

## Section - 1

**Basic Inheritance(Single Inheritance):**

1. Create a class Animal with properties like name and age. Create a subclass Dog that inherits from Animal and adds a property breed. Demonstrate the use of constructors in both the Animal and Dog classes.

```
1
2 public class Animal {
3     String name;
4     int age;
5     public Animal() {
6         System.out.println("Animal");
7     }
8 }
9
10 class Dog extends Animal{
11     String breed;
12
13     public Dog() {
14         super();
15     }
16 }
17 }
18 }
```

```
1 public class MetOverr{
2     public static void main(String[] args) {
3         Dog d= new Dog();
4         d.name="Leo";
5         d.age=4;
6         d.breed="Husky";
7
8         System.out.println("Name"+d.name);
9         System.out.println("Age: "+d.age);
10        System.out.println("Breed: "+d.breed);
11    }
12 }
```

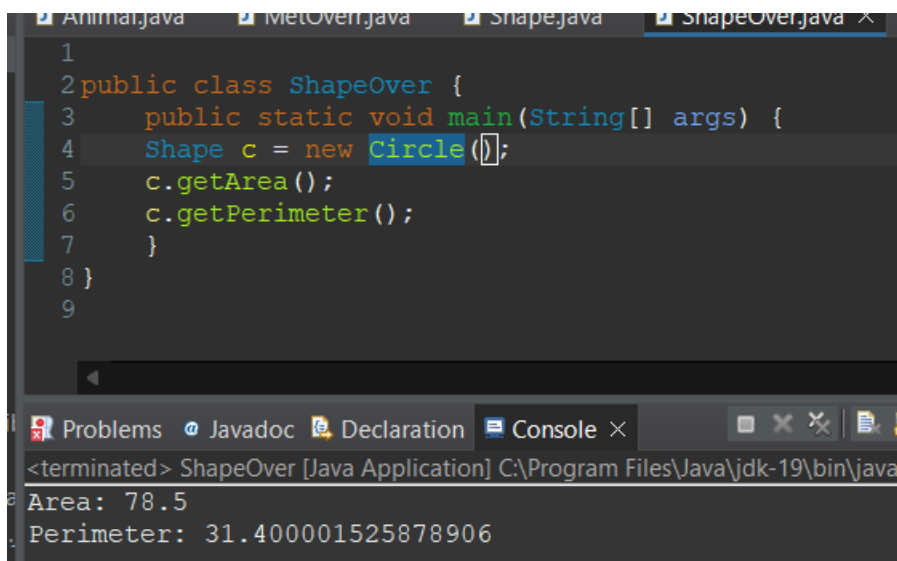
<terminated> MetOverr [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (Dec 21)

Animal  
NameLeo  
Age: 4  
Breed: Husky

**Method Overriding:**

2. Write a Java program to create a class called Shape with methods called getPerimeter() and getArea(). Create a subclass called Circle that overrides the getPerimeter() and getArea() methods to calculate the area and perimeter of a circle

```
1
2 public class Shape {
3     public void getPerimeter() {
4         System.out.println("This is Perimeter.");
5     }
6     public void getArea() {
7         System.out.println("This is Area.");
8     }
9
10 }
11
12 class Circle extends Shape{
13     int r=5;
14     final float pi= (float) 3.14;
15     public void getPerimeter() {
16         double perimeter = 2*pi*r;
17         System.out.println("Perimeter: "+perimeter);
18     }
19     public void getArea() {
20         double area= pi*r*r;
21         System.out.println("Area: "+area);
22     }
23
24
25 }
```



The screenshot shows an IDE with four tabs: Animal.java, MetOver.java, Shape.java, and ShapeOver.java. The ShapeOver.java tab is active, displaying the following code:

```
1
2 public class ShapeOver {
3     public static void main(String[] args) {
4         Shape c = new Circle();
5         c.getArea();
6         c.getPerimeter();
7     }
8 }
9
```

Below the code editor, the console window shows the output of the program:

```
<terminated> ShapeOver [Java Application] C:\Program Files\Java\jdk-19\bin\java
Area: 78.5
Perimeter: 31.400001525878906
```

**Super Keyword:**

3. Extend the Animal and Dog example by adding a constructor to the Animal class that takes a name parameter. In the Dog class, use the super keyword to call the constructor of the Animal class. Create instances of Dog and demonstrate the use of the super keyword.

```
1 public class Animal {
2     String name;
3
4     public Animal(String name) {
5         this.name = name;
6     }
7 }
8
9 class Dog extends Animal {
10     public Dog(String name) {
11         super(name);
12     }
13 }
```

