**Java Core Assignment – 2 -Tanaya Jadhav**

1)

class Singleton  
{  
 private static Singleton *single\_instance*=null;  
  
 public String s;  
  
 private Singleton()  
 {  
 s = "Hello I am a Employee from capgemini";  
 }  
  
 public static Singleton Singleton()  
 {  
 if (*single\_instance* == null)  
 {  
 *single\_instance* = new Singleton();  
 }  
 return *single\_instance*;  
 }  
}  
class Main  
{  
 public static void main(String args[])  
 {  
 Singleton x = Singleton.*Singleton*();  
 Singleton y = Singleton.*Singleton*();  
 Singleton z = Singleton.*Singleton*();  
 x.s = (x.s).toLowerCase();  
  
 System.*out*.println("From Pune, " + x.s);  
 System.*out*.println("From Mumbai ," + y.s);  
 System.*out*.println("From banglore, " + z.s);  
 System.*out*.println("\n");  
 z.s = (z.s).toUpperCase();  
  
 System.*out*.println("From Pune, " + x.s);  
 System.*out*.println("From Mumbai ," + y.s);  
 System.*out*.println("From banglore, " + z.s);  
 }  
}

Output:=

"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\lib\idea\_rt.jar=59139:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\Tanaya\IdeaProjects\Corejava\out\production\Corejava Main

From Pune, hello i am a employee from capgemini

From Mumbai ,hello i am a employee from capgemini

From banglore, hello i am a employee from capgemini

From Pune, HELLO I AM A EMPLOYEE FROM CAPGEMINI

From Mumbai ,HELLO I AM A EMPLOYEE FROM CAPGEMINI

From banglore, HELLO I AM A EMPLOYEE FROM CAPGEMINI

Process finished with exit code 0

Notes:-  In the Singleton class, when we first time call Singleton() method, it creates an object of class Singleton with name single\_instance and return it to the variable. Since single\_instance is static, it is changed from null to some object. Next time if we try to call Singleton() method, since single\_instance is not null, it is returned to the variable, instead of instantiating the Singleton class again.

2)Employee.java having subclasses manager and labour

public class Employee {  
 int employeeId;  
 String employeeName;  
 double salary;  
  
 public Employee(int employeeId, String employeeName, double salary) {  
 super();  
 this.employeeId = employeeId;  
 this.employeeName = employeeName;  
 this.salary = salary;  
 }  
  
 public int getEmployeeId() {  
 return employeeId;  
 }  
 public void setEmployeeId(int employeeId) {  
 this.employeeId = employeeId;  
 }  
 public String getEmployeeName() {  
 return employeeName;  
 }  
 public void setEmployeeName(String employeeName) {  
 this.employeeName = employeeName;  
 }  
 public double getSalary() {  
 return salary;  
 }  
 public static class Manager extends Employee{  
  
 public static final double *Bonus*=0.2;  
 public Manager(int employeeId, String employeeName, double incentive) {  
 super(employeeId, employeeName, incentive);  
 }  
 public double getSalary() {  
 return salary+salary\**Bonusrate*;  
 }  
 }  
 public static class Labour extends Employee{  
 public static final double *Bonus*=0.1;  
  
 public Labour(int employeeId, String employeeName, double salary) {  
 super(employeeId, employeeName, salary);  
 }  
  
 public double getSalary() {  
  
 return salary+salary\**Bonus*;  
 }  
 }  
}

methodoverdingmain.java

public class MethodOverridingMain {  
  
 */\*\*  
 \** ***@author*** *Arpit Mandliya  
 \*/* public static void main(String[] args) {  
 Employee.Labour l1=new Employee.Labour(23,"ramesn" ,2000);  
 Employee.Labour l2=new Employee.Labour(24,"suresh" ,5000);  
 Employee.Manager m1=new Employee.Manager(1,"Amit" ,30000);  
 Employee.Manager m2=new Employee.Manager(2,"Tanaya" ,50000);  
  
