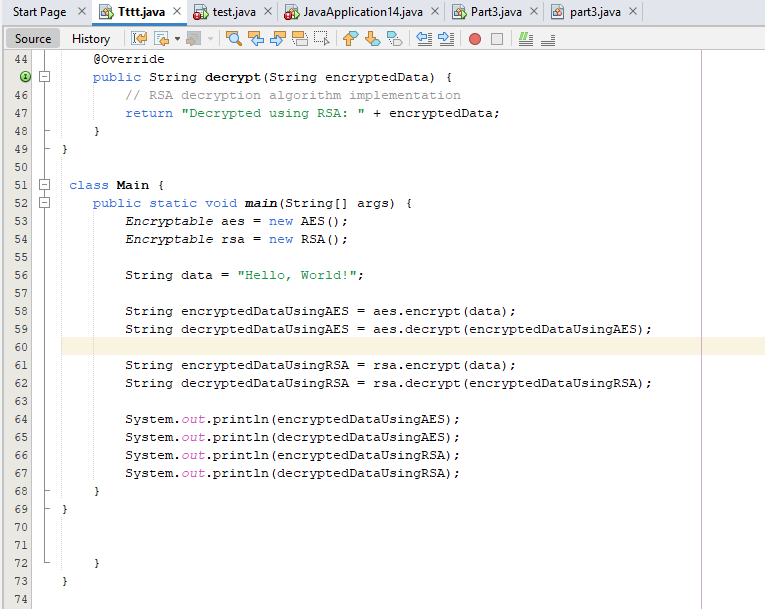
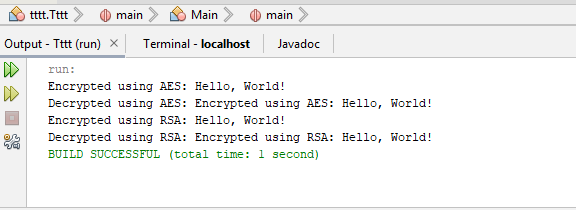
**WORKSHOP 1**

**PART1**

# Java Interface

1. Write a Java program to create an interface Shape with the getArea() method. Create three classes Rectangle, Circle, and Triangle that implement the Shape interface. Implement the getArea() method for each of the three classes.
2. Write a Java program to create a Animal interface with a method called bark() that takes no arguments and returns void. Create a Dog class that implements Animal and overrides speak() to print "Dog is barking". 
3. Write a Java program to create an interface Flyable with a method called fly\_obj(). Create three classes Spacecraft, Airplane, and Helicopter that implement the Flyable interface. Implement the fly\_obj() method for each of the three classes.  
4. Write a Java programming to create a banking system with three classes - Bank, Account, SavingsAccount, and CurrentAccount. The bank should have a list of accounts and methods for adding them. Accounts should be an interface with methods to deposit, withdraw, calculate interest, and view balances. SavingsAccount and CurrentAccount should implement the Account interface and have their own unique methods.     
5. Write a Java program to create an interface Resizable with methods resizeWidth(int width) and resizeHeight(int height) that allow an object to be resized. Create a class Rectangle that implements the Resizable interface and implements the resize methods.  
6. Write a Java program to create an interface Drawable with a method draw() that takes no arguments and returns void. Create three classes Circle, Rectangle, and Triangle that implement the Drawable interface and override the draw() method to draw their respective shapes.  
7. Write a Java program to create an interface Sortable with a method sort() that sorts an array of integers in ascending order. Create two classes BubbleSort and SelectionSort that implement the Sortable interface and provide their own implementations of the sort() method. 
8. Write a Java program to create an interface Playable with a method play() that takes no arguments and returns void. Create three classes Football, Volleyball, and Basketball that implement the Playable interface and override the play() method to play the respective sports.  
9. Write a Java program to create an interface Searchable with a method search(String keyword) that searches for a given keyword in a text document. Create two classes Document and WebPage that implement the Searchable interface and provide their own implementations of the search() method.  
10. Write a Java program to create an interface Encryptable with methods encrypt (String data) and decrypt (String encryptedData) that define encryption and decryption operations. Create two classes AES and RSA that implement the Encryptable interface and provide their own encryption and decryption algorithms.   