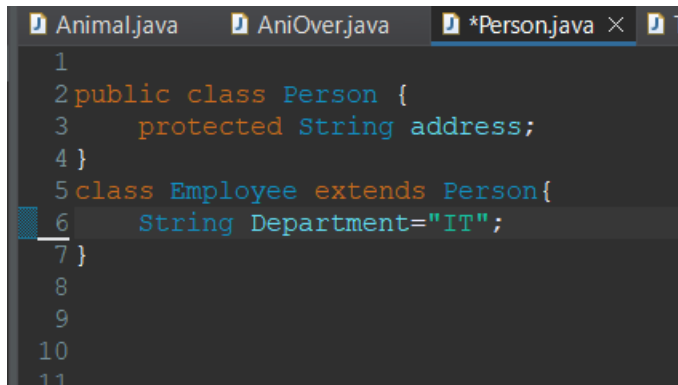
```
*Animal.java  Shape.java  ShapeOver.java  AniOver.java x 55
1 public class AniOver {
2     public static void main(String[] args) {
3         // Create a Dog instance with the name "Buddy"
4         Animal a = new Dog("Buddy");
5
6         // Print the name of the animal
7         System.out.println("Name of my dog is: "+a.name);
8     }
9 }
```

Problems Javadoc Declaration Console x

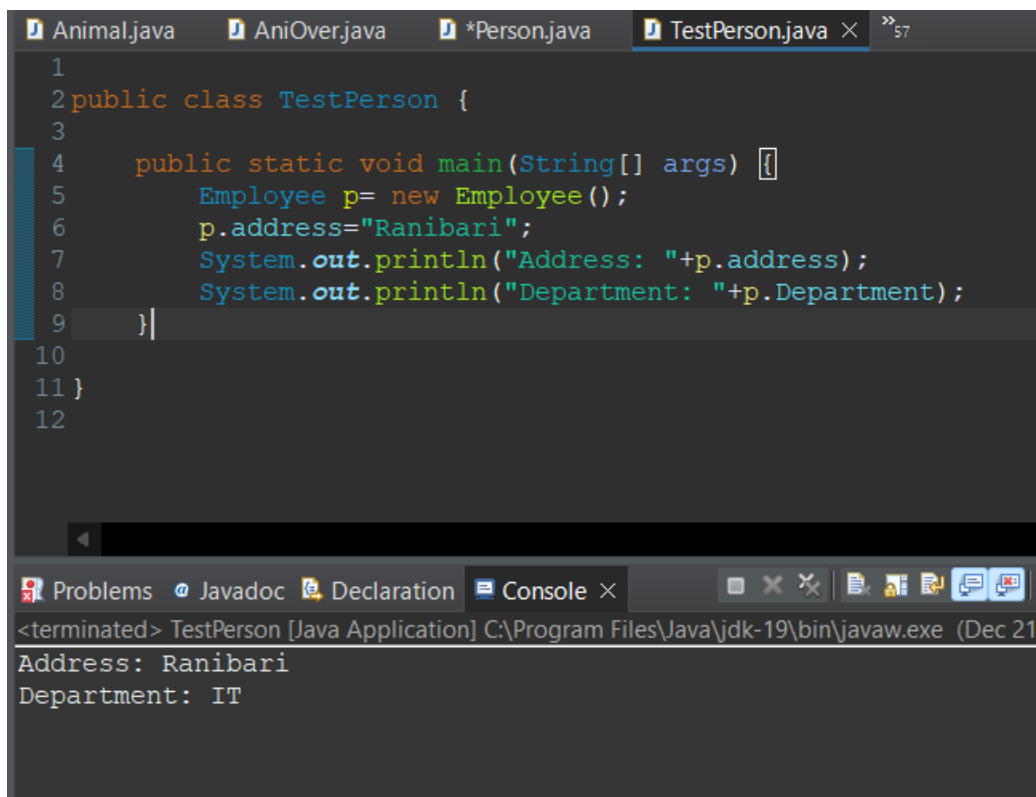
<terminated> AniOver [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (Dec 21, 2023)  
Name of my dog is: Buddy

