National University of Computer and Emerging Sciences



**Laboratory Manual**

***(OOAD)***

|  |  |
| --- | --- |
| Semester | Fall 2018 |

Department of Computer Science

FAST-NU, Lahore

Keep the following good programming practices in mind when writing your code:

* Comment your code intelligently.
* Indent your code properly.
* Use meaningful variable names.
* Use meaningful prompt lines/labels for input/output.
* use meaningful project and JAVA file name

**Task 1 (Practice)**

Create a project name as “Lab\_2”, create a package name as “Inheritance” by right clicking the project name. Add JAVA files Bicycle, MountainBike and driver in Inheritance package. Code of these JAVA files are given in manual folder. Run driver class understand the working and syntax.

**Task 2**

We are going to implement an Employee management system for a company. There are two types of employees exist in a company. One is contract based employees and other type is permanent employees. System must save following details in system.

* CNIC
* Name
* Date of joining (Permanent employees)
* Date of resign (Permanent employees)
* Contract Signed and Expired Date (Contract base employees)
* Employee Grade e.g. 1 up to 6
* Employee Designation
* Salary basic

Salary of employees must be computed on the basis of given conditions.

Contract based employees’ salary will be computed after deducting 17% tax of its basic salary.

1. Implement the inheritance hierarchy.
2. Implement a function called EmployeeDetails. This function will show the employee type and its details as mentioned above.
3. Print employee detail and salary for each employee using reference of base class.
4. Your output should be properly formatted.
5. Do not store computed values

**Task 3**

Suppose we are developing a program that compute the tax of vehicles.

There is top class name vehicle: which has a **compute Tax** function, and a **Print Vehicle** function. No return type, no parameter in both functions

**Types of vehicles**

There are two type of vehicle

**HTV: heavy travelling vehicles**

**LTV: Light travelling vehicles**

* HTV class has data member name as Fix Tax, which will be used while computing the tax of heavy travelling vehicles.

Value of fix tax will be passed whenever a HTV type object is created.

* LTV class has thee slabs for different **Cubic Centimeters (CC)** of Light travelling vehicles.

It has three integer data member, X , Y, Z. Tax rates.

Value of x, y and z will be passed whenever a LTV type object is created

**Vehicles**

You have following vehicles:

* **Car:** A car has its makers, year model, name and CC, a **Print Vehicle** function, **compute Tax** function.

If CC <= 1000 Tax = x

If 1000 < CC <= 2000 Tax = y

If CC > 2000 Tax = z

* **Bike:** A bike has its makers, year model and CC, a **Print Vehicle** function , **compute Tax** function

If CC <= 75 Tax = x

If 1000 < CC <= 100 Tax = y

If CC > 250 Tax = z

* **Bus:** A bus has its number Of seats, a **Print Vehicle** function , **compute Tax** function, tax of bus will be associated with number of seats of bus, you have a per seat tax which is Rs 20, it will be applied when number of seats will be greater than 20. Fix tax of HTV will be added in final tax
* **Truck:** A truck has its load capacity, a **Print Vehicle** function , **print Tax** function, tax of truck will be associated with load capacity of truck, you have per KG tax which is Rs 100, it will be applied when load capacity will be greater than 1000 KG. Fix tax of HTV will be added in final tax. Then prints the final tax.

Sample main program:

|  |
| --- |
| public static void main(String args[])  {  **//Last there parameters are x,y,z for LTV class**  Car c1 = new Car("Honda" , 2012, "City" , 1500, 1000,1500,2000);  Car c2 = new Car("Mercedes" , 2016 , "Kompressor", 22000 , 1000,1500,2000);  Car c3 = new Car("Suzuki" , 2016 , "Wagon\_r", 8000, 1000,1500,2000);  Bike b1 = new Bike("Honda" , 2012, 100, 50,200,300);  Bike b2 = new Bike("Uniqe" , 2016 , 70, 50,200,300);  Bike b3 = new Bike("Suzuki" , 2016 , 125, 50,200,300);  **//second parameter is Fix tax for HTV class**  Truck t1 = new Truck (1500 , 5000);  Truck t2 = new Truck (1500 , 5000 );  Truck t3 = new Truck (800 , 5000);  Bus bb1 = new Bus (100, 5000);  Bus bb2 = new Bus (70, 5000);  Bus bb3 = new Bus (125, 5000);  **//Use Polymorphism by storing a reference of sub class in super class to execute the correct function call based on type of the object referenced at run time.**  }  } |
|  |