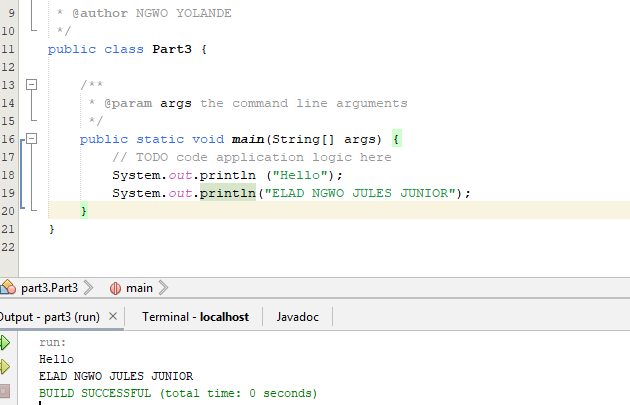
**PART2**

1. Write a Java program to print 'Hello' on screen and your name on a separate line.

*Expected Output* :

Hello

Alexandra Abramov

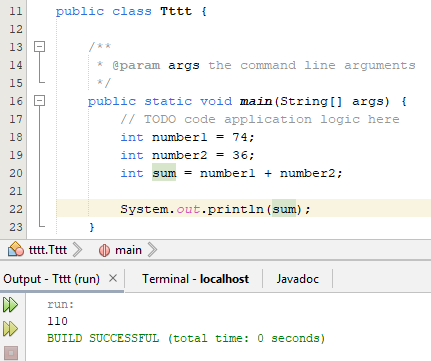


1. Write a Java program to print the sum of two numbers.

Test Data:

74 + 36

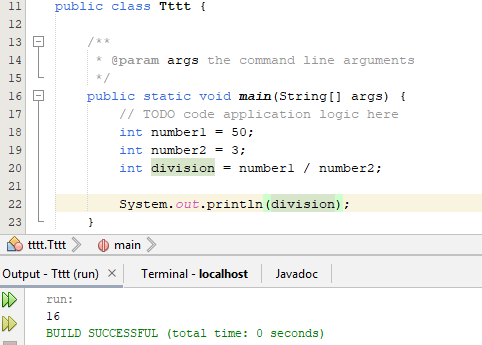
*Expected Output* : 110



1. Write a Java program to divide two numbers and print them on the screen. Test Data :

50/3

*Expected Output* : 16



1. Write a Java program to print the results of the following operations.

*Test Data:*

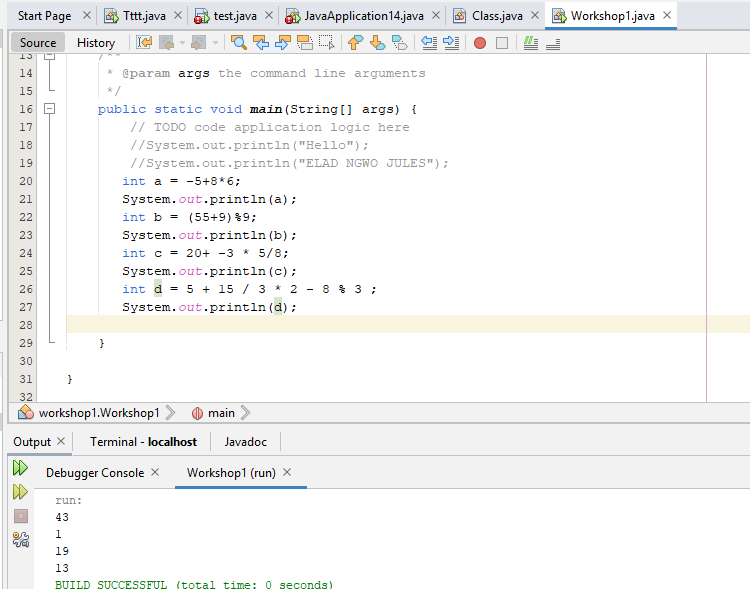
1. -5 + 8 \* 6
2. (55+9) % 9
3. 20 + -3\*5 / 8
4. 5 + 15 / 3 \* 2 - 8 % 3 *Expected Output* :

