DAY 2

1. Class And Object :

- Box.Java

```
package day2.classandobject;

public class Box

{
	int length;
	int width;

	/*int calculateArea() // returning a value in main method
	{
		int area = length * width;
		System.out.println("Area = " +area );
		return area;
	}*/


	/*int calculateArea(int x) // passing single parameters/arguments
	{
		int area = length * width * x;
		System.out.println("Area = " +area );
		return area;
	}*/


	/*int calculateArea(int length , int width) // passing method level
arguments
	{
		int area = length * width;
		System.out.println("Area = " +area );
		return area;
	}*/


	int calculateArea(int length , int width) // passing class level arguments
using this keyword
	{
		int area = this.length * this.width;
		System.out.println("Area = " +area );
		return area;
	}
}
```

- Employee.Java

```
package day2.classandobject;

public class Employee

{
	double salary;
	double bonus;

	void calculateTotalPay()   //not returning any value in main method
	{
		double totalpay = salary + bonus;
		System.out.println("Total Pay = " +totalpay);
	}
}
```

- TestBox.Java

```
package day2.classandobject;

public class TestBox {

	public static void main(String[] args)

	{

		Box ups = new Box();
		Box fedex = new Box();

		/*ups.length =5;
		ups.width =10;
		int area1 = ups.calculateArea();
										  //
returning a value in main method
		fedex.length = 7;
		fedex.width =9;
		int area2 = fedex.calculateArea();*/



		/*ups.length =5;
		ups.width =10;
		int area1 = ups.calculateArea(4);
										  // passing
single parameters/arguments
		fedex.length = 7;
		fedex.width =9;
		int area2 = fedex.calculateArea(5);*/
		/*ups.length =5;
		ups.width =10;
		int area1 = ups.calculateArea(4,5);                                                                                     // passing
method level arguments
		fedex.length = 7;
		fedex.width =9;
		int area2 = fedex.calculateArea(5,7);*/



		ups.length =5;
		ups.width =10;
		int area1 = ups.calculateArea(4,5);                                                                                     // passing
class level arguments using this keyword
		fedex.length = 7;
		fedex.width =9;
		int area2 = fedex.calculateArea(5,7);

		System.out.println("Area of ups & fedex = " + (area1 + area2));

	}

}
```

- TestEmployee.Java

```
 package day2.classandobject;

public class TestEmployee {

	public static void main(String[] args)

	{

		Employee alex = new Employee();
		Employee sushant = new Employee();
		Employee vivek = new Employee();

		alex.salary = 90000;
		alex.bonus = 10000;
		alex.calculateTotalPay();

		sushant.salary =2000000;
		sushant.bonus =300000;
		sushant.calculateTotalPay();

		vivek.salary = 40000;
		vivek.bonus = 20000;
		vivek.calculateTotalPay();

	}
}
```

2. Constructors

- SmallBox.Java

```
package day2.constructors;

public class SmallBox

{
	int length;
	int width;

	//Constructor : Is a method which has a same name as the class
	//It is executed when an object is created
	//It is used to set default values
	//does not return anything including void

	SmallBox()
	{
		this.length = 5;					// this is 
constructor
		this.width = 7;
		System.out.println("Constructor fired");
	}


	SmallBox(int length , int width)

	{
		this.length = length;
		this.width = width;
		System.out.println("Parameterized constructor fired");
	}


	void calculateArea()
	{
		System.out.println("Area = " + (length*width));
	}
}
```

- TestSmallBox.Java

```
package day2.constructors;

public class TestSmallBox

{

	public static void main(String[] args)

	{
		SmallBox obj = new SmallBox();
		obj.calculateArea();


		SmallBox ups = new SmallBox(4,3);
		ups.calculateArea();

	}

}
```

3. OOP

I. Data Hiding

- Employee.Java

```java
package day2.oop.datahiding;

public class Employee

{
	private double salary;
	double bonus;

	void setSalary(double salary)
	{
		if(salary >= 40000 && salary <= 200000)
		{
			this.salary = salary;
		}

		else
		{
			this.salary = 0;
			System.out.println("Please check salary");
		}




	}

	public double getSalary()
	{
		return salary;
	}

	void calculateTotalPay()
	{
		double totalpay = salary + bonus;
		System.out.println("Total Pay : " + totalpay);
	}
}
```

- TestEmployee.Java

```java
package day2.oop.datahiding;

public class TestEmployee {

	public static void main(String[] args)

	{

		Employee sushant = new Employee();

		sushant.setSalary(100000);
		sushant.bonus = 20000;
		sushant.calculateTotalPay();

		System.out.println("Sushant's salary : " + sushant.getSalary());

	}

}
```

II. Overloading

- Box.Java

```java
package day2.oop.overloading;

public class Box

// Polymorphism :
// When method of same name is differentiated by their passing arguments

{
	void calculateArea(int length) {
		System.out.println("Area = " +(length * length));
	}


	void calculateArea(double length) {
		System.out.println("Area = " +(length * length));
	}


	void calculateArea(int length , int width) {
		System.out.println("Area = " +(length * width));
	}
}
```

- TestBox.Java

```
package day2.oop.overloading;

public class TestBox {

	public static void main(String[] args)

	{

		Box obj = new Box();
		obj.calculateArea(3); // passing integer argument
		obj.calculateArea(3.7); // passing double argument
		obj.calculateArea(2, 3); // passing two integer argument

	}

}
```

4. SampleProject

- Department.Java

```
package day2.sampleproject;

public class Department

{
	private String deptName;
	private double budget;

	private Employee[] emps = new Employee[5]; // for an association of
employees with each department we are creating this array of employess
	private int counter = 0;



	public Department(String deptName)
	{
		this.deptName = deptName;
		this.budget = 50000;		// we are not passing budget in
constructor because our budget is fix for each department i.e. 50k
	}




	public void addEmployee(Employee obj)
	{
		emps[counter] = obj;
		counter++;


		if(obj.getGrade() >= 5)
		{
			this.budget += 150000;
		}
		else
		{
			this.budget += 100000;
		}

	}
							//addEmployee() method


		public void describe()
		{
			String temp = "Dept Name : " + this.deptName
					+ "\nBudget : " + this.budget
					+ "\nEmployees : ";
```

```java
				for(Employee x : emps)
				{
					if(x != null)
					{
					temp += x.getEmployeeInfo() + " ";
					}
				}
				System.out.println(temp);
			}
									//describe() method
		}
```

- Employee.Java

```java
package day2.sampleproject;

public class Employee

{
	private String empname;	//since the data is private you have to either
create a constructor or create a setter method to pass arguments in main class
	private int grade;


	public int getGrade()
	{
		return grade;
	}


	public Employee(String empname,int grade)
	{
		this.empname = empname;
		this.grade = grade;
	}


	public String getEmployeeInfo()
	{
		return empname + "(" +grade+ ")";
	}
}
```

- TestCompany.Java

```java
package day2.sampleproject;
```

```java
public class TestCompany {

       public static void main(String[] args)

       {
       Employee alex = new Employee("Alex Rod" , 6);
       Employee linda = new Employee("Linda Berry" , 7);
       Employee john = new Employee("John Doe" , 3);
       Employee sara = new Employee("Sara Time" , 7);
       Employee james = new Employee("James Doe" , 4);

       Department sales = new Department("XYZ Sales");
       Department it = new Department("XYZ IT");


       sales.addEmployee(alex);
       sales.addEmployee(linda);
       sales.addEmployee(john);

       it.addEmployee(sara);
       it.addEmployee(james);


       sales.describe();
       System.out.println("---------------------------");
       it.describe();

       }

}
```