|  |  |
| --- | --- |
| File:COMSATS new logo.jpg - Wikimedia Commons  **Polymorphism** | **Subject:**  **Object Oriented Programming**  **submitted by:**  **Daoud Hussain**  (Sp21-bcs-102)  **Class:**  **bcs-3b**  **submitted to:**  **Mam Saneeha Amir**  **date of submission:**  **april 15 , 2022** |

1. Employee Class

public abstract class Employee {

protected String firstName;

protected String lastName;

protected String ssn;

//Argument-Constructors

public Employee(String fn, String ln, String secSecNum)

{

if(fn != "" && ln != "" && secSecNum != ""){

firstName = fn;

lastName = ln;

ssn = secSecNum;

}

}

//Abstract class method

public abstract double earnings();

//Method to display values

public String toString(){

String text = "First Name: " + firstName + " Last Name: " + lastName + " SSN: " + ssn;

return text;

}

}

1. Commision Employee Class

public class CommisionEmployee extends Employee {

protected int grossSales;

protected int commisionRate;

//Argument-Constructor

public CommisionEmployee(String fn, String ln, String secSecNum, int gs, int cr){

super(fn, ln, secSecNum);

grossSales = gs;

commisionRate = cr;

}

//Overriding Abstract class method earnings()

public double earnings(){

return commisionRate \* grossSales;

}

//Method to display values

public String toString(){

String text = "First Name: " + firstName + " Last Name: " + lastName + " SSN: " + ssn + " Gross Sales: " + grossSales + " Commision Rate: " + commisionRate;

return text;

}

}

1. BasePlusCommisionEmployee Class

public class BasePlusCommisionEmployee extends CommisionEmployee {

private int baseSalary;

//Argument-Constructor

public BasePlusCommisionEmployee(String fn, String ln, String secSecNum, int gs, int cr, int bs){

super(fn, ln, secSecNum, gs, cr);

baseSalary = bs;

}

//Overriding Abstract class method earnings()

public double earnings(){

return (commisionRate \* grossSales) + baseSalary;

}

//Method to display values

public String toString(){

String text = "First Name: " + firstName + " Last Name: " + lastName + " SSN: " + ssn + " Gross Sales: " + grossSales + " Commision Rate: " + commisionRate + " Base Salary: " + baseSalary;

return text;

}

}

1. Weekly Employee Class

public class WeeklyEmployee extends Employee{

private double WeeklySalary;

//Argument-Constructor

public WeeklyEmployee(String fn, String ln, String secSecNum, double ws){

super(fn, ln, secSecNum);

if(ws > 0){

WeeklySalary = ws;

}

}

//Method to calculate Weekly Salary

public double earnings(){

return WeeklySalary;

}

//Method to display values

public String toString(){

String text = "First Name: " + firstName + " Last Name: " + lastName + " SSN: " + ssn + " Weekly Salary: " + WeeklySalary;

return text;

}

}

1. Hourly Employee Class

public class HourlyEmployee extends Employee {

private int hoursWorked;

private int hourlyWages;

//Argument-Constructor

public HourlyEmployee(String fn, String ln, String secSecNum, int hrsWrk, int hrsWag){

super(fn, ln, secSecNum);

hoursWorked = hrsWrk;

hourlyWages = hrsWag;

}

//Overriding Abstract class method earnings()

public double earnings(){

if(hoursWorked < 40){

return hoursWorked \* hourlyWages;

}

else{

return 40 \* hourlyWages + (hoursWorked - 40) \* hourlyWages \* 1.5;

}

}

//Method to display values

public String toString(){

String text = "First Name: " + firstName + " Last Name: " + lastName + " SSN: " + ssn + " Hourly Wages: " + hourlyWages + " Hours Worked: " + hoursWorked;

return text;

}

}

1. Runner Employee class

public class Runner{

public static void main(String[] args) {

Employee[] emp = new Employee[4];

emp[0] = new CommisionEmployee("Daoud", "Hussain", "345924", 1270, 1500);

emp[1] = new WeeklyEmployee("Daoud", "Hussain", "345924", 8000);

emp[2] = new HourlyEmployee("Daoud", "Hussain", "345924", 9, 40);

emp[3] = new BasePlusCommisionEmployee("Daoud", "Hussain", "345924", 1270, 1500, 25000);

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

System.out.println(emp[i].toString());

emp[i].earnings();

}

}

}

2. Package Class

public abstract class Package{

//Data Members

protected String name;

protected String address;

protected double weightPerOunce;

protected double costPerOunce;

//Full-Argument Constructor

public Package(String n, String add, double wpo, double cpo){

if(n != "" && add != "" && wpo != 0 && cpo != 0){

name = n;

address = add;

weightPerOunce = wpo;

costPerOunce = cpo;

}

}

public abstract double calculateCost();

}

2. OverNightPackage Class

public class OverNightPackage extends Package{

private int additionalFee;

//Full-Argumented Constructor

public OverNightPackage(String n, String add, double wpo, double cpo, int ff){

super(n, add, wpo, cpo);

additionalFee = ff;

}

//Over-riding method in child class

public double calculateCost(){

return (costPerOunce \* weightPerOunce) + additionalFee ;

}

}

2. TwoDayPackage Class

public class TwoDayPackage extends Package{

private int flatFee;