**Protected Keyword:**

4. Create a class Person with a protected property address. Extend it with a subclass Employee that adds a department property. Demonstrate how the protected keyword allows access to the address property in the Employee subclass.



```
1
2 public class Person {
3     protected String address;
4 }
5 class Employee extends Person {
6     String Department="IT";
7 }
8
9
10
11
```



```
1
2 public class TestPerson {
3
4     public static void main(String[] args) {
5         Employee p= new Employee();
6         p.address="Ranibari";
7         System.out.println("Address: "+p.address);
8         System.out.println("Department: "+p.Department);
9     }
10
11 }
12
```

<terminated> TestPerson [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (Dec 21  
Address: Ranibari  
Department: IT

**Access Modifiers and Inheritance:**

5. Create class Parent with a private variable, a protected variable, and a public variable. Create a subclass Child and demonstrate how each type of variable is accessed (or not accessed) within the subclass.

```
1
2 public class Parent {
3     private String name="Merina";
4     protected int age=19;
5     public String caste="Shrestha";
6
7 }
8 class Child extends Parent{
9     public String name() {
10         return name; // Error: privateVariable
11     }
12     public int age() {
13         return age;
14     }
15     public String caste() {
16         return caste;
17     }
18 }
19
```

```
1
2 public class TestParent {
3
4     public static void main(String[] args) {
5         Child child = new Child();
6         System.out.println("Protected Variable: " + child.age());
7         System.out.println("Public Variable: " + child.caste());
8     }
9
10 }
11
```

Problems Javadoc Declaration Console ×

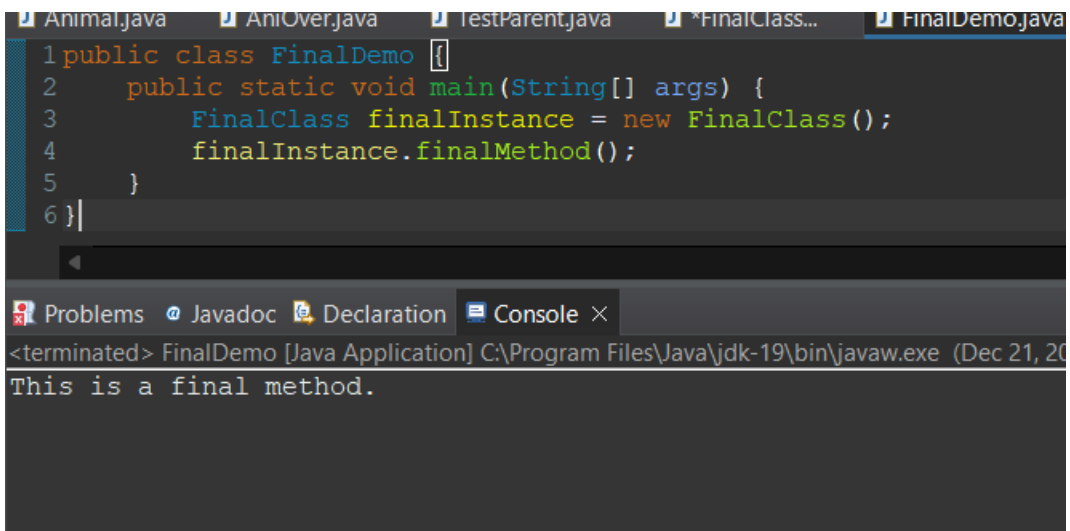
<terminated> TestParent [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (Dec 21, 2023, 1:57)

Protected Variable: 19  
Public Variable: Shrestha

**Final Classes and Methods:**

6. Create a final class FinalClass. Attempt to extend it with another class and observe the compiler error. Also, create a final method within a class and try to override it in a subclass.

```
1 final class FinalClass {
2     // Final method in the final class
3     public final void finalMethod() {
4         System.out.println("This is a final method.");
5     }
6 }
7
8 // Uncomment the following code to observe compiler error when attempting
9
10 // class Subclass extends FinalClass {
11 //     // Error: Cannot inherit from final 'FinalClass'
12 // }
13
14
15 class AnotherClass {
16     // Attempt to override a final method (will result in a compilation
17     // Uncomment the following code to observe the compilation error
18     /*
19     public final void finalMethod() {
20         System.out.println("Attempting to override a final method.");
21     }
22     */
23 }
24
25
```



The screenshot shows an IDE with several tabs open: Animal.java, AniOver.java, TestParent.java, \*FinalClass..., and FinalDemo.java. The FinalDemo.java tab is active, showing the following code:

```
1 public class FinalDemo {
2     public static void main(String[] args) {
3         FinalClass finalInstance = new FinalClass();
4         finalInstance.finalMethod();
5     }
6 }
```