 System.*out*.println("Name of Employee:" +l1.getEmployeeName()+"---"+"Salary:"+l1.getSalary());  
 System.*out*.println("Name of Employee:" +l2.getEmployeeName()+"---"+"Salary:"+l2.getSalary());  
 System.*out*.println("Name of Employee:" +m1.getEmployeeName()+"---"+"Salary:"+m1.getSalary());  
 System.*out*.println("Name of Employee:" +m2.getEmployeeName()+"---"+"Salary:"+m2.getSalary());  
 }  
}

Output:-

"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\lib\idea\_rt.jar=50029:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\Tanaya\IdeaProjects\Corejava\out\production\Corejava MethodOverridingMain

Name of Employee:ramesn---Salary:2200.0

Name of Employee:suresh---Salary:5500.0

Name of Employee:Amit---Salary:36000.0

Name of Employee:Tanaya---Salary:60000.0

Process finished with exit code 0

3)

public class BankAccount {  
 private String name;  
 protected double balance;  
  
 public String getName() {  
 return this.name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public void deposite(double amount) {  
 this.balance += amount;  
 }  
  
 public boolean withdraw(double amount) {  
 if (balance > amount) {  
 balance -= amount;  
 return true;  
 } else {  
 return false;  
 }  
 }  
 public class CurrentAccount extends BankAccount {  
  
 }  
 public static class SavingsAccount extends BankAccount {  
  
 }  
 public static class currentaccount extends BankAccount {  
  
 public boolean withdraw(double amount) {  
 balance -= amount;  
 return true;  
 }  
 }  
 }

public class Bankaccount {  
 public static void main(String[] args) {  
 BankAccount.currentaccount c = new BankAccount.currentaccount();  
 BankAccount.SavingsAccount s = new BankAccount.SavingsAccount();  
 c.deposite(500.00);  
 s.deposite(500.00);  
 *doWithdrawal*(c); // Withdraw succeed.  
 *doWithdrawal*(s); // Withdraw failed.  
 }  
  
 public static void doWithdrawal(BankAccount acc) {  
 boolean result = acc.withdraw(1000.00);  
 if (result) {  
 System.*out*.println("Withdraw succeed.");  
 } else {  
 System.*out*.println("Withdraw failed.");  
 }  
 }  
 }

"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\lib\idea\_rt.jar=53084:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\Tanaya\IdeaProjects\Corejava\out\production\Corejava BankDemo

Withdraw succeed.

Withdraw failed.

Process finished with exit code 0

4)

* A class that is declared as abstract is known as an abstract class. It needs to be extended and its method implemented. It cannot be instantiated. It can have abstract methods, non-abstract methods, constructors, and static methods. It can also have the final methods which will force the subclass not to change the body of the method. Consider the following example.
* if there is an abstract method in a class, that class must be abstract
* if there is an abstract method in a class, that class must be abstract
* Can we use abstract and final both with a method? No, because we need to override the abstract method to provide its implementation, whereas we can't override the final method.
* is it possible to instantiate the abstract class? No, the abstract class can never be instantiated even if it contains a constructor and all of its methods are implemented.
* we can not declare class as final and abstract at the same time. Final means No other class can extend that class and Abstract means Class must be extended by any other class.

5)Shape.java

abstract class Shape{  
  
 public void draw()  
 {  
 System.*out*.println("Draw method in shape class given below");  
 }  
  
 public abstract double calculateArea();  
}  
  
class Rectangle extends Shape  
{  
 double length;  
 double breadth;  
  
  
 public Rectangle(double length, double breadth) {  
 super();  
 this.length = length;  
 this.breadth = breadth;  
 }  
  
 @Override  
 public double calculateArea() {  
 return length\*breadth;  
 }  
}  
  
class Cube extends Shape  
{  
 double side;  
  
 public Cube(double side) {  
 super();  
 this.side = side;  
 }  
  
 @Override  
 public double calculateArea() {  
 return 6\*side\*side;  
 }  
}

Abstractmain.java

import java.sql.SQLOutput;  
  
public class AbstractClassMain {  
  
 public static void main(String[] args)  
 {  
 System.*out*.println("============================");  
 Rectangle rec=new Rectangle(15, 20);  
 rec.draw();  
 System.*out*.println("Area of rectangle is "+rec.calculateArea());  
 System.*out*.println("\*\*\*\*\*\*\*\*\*\*\*Tanaya\*\*\*\*\*\*\*\*\*");  
  
