|  |  |
| --- | --- |
| File:COMSATS new logo.jpg - Wikimedia Commons  **interface - lab** | **Subject:**  **object oriented programming**  **submitted by:**  **Daoud Hussain**  (Sp21-bcs-102)  **Class:**  **bcs-3b**  **submitted to:**  **mam saneeha amir**  **date of submission:**  **april 05 , 2022** |

1. Shape interface

public interface Shape{

double getArea();

}

1. Circle Class

public class Circle implements Shape{

double radius;

final double PI = 3.1415;

public Circle(double r){

radius = r;

}

public double getArea(){

return PI\*radius\*radius;

}

}

1. Rectangle Class

public class Rectangle implements Shape{

double length;

double height;

public Rectangle(double b, double h){

length = b;

height = h;

}

public double getArea(){

return length\*height;

}

}

1. Runner Shape interface

public class Runner{

public static void main(String[] args) {

Shape[] shp = new Shape[2];

shp[0] = new Circle(5);

shp[1] = new Rectangle(3, 5);

for(int i = 0; i < shp.length; i++){

System.out.println(shp[i].getArea());

}

}

}

2. Payable interface

public interface Payable{

double getPaymentAmount();

}

2. Invoice Class

public class Invoice implements Payable{

private String partNumber;

private String partDescription;

private int quanity;

private double pricePerItem;

public Invoice(String pNumber, String pDes, int quan, double ppi){

partDescription = pDes;

partNumber = pNumber;

quanity = quan;

pricePerItem = ppi;

}

public double getPaymentAmount(){

return quanity \* pricePerItem;

}

}

2. Employee Class

public abstract class Employee implements Payable{

private String firstName;

private String lastName;

private String socialSecurityNumber;

public Employee(String fName, String lName, String ssn){

firstName = fName;

lastName = lName;

socialSecurityNumber = ssn;

}

}

2. SalariedEmployee Class

public class SalariedEmployee extends Employee implements Payable{

private double weeklySalary;

public SalariedEmployee(String fName, String lName, String ssn, double ws){

super(fName, lName, ssn);

weeklySalary = ws;

}

public double getPaymentAmount(){

return weeklySalary;

}

}

2. Runner Payable Interface

public class Runner{

public static void main(String[] args) {

Payable[] pb = new Payable[2];

pb[0] = new Invoice("10", "Good", 3, 900);

pb[1] = new SalariedEmployee("Daoud", "Hussain", "122", 15540);

for(int i = 0; i < pb.length; i++){

System.out.println(pb[i].getPaymentAmount());

}

}

}

3. Point Cloneable Class

public class Point implements Cloneable{

//Data members

private int x;

private int y;

//Full-Argument constructor

public Point(int xVal, int yVal){

x = xVal;

y = yVal;

}

//Overriding Interface method

public Object clone(){

Point p = new Point(this.x, this.y);

return p;

}

public void move(int xVal, int yVal){

x += xVal;

y += yVal;

}

public int getX(){

return this.x;

}

public int getY(){

return this.y;

}

}

3. Point Runner

public class Runner{

public static void main(String[] args) {

Point origin = new Point(3,4);

Point p1 = (Point)origin.clone();

System.out.print(p1.getX() + " ");

System.out.print(p1.getY());

}

}