//Full-Argumented Constructor

public TwoDayPackage(String n, String add, double wpo, double cpo, int ff){

super(n, add, wpo, cpo);

flatFee = ff;

}

//Over-riding method in child class

public double calculateCost(){

return (costPerOunce \* weightPerOunce) + flatFee ;

}

}

2. Runner Package Class

public class Runner{

public static void main(String[] args) {

Package[] p = new Package[2];

p[0] = new OverNightPackage("Daoud", "Park-Road Islamabad", 20.4, 9.0, 20000);

p[1] = new TwoDayPackage("Hussain", "Hostel-City Islamabad", 20.4, 9.0, 1500);

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

System.out.println(p[i].calculateCost());

}

}

}

3. Movie Class

public abstract class Movie{

protected String movieName;

protected int ID;

protected int numberOfDays;

//Setters

public void setMovieName(String mn){

if(mn != ""){

movieName = mn;

}

}

public void setID(int id){

if(id != 0){

ID = id;

}

}

public void setNumberOfDays(int ndys){

if(ndys != 0){

numberOfDays = ndys;

}

}

//Getters

public String getMovieName(){

return movieName;

}

public int getID(){

return ID;

}

public int getNumberOfDays(){

return numberOfDays;

}

public boolean equals(Movie m){

if(this.movieName == m.movieName && this.ID == m.ID && this.numberOfDays == m.numberOfDays ){

return true;

}

return false;

}

public abstract double calculateFees();

}

3. Drama Class

import java.util.\*;

public class Drama extends Movie{

public double calculateFees(){

Scanner input = new Scanner(System.in);

System.out.print("Enter the number of movie late days: ");

int days = input.nextInt();

return days \* 2;

}

}

3. Comedy Class

import java.util.\*;

public class Comedy extends Movie{

public double calculateFees(){

Scanner input = new Scanner(System.in);

System.out.print("Enter the number of movie late days: ");

int days = input.nextInt();

return days\*2.50;

}

}

3. Runner Movie Class

public class Runner{

public static void main(String[] args) {

Movie[] mv = new Movie[2];

mv[0] = new Comedy();

mv[1] = new Drama();

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

System.out.println(mv[i].calculateFees());

}

System.out.print(mv[0].equals(mv[1]));

}

}

4. Person Class

public abstract class Person{

//Data Members

protected String name;

//Setter

public void setName(String n){

if(n!=""){

name = n;

}

}

//Getters

public String getName(){

return name;

}

public abstract boolean isOutstanding();

}

4. Student Class

public class Student extends Person{

double cgpa;

//Setter of name in child class

public void setCGPA(String nam, double cgpaStu){

super.setName(nam);

if(cgpaStu!=0){

cgpa = cgpaStu;

}

}

//Getters

public double getcgpa(){

return cgpa;

}

//Over-riding method in child class

public boolean isOutstanding(){

if(cgpa > 3.5){

return true;

}

return false;

}

}

4. Professor Class

public class Professor extends Person{

int numberOfPublications;

//Setter

public void setNumberOfPublications(String nam, int n){

super.setName(nam);

if(n!=0){

numberOfPublications = n;

}

}

//Getters

public int getNumberOfPublications(){

return numberOfPublications;

}

//Over-riding method in child class

public boolean isOutstanding(){

if(numberOfPublications > 50){

return true;

}

return false;

}

}

4. Runner Person Class

public class Runner{

public static void main(String[] args) {

Student stu = new Student();

stu.setCGPA("Daoud", 3.04);

Professor pro = new Professor();

pro.setNumberOfPublications("Daoud", 100);

Person[] per = new Person[2];

per[0] = stu;

per[1] = pro;

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

System.out.println(per[i].isOutstanding());

}

}

}

5. Convert Class

public abstract class Convert{

//Data Members

protected double val1;

protected double val2;

//One-Argument-Constructor

public Convert(double v1){

if(v1 != 0){

val1 = v1;

}

}

public void setVal2(double v){

val2 = v;

}

public abstract double compute();

}

5. FahrenheitToCelcius Class

public class FahrenheitToCelcius extends Convert{

//One-Argument-Constructor

public FahrenheitToCelcius(double v1){

super(v1);

}

//Over-riding method in child class

public double compute(){

double res = (val1 - 32)/1.8000;

super.setVal2(res);

return val2;

}

}

5. LitersToGallons Class

public class LitersToGallons extends Convert{

//One-Argument-Constructor

public LitersToGallons(double v1){

super(v1);

}

//Over-riding method in child class

public double compute(){

double res = val1 \* 0.264172 ;

super.setVal2(res);

return val2;

}

}

5. FeetToMeters Class

public class FeetToMeters extends Convert{

//One-Argument-Constructor

public FeetToMeters(double v1){

super(v1);

}

//Over-riding method in child class

public double compute(){

double res = val1 \* 0.3048 ;

super.setVal2(res);

return val2;

}

}

5. Runner Convert Class

public class Runner{

public static void main(String[] args) {

Convert[] cvt = new Convert[3];

cvt[0] = new FahrenheitToCelcius(10);

cvt[1] = new LitersToGallons(10);

cvt[2] = new FeetToMeters(10);

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

System.out.println(cvt[i].compute());

}

}

}