43

1

19

13

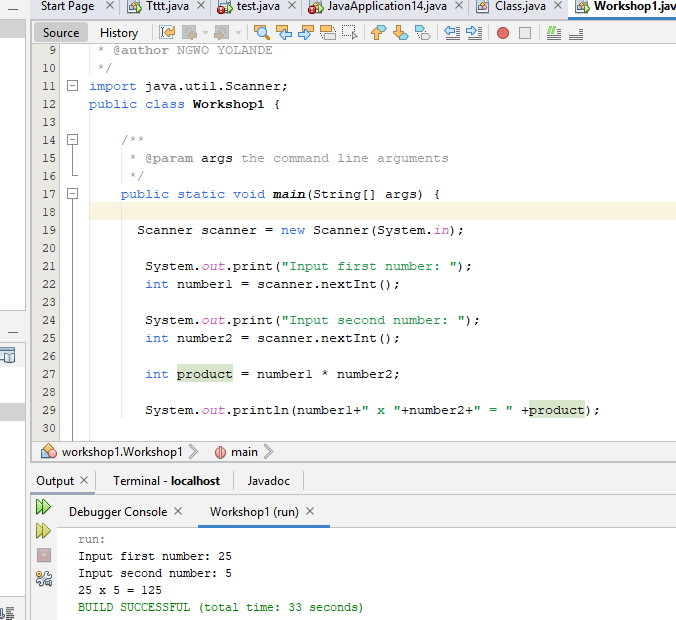


**5.** Write a Java program that takes two numbers as input and displays the product of two numbers.

*Test Data:*

Input first number: 25 Input second number: 5 *Expected Output* :

25 x 5 = 125



PART3

## Java Polymorphism

**Polymorphism**

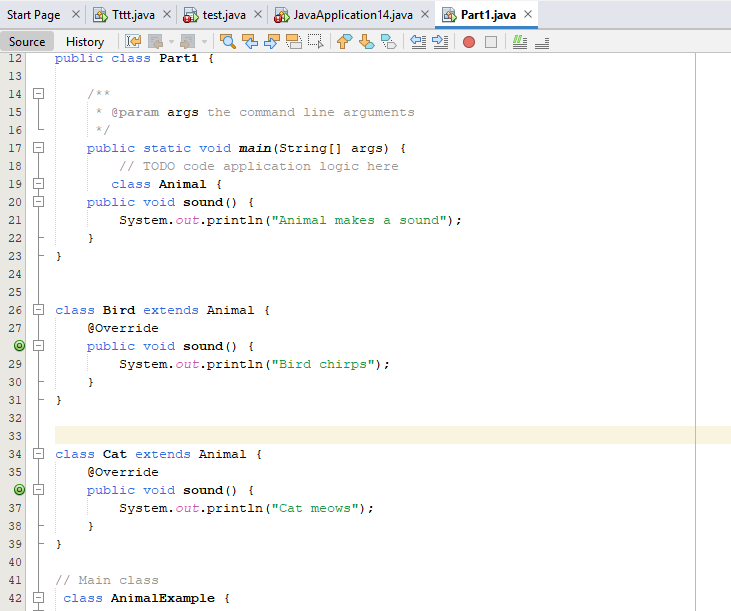
In Core, Java Polymorphism is one of easy concept to understand.

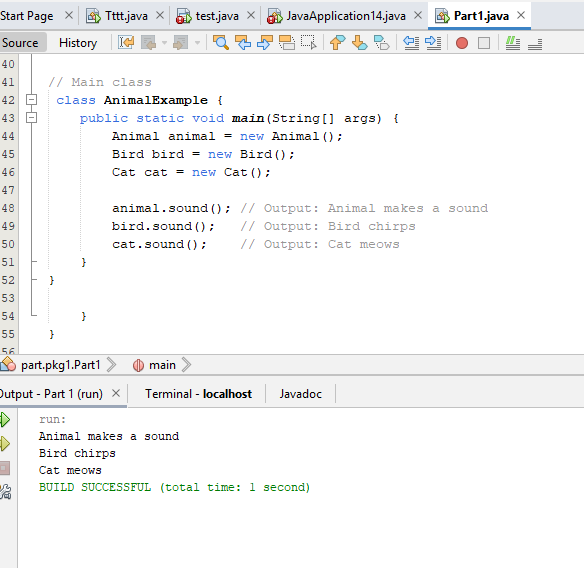
Polymorphism definition is that Poly means many and morphos means forms. It describes the feature of languages that allows the same word or symbol to be interpreted correctly in different situations based on the context. There are two types of Polymorphism available in Java. For example, in English, the verb