 Cube cub=new Cube(7);  
 cub.draw();  
 System.*out*.println("Area of cube is "+cub.calculateArea());  
 System.*out*.println("\*\*\*\*\*\*\*\*\*\*\*Tanaya\*\*\*\*\*\*\*\*\*");  
  
  
 }  
  
}

Output:=

"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\lib\idea\_rt.jar=61039:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\Tanaya\IdeaProjects\Corejava\out\production\Corejava AbstractClassMain

============================

Draw method in shape class given below

Area of rectangle is 300.0

\*\*\*\*\*\*\*\*\*\*\*Tanaya\*\*\*\*\*\*\*\*\*

Draw method in shape class given below

Area of cube is 294.0

\*\*\*\*\*\*\*\*\*\*\*Tanaya\*\*\*\*\*\*\*\*\*

Process finished with exit code 0

7)

public class DessertShoppe {  
  
 public final static double *TAX\_RATE* = 6.5; // 6.5%  
 public final static String *STORE\_NAME* = "Tanaya Dessert Shoppe";  
 public final static int *MAX\_ITEM\_NAME\_SIZE* = 25;  
 public final static int *COST\_WIDTH* = 6;  
  
 public static String cents2dollarsAndCents(int cents) {  
 String s = "";  
  
 if (cents < 0) {  
 s += "-";  
 cents \*= -1;  
 }  
  
 int dollars = cents/100;  
 cents = cents % 100;  
  
 if (dollars > 0)  
 s += dollars;  
  
 s +=".";  
  
 if (cents < 10)  
 s += "0";  
  
 s += cents;  
 return s;  
 }  
}  
  
  
abstract class DessertItem {  
  
 protected String name;  
  
 public DessertItem() {  
 this("");  
 }  
  
 public DessertItem(String name) {  
 if (name.length() <= DessertShoppe.*MAX\_ITEM\_NAME\_SIZE*)  
 this.name = name;  
 else  
 this.name = name.substring(0,DessertShoppe.*MAX\_ITEM\_NAME\_SIZE*);  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public abstract int getCost();  
}  
  
  
class Cookie extends DessertItem{  
  
 protected double number;  
 protected double pricePerDoze;  
  
 public Cookie(String \_n, double \_ppd, int \_number){  
 super(\_n);  
 pricePerDoze = \_ppd;  
 number = \_number;  
 }  
  
 public int getCost(){  
 return (int)Math.*round*(number / 12 \* pricePerDoze);  
 }  
}  
  
  
class Candy extends DessertItem{  
  
 protected double weight;  
 protected double pricePerPound;  
  
 public Candy(String \_n, double \_ppp, int \_w){  
 //using parent's constructor with name while storing its own properties  
 super(\_n);  
 pricePerPound = \_ppp;  
 weight = \_w;  
 }  
  
 public int getCost(){  
 return (int)Math.*round*(weight \* pricePerPound);  
 }  
}  
  
  
class IceCream extends DessertItem{  
  
 protected int cost;  
  
 public IceCream(String \_n, int \_cost){  
 super(\_n);  
 cost = \_cost;  
 }  
  
 public int getCost(){  
 return cost;  
 }  
}  
  
  
class Sundae extends IceCream{  
  
 protected String topName;  
 protected int topCost;  
  
 public Sundae(String \_n0, int \_cost0, String \_n1, int \_cost1){  
 //put the icecream name in icecream while putting top name and cost in a separate property  
 super(\_n0, \_cost0);  
 topName = \_n1;  
 topCost = \_cost1;  
 }  
  
 public final String getName(){  
 //return both the icecream name and the topping name  
 return name + " " + topName;  
 }  
  
 public int getCost(){  
 //return the sum of the icecream and the topping  
 return cost + topCost;  
 }  
}  
  
  
class Checkout{  
  
 protected int size;  
 protected DessertItem[] dessertItems;  
 protected int amount;  
 protected int sum;  
 protected final double taxRate;  
  
