

Objectives:

- Write Programs that make use of Inheritance, constructors, overriding.

**Exercise 1 (Time: 5 minutes)**

- a) Create a class called **Student**. Student has a name, ditno and an address. Use appropriate data types also.
- b) Create another class called **Test**, create two students objects from students class, and assign different values to the attributes.
- c) Display the values.

**Exercise 2 (Time: 5 minutes)**

- d) Modify the previously written **Student** class as follows:
  - a) Add a constructor to initialize all three variables.
  - b) Then create objects using the above constructor.

**Exercise 3 (Time: 5 minutes)**

- a) Modify the **Student** class and add 3 Mutator methods and 3 Accessor methods as given below to implement data hiding.
- b) For proper data hiding make attributes private.
  - ☐ `getName()`, `getAddress()` and `getDit()` that return Strings.
  - ☐ `setName()`, `setAddress()` and `setDit()` to provide values as mutator methods.
- c) Use those methods in the **Test** class.

**Exercise 4 (Time: 5 minutes)**

- a) Modify the Student class and add a method called `getDetails()` which returns a String.
- b) This method will return the details of the student as below:  

```
I am a Student.  
My name is Udaya.  
I am from Malabe.  
My dit no is DIT/11/C1/0010
```
- c) Call `getDetails()` method in the Test class.

**Exercise 5 (Time: 10 minutes)**

- a) Write a class called **Person**.
- b) A person has `name` and `address` as their attribute.
- c) Also, person class has a method call `showDetails()` which shows the `name` and the `address` of a person.
- d) Create an object of **Person** class inside **InheritanceDemo** class and call the `showDetails()` method.

**Exercise 6 (Time: 10 minutes)**

- e) Write a class called **Student**, which is a child of **Person** class.
- a) A Student has a `studentid` as a special attribute.
- f) Create an object of **Student** class inside **InheritanceDemo** class and call `showDetails()` method.

**Exercise 7 (Time: 10 minutes)**

- a) Write a class called **PartTimeStudent**, which is a child of **Student** class.
- b) A **PartTimeStudent** has number of working hours as a special attribute.
- c) Create an object of **PartTimeStudent** class inside **InheritanceDemo** class and call `showDetails()` method.

**Example 8 (Time: 10 minutes)**

- a) Try the following program (Note : You can access all three classes at <https://goo.gl/WZ68mt>)

```
public class Pet {
    private String name;
    private String owner;
    private int age;
    public Pet(String n, String o, int a) {
        this.name = n;
        this.owner = o;
        this.age = a;
    }
    public void showDetails(){
        System.out.println("I am a pet. My name is "
            +this.name+". My owner is "+this.owner);
    }
} //end of the pet class
```

- a) Take a new class and type the following program as well. This shows the inheritance:

```
public class Cat extends Pet{
    private int livesLeft;
    public Cat(String n, String o, int a,
               int l) {
        super(n, o, a);
        this.livesLeft = l;
    }
} //end of the pet class
```

- b) Create another class, which demonstrate the above two classes as below.

```
public class Main {
    public static void main(String[] args){
        Pet p = new Pet("Lissie","Smith",3);
        p.showDetails();
        Cat c = new Cat("Kyan", "Silva", 4, 4);
        c.showDetails();
    }
} //end of the demo class
```

**Exercise 9 (Time: 10 minutes)**

- a) Overload the constructors in **Pet** class and in **Cat** class
- b) Add a constructor to **Pet** class as given below to create a newborn pet object.  

```
public Pet (String n, String o)
```
- c) The age of a newborn pet is 0.
- d) Add a constructor to **Cat** class as given below to create a newborn cat object.  

```
public Cat (String n, String o)
```
- e) Age of a newborn cat will be 0 and it will have 7 lives to live.
- f) Create two more objects (a **Pet** object and **Cat** Object) in the **Main** class and call `showDetails()` method.

**Exercise 10 (Time: 10 minutes)**

- a) Write a class called **Dog**, which is a child of **Pet** class.
- b) **Dog** class has one special attribute as follows.  

```
int noOfMasters
```
- a) Overload the constructor as given below.  

```
Dog (String n, String o, int a, int m)
```
- b) All the values are given.  

```
Dog (String n, String o)
```
- c) Creates a newborn **Dog** object. The age of a newborn dog will be 0 and it will have 1 master.
- d) Create two **Dog** objects inside the **Main** class using two constructors and call `showDetails()` method.

---

**Example 11 (Time: 10 minutes) – Method Overriding**

- a) Refer how the method `showDetails()` has been overridden in **Cat** class. Try that out.

```
public class Cat extends Pet{
    private int livesLeft;
    public Cat(String n, String o, int a, int l) {
        super(n, o, a);
        this.livesLeft = l;
    }
    public Cat(String n, String o) {
        super(n, o);
        this.livesLeft = 7;
    }
    public void showDetails(){
        super.showDetails();
        System.out.println("I am a cat. " +
            this.livesLeft + " lives remain for me.");
    }
} //end of the cat class
```

- a) Try the following code inside **Main** class and observe the result. Try to understand why you get that result.

```
public class Main {
    public static void main(String[] args){
        .....//fill the code
        .....
        Pet p3 = c;
        p3.showDetails(); } //end of demo class
```

**Exercise 10 (Time: 10 minutes)**

- a) Override `showDetails()` method inside `Dog` class. It should produce an output as follows.

```
I am a pet. My name is Syndy. My owner is Nimal.  
I am a dog. I have 2 masters at home.
```