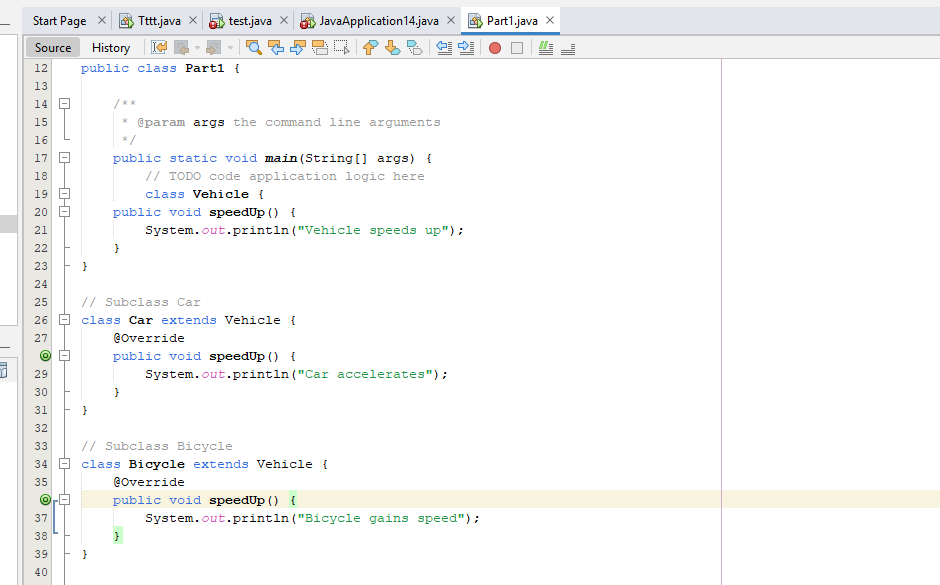
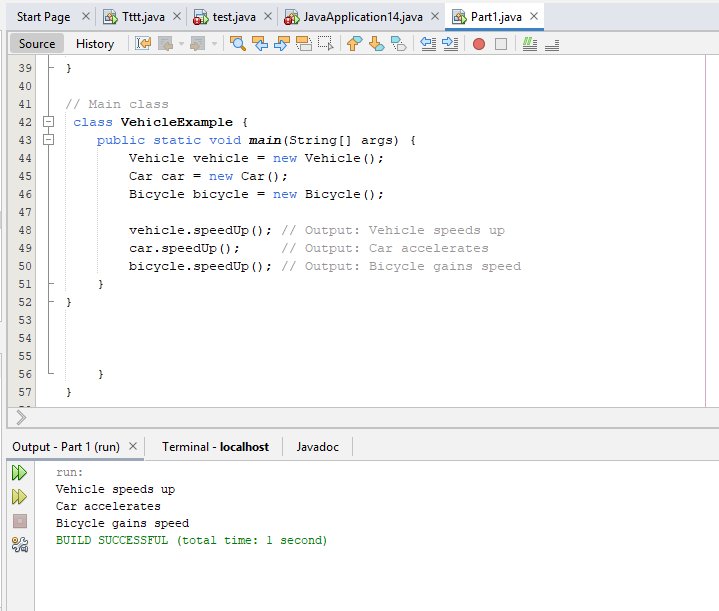
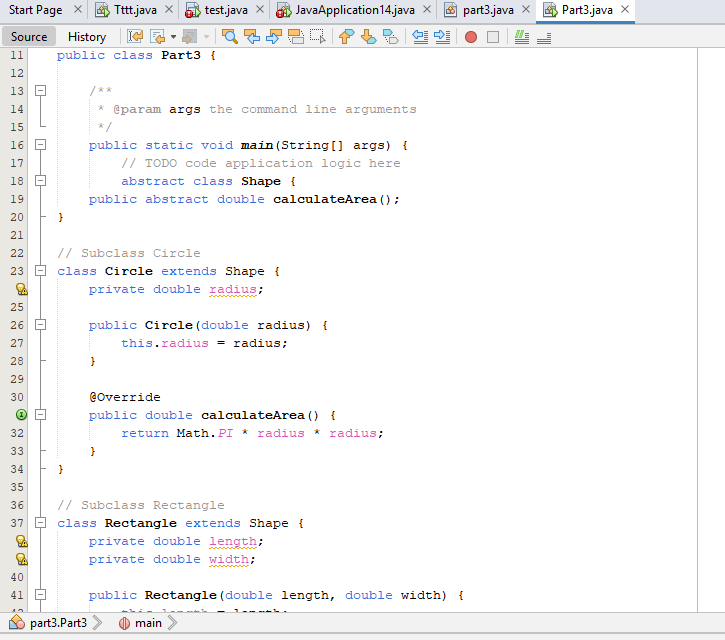
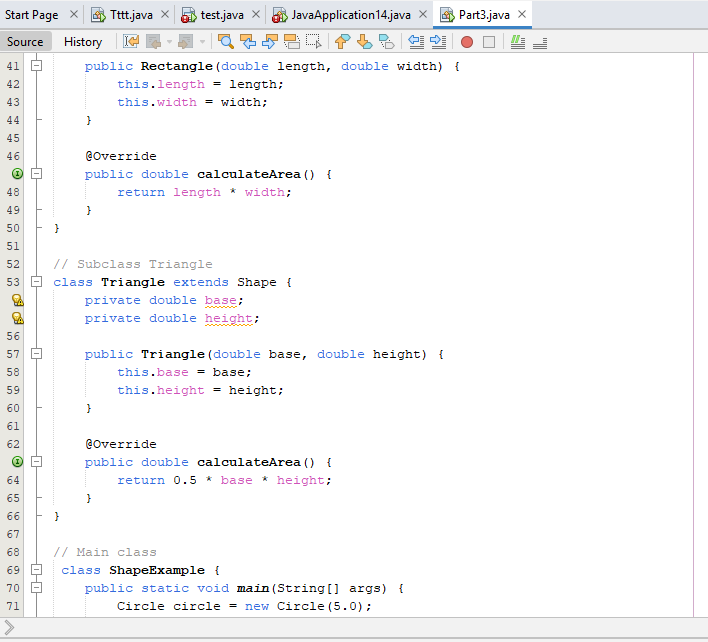
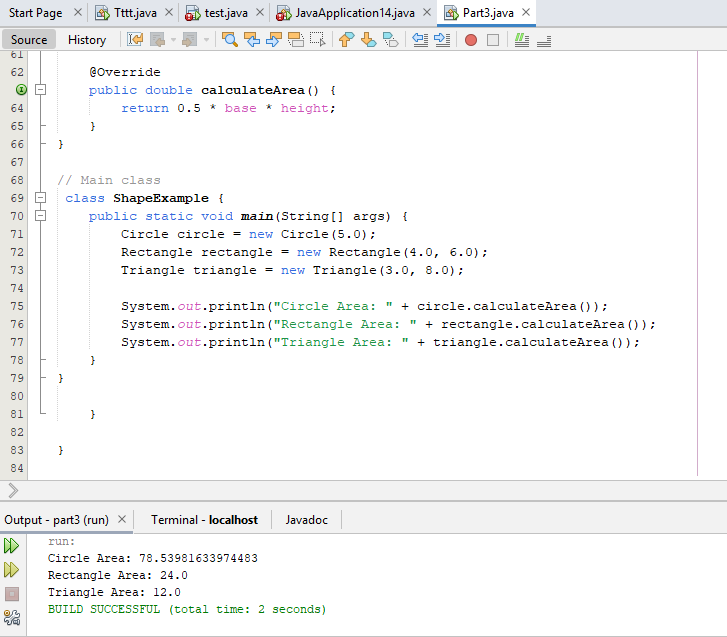
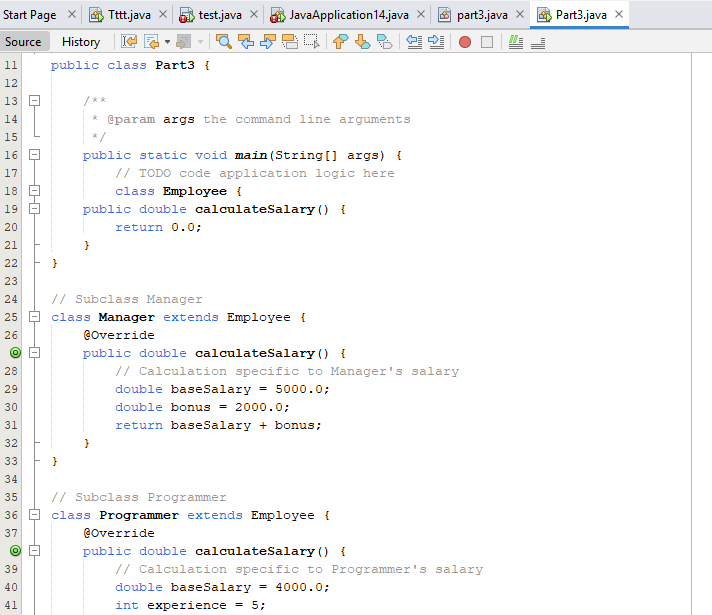
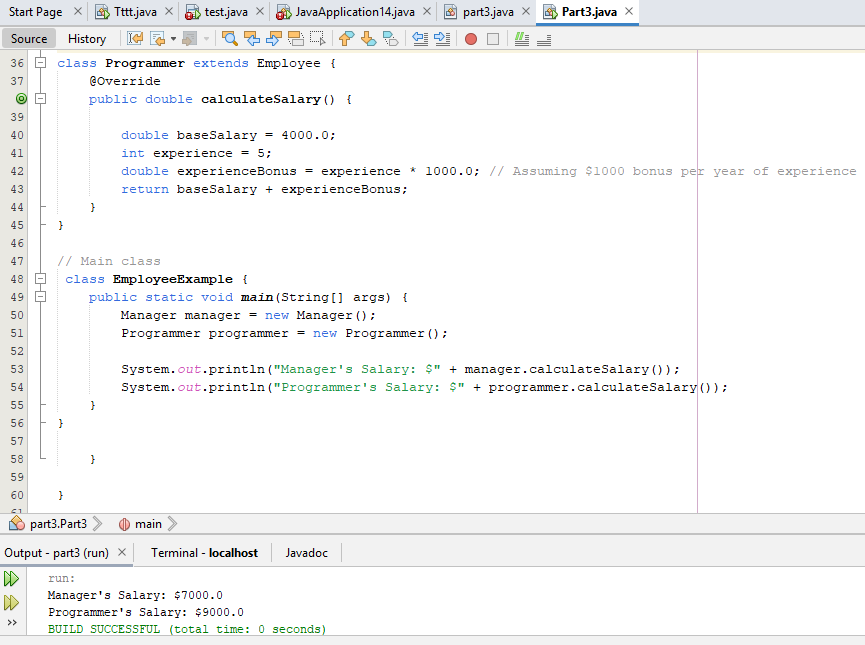
“run” means different things if you use it with “a footrace,” a “business,” or “a computer.” You understand the meaning of “run” based on the other words used with it..

1. Write a Java program to create a base class Animal (Animal Family) with a method called Sound(). Create two subclasses Bird and Cat. Override the

Sound() method in each subclass to make a specific sound for each animal.





1. Write a Java program to create a class Vehicle with a method called speedUp(). Create two subclasses Car and Bicycle. Override the speedUp() method in each subclass to increase the vehicle's speed differently.  
2. Write a Java program to create a base class Shape with a method called calculateArea(). Create three subclasses: Circle, Rectangle, and Triangle. Override the calculateArea() method in each subclass to calculate and return the shape's area.   
3. Write a Java program to create a class Employee with a method called calculateSalary(). Create two subclasses Manager and Programmer. In each subclass, override the calculateSalary() method to calculate and return the salary based on their specific roles.  
4. Write a Java program to create a base class Sports with a method called play(). Create three subclasses: Football, Basketball, and Rugby. Override the play() method in each subclass to play a specific statement for each sport.