Below the code editor, there is a console window with the following output:

```
<terminated> FinalDemo [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (Dec 21, 2023)
This is a final method.
```

**Method overloading**

7. Create a class ,'Calculator' that should be able to perform addition operations for both integers and doubles. Implement the following steps:
  - a. Create an instance of the Calculator class.
  - b. Use the add method to add two integers (e.g., 5 and 8) and display the result.
  - c. Call the add method with three integers (e.g., 10, 15, and 20) and display the result.
  - d. Use the add method to add two doubles (e.g., 3.5 and 2.7) and display the result.
  - e. Call the add method with three doubles (e.g., 1.1, 2.2, and 3.3) and display the result.

```
1 public class Calculator {
2
3     public int add(int a, int b) {
4         return a + b;
5     }
6
7     public int add(int a, int b, int c) {
8         return a + b + c;
9     }
10
11    public double add(double a, double b) {
12        return a + b;
13    }
14    public double add(double a, double b, double c) {
15        return a + b + c;
16    }
17
18    public static void main(String[] args) {
19
20        Calculator calculator = new Calculator();
21
22        int resultIntTwo = calculator.add(9, 4);
23        System.out.println("Result of adding two integers: " + resultIntTwo);
24
25        int resultIntThree = calculator.add(11, 25, 20);
26        System.out.println("Result of adding three integers: " + resultIntThree);
27
28        double resultDoubleTwo = calculator.add(4.5, 2.7);
29        System.out.println("Result of adding two doubles: " + resultDoubleTwo);
30
31        double resultDoubleThree = calculator.add(1.5, 3.5, 3.5);
32        System.out.println("Result of adding three doubles: " + resultDoubleThree);
33    }
34 }
```

Problems Javadoc Declaration Console X

<terminated> Calculator [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (Dec 21, 2023, 2:21:40 PM – 2:21:42 PM) [pid: 19]

Result of adding two doubles: 7.2

Result of adding three doubles: 8.5

### Section -2

#### Case Study

Online ordering has enabled many restaurants to manage their peak business hours very effectively. Thanks to online ordering, many people manage to prevent the painful experience of wasting time in a long queue.

AZA is one of the biggest restaurant chains in the United Kingdom. They decided to offer their customers a convenient and contactless mobile ordering solution in response to the growing business need and COVID restrictions.

As part of the development team in the SAS software solutions, you are required to design and develop the mobile ordering program.

Before the application release deadline, you are required to submit the following deliverables:

1. Java program for the two tasks as per the description that follows in this document.

#### Program Description:

In this program, you have to create a signup process for the mobile ordering application for a restaurant. The restaurant has a variety of cuisines to offer their customers.

When the program starts, the user is given the following options:

1. Sign up
2. Quit Application

Output:

Please enter 1 for Sign up.  
Please enter 2 for Quit.

Your program should keep running and enable multiple users to sign up until the quit application option is selected.

Output: If User enters 3

Thank you for using the Application.



## Criteria

To start using the mobile app, users should sign up for an account. You are required to create a Java program for the signup process. The users will be asked to enter their full name, contact number, date of birth, password, and password confirmation.

1. The signup process must not be successful until:
  - a. The full name is enter. The length must be greater than four.
  - b. The mobile number has 10 digits starting with 0.
  - c. The Password must initiate with alphabets followed by either one of @, & and ending with numeric.  
(For Example: John@0125 or John&25) .
  - d. The password confirmation matches the initial entered password.
  - e. The DOB is in the format DD/MM/YYYY or MM/DD/YYYY.
  - f. The user is at least 21 years old. The age should be calculated based on the year entered in the DOB (Only consider year).
2. If any of the above-mentioned conditions is not fulfilled; the sign-up process should fail, and a descriptive message should be displayed for the user explaining what has gone wrong and providing hints on the correct expected input. The program should keep asking the user to re-enter his details as long as one or more of the input fields are not correctly entered. If all fields are entered successfully, the program should stop asking the user to re-enter his details and display a message that the signup process has been completed successfully.
3. If any field is entered incorrectly, some examples of sample outputs are given below.

Output 1:

You have entered the Date of Birth in invalid format.  
Please start again.

Output 2:

Your passwords are not matching.  
Please start again.

4. If all of the above-mentioned conditions are successful, the user data is saved in appropriate data structure (Hint: Arrays can be used) to enable data checks during the login process in future builds.