 Checkout(){  
 size = 100;  
 dessertItems = new DessertItem[size];  
 amount = 0;  
 sum = 0;  
 taxRate = DessertShoppe.*TAX\_RATE*;  
 }  
  
 public void enterItem(DessertItem d){  
 dessertItems[amount] = d;  
 amount ++;  
 }  
  
 public int numberOfItems(){  
 return amount;  
 }  
  
 public int totalCost(){  
 //make sum into zero, and calculate price from every item  
 sum = 0;  
 for(int i = 0; i < amount; i ++){  
 sum += dessertItems[i].getCost();  
 }  
 return sum;  
 }  
  
 public int totalTax(){  
 //use the totalCost method  
 return (int)(Math.*round*(this.totalCost() \* taxRate / 100));  
 }  
  
 public void clear(){  
 //clear the array  
 for(DessertItem d : dessertItems){  
 d = null;  
 }  
 amount = 0;  
 sum = 0;  
 }  
  
 public String toString(){  
 String result = "Thank You! \n";  
  
 result += DessertShoppe.*STORE\_NAME* + "\n";  
  
 result += "Purchased: ";  
  
 String totalPay = DessertShoppe.*cents2dollarsAndCents*( totalCost()+totalTax() );  
 if(totalPay.length() > DessertShoppe.*COST\_WIDTH*){  
 totalPay = totalPay.substring(0, DessertShoppe.*COST\_WIDTH*);  
 }  
 result += "$" + totalPay;  
 return result;  
 }  
}  
  
  
class TestCheckout {  
  
 public static void main(String[] args) {  
  
 Checkout checkout = new Checkout();  
  
 checkout.enterItem(new Candy("Peanut Butter Fudge", 2.25, 399));  
 checkout.enterItem(new IceCream("Vanilla Ice Cream",105));  
 checkout.enterItem(new Sundae("Choc. Chip Ice Cream",145, "Hot Fudge", 50));  
 checkout.enterItem(new Cookie("Oatmeal Raisin Cookies", 4, 399));  
  
 System.*out*.println("\nNumber of items: " + checkout.numberOfItems() + "\n");  
 System.*out*.println("\nTotal cost: " + checkout.totalCost() + "\n");  
 System.*out*.println("\nTotal tax: " + checkout.totalTax() + "\n");  
 System.*out*.println("\nCost + Tax: " + (checkout.totalCost() + checkout.totalTax()) + "\n");  
 System.*out*.println(checkout);  
  
 checkout.clear();  
  
 checkout.enterItem(new IceCream("Strawberry Ice Cream",145));  
 checkout.enterItem(new Sundae("Vanilla Ice Cream",105, "Caramel", 50));  
 checkout.enterItem(new Candy("Gummy Worms", 1.33, 89));  
 checkout.enterItem(new Cookie("Chocolate Chip Cookies", 4, 399));  
 checkout.enterItem(new Candy("Salt Water Taffy", 1.5, 209));  
 checkout.enterItem(new Candy("Candy Corn",3.0, 109));  
  
 System.*out*.println("\nNumber of items: " + checkout.numberOfItems() + "\n");  
 System.*out*.println("\nTotal cost: " + checkout.totalCost() + "\n");  
 System.*out*.println("\nTotal tax: " + checkout.totalTax() + "\n");  
 System.*out*.println("\nCost + Tax: " + (checkout.totalCost() + checkout.totalTax()) + "\n");  
 System.*out*.println(checkout);  
 }  
}

output:-

Checkout checkout = new Checkout();

"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\lib\idea\_rt.jar=61621:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\Tanaya\IdeaProjects\Corejava\out\production\Corejava TestCheckout

Number of items: 4

Total cost: 1331

Total tax: 87

Cost + Tax: 1418

Thank You!

Tanaya Dessert Shoppe

Purchased: $14.18

Number of items: 6

Total cost: 1192

Total tax: 77

Cost + Tax: 1269

Thank You!

Tanaya Dessert Shoppe

Purchased: $12.69

Process finished with exit code 0