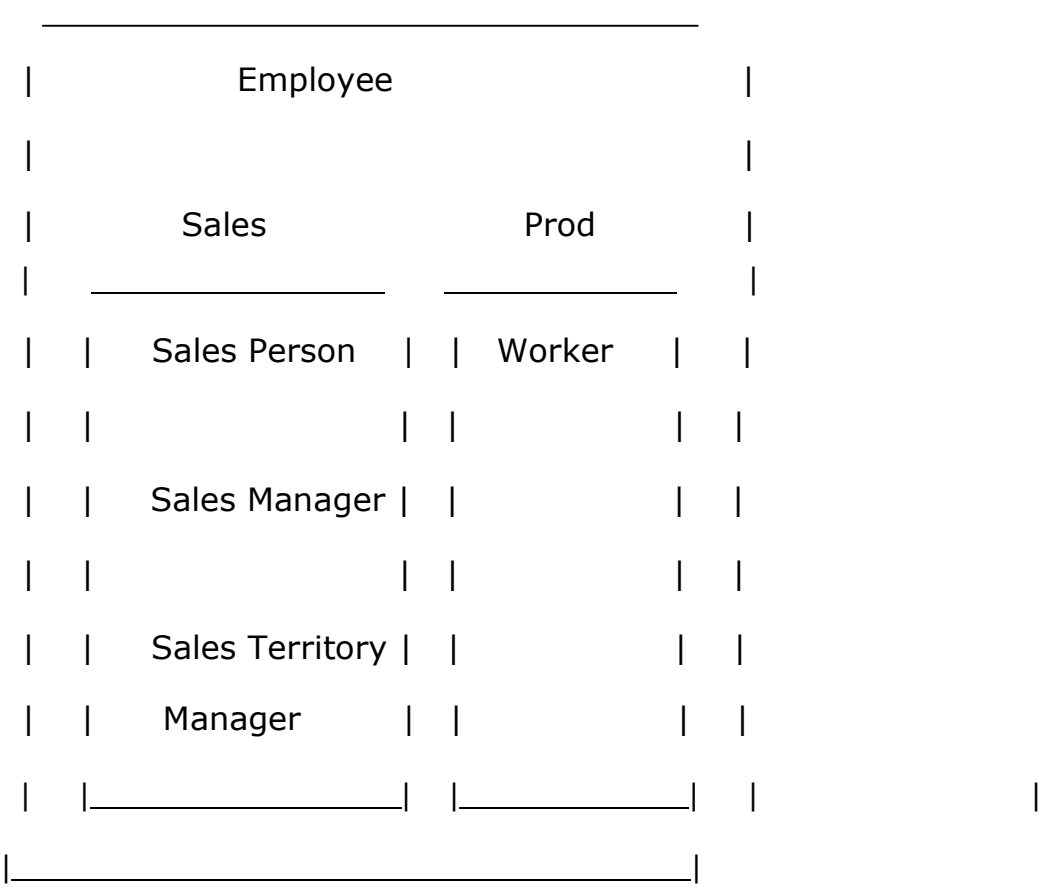
(interface and packages)

1. Write pay Roll system with following package Structure.



2. Write program to calculate Area and volume depending upon type of figure by implementing two interfaces for CalcArea and CalcVolume

Figure	Area	Perimeter	Surface Area	Volume

	pi*r*r	2*pi*r		
Circle				
Square	a*a	4*a		
Tringle	s = (a+b+c)/2	s1+s2+s3		
	area = $\sqrt{[s(s-a)(s-b)(s-c)]}$			
Sphere	pi*r*r	2*pi*r	4*pi*r*r	(4/3)pi*r*r*r
Cuboid	a*a	4*a	6*a*a	a*a*a

3. Write a program in which stack interface with two methods push() and pop() will be implemented by classes "FixedStack" and "Dynamic Stack".
4. Create 3 interfaces each having two methods. Create a new interface extending these 3 interfaces. Add a new method to the interface. Create a class implementing the new interface and also inheriting from class. Write four methods each taking one of the four interfaces as an argument. In main() create a object of your class and pass it to each of these methods.

5. Create an interface with 3 methods in a package. Implement this in a class in different package
6. Create a class in a package. Write a protected method in it. From outside the package, try to call the method and see the result. Now inherit from your class and call the protected method from inside a method of your derived class
7. Create an abstract superclass named four-wheeler and a concrete subclass named car. The superclass should belong to a package called vechicle and the subclass can belong to the default package (meaning it isn't put into a package explicitly). Make the superclass public and give the subclass default access. Compile these 2 files successfully

(exceptionhandling)

1. Create a class with a method throwing an object of class IOException. Neither catch this object nor use throws clause. Test and see the result
2. Try the following and observe the result. Also try it with a statement throwing some exception in line 1

```
class sample{ public
voidmth1()
{
mth2(); System.out.println("caller");
}
public void mth2()
{
try
{
// line 1 return;
```

```
}
catch(Exception e){ System.out.println("catch-mth2");}
```

```
finally{System.out.println("finally-mth2");}
}
public static void main(Stringp[])
{
sample s=new
sample();
s.mth1();
}
}
```

3. try the above by replacing the return statement inside try block by System.exit(0);
4. Create your own exception. In main() create a try-catch clause to exercise your new exception
5. Write a class with a method that throws an exception of the type created in previous exercise. Compile this without catch the exception
6. Write a program to check whether a derived class constructor can catch exception thrown by base class constructor.
7. Write a program such that the user is repeatedly asked for the numerator and the divisor. For each set of data, the program prints out the result or an informative error message if there is a problem (division by zero or poor input data).
    a. The program continues looping, even if there is a problem

b. Exit the loop when data entered for the numerator start with characters "q" or "Q". Don't print out an error message in this case.

8. Write a Program to take care of Number Format Exception if user enters values other that integer for calculating average of marks ten students. Do not allow program to terminate but again continue with the program after giving respective message to User.

9. In the same Program write your own Exception classes to take care of Negative values and values out of range (i.e. other than in the range of 0-100) and do not allow program to terminate give the message depending upon the wrong input submitted by the user.