Output 1:

Please enter your full name: Sam Jahn

Please enter your mobile number (username): 0445544455

Please enter your password: John@21

Please confirm your password: John@21

Please enter your Date of Birth #DD/MM/YYYY (No space): 21/01/1984

You have successfully signed up.

Please enter 1 for Sign up.

Please enter 2 for Quit.

```
Test.java  Operation.java  Value.java  Logical.java  Compare.java  Concat.java  Equal.java  Length.java
1 import java.util.Scanner;
2 public class MobileOrdering {
3     public static void main(String[] args) {
4         Scanner scanner = new Scanner(System.in);
5         while (true) {
6             System.out.println("Please enter 1 for Sign up.");
7             System.out.println("Please enter 2 for Quit.");
8
9             int choice = scanner.nextInt();
10            scanner.nextLine();
11
12            if (choice == 1) {
13                signUpProcess(scanner);
14            } else if (choice == 2) {
15                System.out.println("Thank you for using the Application.");
16                break;
17            } else {
18                System.out.println("Invalid input. Please enter 1 or 2.");
19            }
20        }
21        scanner.close();
22    }
23
24    private static void signUpProcess(Scanner scanner) {
25        String fullName, contactNumber, password, confirmPassword, dob;
26        while (true) {
27            System.out.println("Please enter your full name:");
28            fullName = scanner.nextLine();
29            System.out.println("Please enter your mobile number (username):");
30            contactNumber = scanner.nextLine();
31            System.out.println("Please enter your password:");
32            password = scanner.nextLine();
33            System.out.println("Please confirm your password:");
34            confirmPassword = scanner.nextLine();
35            System.out.println("Please enter your Date of Birth #DD/MM/YYYY (No space):");
36            dob = scanner.nextLine();
37        }
38    }
39 }
```

```
TestJava  OperationJava  ValueJava  LogicalJava  CompareJava  ConcatJava  EqualJava  LengJava  MaxJava  AgeJava  GreatJava  MinJava
39         if (validateInput(fullName, contactNumber, password, confirmPassword, dob)) {
40             System.out.println("You have successfully signed up.");
41             break;
42         }
43     }
44 }
45
46 private static boolean validateInput(String fullName, String contactNumber, String password, String confirmPassword, String dob) {
47     if (fullName.length() <= 4) {
48         System.out.println("Full name must be greater than four characters. Please start again.");
49         return false;
50     }
51
52     if (!contactNumber.startsWith("0") || contactNumber.length() != 10) {
53         System.out.println("Mobile number must start with 0 and have 10 digits. Please start again.");
54         return false;
55     }
56
57     if (!password.matches("[a-zA-Z]+[@&]\\d+$")) {
58         System.out.println("Password must start with alphabets, followed by @ or & and end with numeric. Please start again.");
59         return false;
60     }
61
62     if (!password.equals(confirmPassword)) {
63         System.out.println("Your passwords are not matching. Please start again.");
64         return false;
65     }
66
67     if (!dob.matches("\\d{2}/\\d{2}/\\d{4}")) {
68         System.out.println("You have entered the Date of Birth in invalid format. Please start again.");
69         return false;
70     }
71 }
72 }
```

```
72
73     int birthYear = Integer.parseInt(dob.substring(6));
74     int currentYear = java.time.Year.now().getValue();
75     if ((currentYear - birthYear) < 21) {
76         System.out.println("You must be at least 21 years old. Please start again.");
77         return false;
78     }
79     // If all conditions are met, return true
80     return true;
81 }
82 }
```

Output:

```
Please enter 1 for Sign up.
Please enter 2 for Quit.
1
Please enter your full name:
Merina Shrestha
Please enter your mobile number (username):
987654321
Please enter your password:
merina@1
Please confirm your password:
merina@2
Please enter your Date of Birth #DD/MM/YYYY (No space):
12/12/2000
Mobile number must start with 0 and have 10 digits. Please start again.
Please enter your full name:
Merina Shrestha
Please enter your mobile number (username):
0987654321
Please enter your password:
merina@1
Please confirm your password:
merina@2
Please enter your Date of Birth #DD/MM/YYYY (No space):
12/12/2000
Your passwords are not matching. Please start again.
Please enter your full name:
Merina Shrestha
Please enter your mobile number (username):
0987654321

Please enter your password:
merina@1
Please confirm your password:
merina@1
Please enter your Date of Birth #DD/MM/YYYY (No space):
12/12/2000
You have successfully signed up.
Please enter 1 for Sign up.
Please enter 2 for Quit.
2
Thank you for using the Application.
```