

**CSC186 – OBJECT ORIENTED**

**PROGRAMMING**

**LAB ASSIGNMENT 7**

NAME : MUHAMMAD REDZA BIN MAHAYADIN

STUDENT ID : 2022676696

GROUP : RCDCS1102B

LECTURER : SIR MOHD NIZAM BIN OSMAN

QUESTION 7.6

SOURCE CODE 1.1 :Main Class

import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in); // for strings

        Scanner in1 = new Scanner (System.in); // for others

        System.out.print("Enter number of order: ");

        int numOrder = in1.nextInt();

        DesignService[] ds = new DesignService[numOrder];

        for (int i = 0; i < numOrder; i++) {

            System.out.println();

            System.out.println("\tOrder " + (i + 1));

            System.out.print("Enter customer name: ");

            String custName = in.nextLine();

            System.out.print("Enter deposit: ");

            double deposit = in1.nextDouble();

            System.out.print("Is urgent order? [true/false]: ");

            boolean urgentOrder = in1.nextBoolean();

            System.out.print("\n\t1. Physical Design\n\t2. Digital Design\nEnter your choice [1-2]: ");

            int choice = in1.nextInt();

            if (choice == 1) {

                System.out.println();

                System.out.print("\t1. Banner\n\t2. Brochure\n\t3. Poster\n\t4. BusinessCard\nEnter your choice: [1-4]");

                int physicalType = in1.nextInt();

                System.out.print("Enter Printing (true/false): ");

                boolean print = in1.nextBoolean();

                int copy = 0;

                if (print) {

                    System.out.print("Enter number of copy: ");

                    copy = in1.nextInt();

                }

                ds[i] = new PhysicalDesign(custName, deposit, urgentOrder, physicalType, print, copy);

            } else if (choice == 2) {

                System.out.print("\n\t1. Website\n\t2. Advertisement\nEnter Digital Type: [1-2]");

                int digitalType = in1.nextInt();

                ds[i] = new DigitalDesign(custName, deposit, urgentOrder, digitalType);

            }//end if

        } //end for

        System.out.println();

        System.out.println("\tPhysical Design Order List");

        boolean foundPhysicalOrder = false;

        int cntUrgentWebsite = 0;

        for (DesignService designService : ds) {

            // Display all physical order

            if (designService.getCustName() != null) {

                if (designService instanceof PhysicalDesign) {

                PhysicalDesign pd = (PhysicalDesign) designService;

                    System.out.println(pd);

                    System.out.println("Payment: RM" + pd.calcPayment());

                    System.out.println();

                    foundPhysicalOrder = true;

                }

            }

            if (designService instanceof DigitalDesign) {

                // Count and display total urgent website design order

                DigitalDesign dd = (DigitalDesign) designService;

                if (dd.isUrgentOrder()) {

                    if(dd.getDigitalType() == 1) {

                        cntUrgentWebsite++;

                    }

                }

            }

        }

        if (!foundPhysicalOrder) {

                System.out.println("No Physical Order Available.");

        }

        System.out.println();

        System.out.println("Total urgent website design order: " + cntUrgentWebsite);

        in.close();

        in1.close();

    } //end main

} //end class

SOURCE CODE 1.2 : DesignService Class

public abstract class DesignService {

    protected String custName;

    protected double deposit;

    protected boolean urgentOrder;

    public DesignService(String custName, double deposit, boolean urgentOrder) {

        this.custName = custName;

        this.deposit = deposit;

        this.urgentOrder = urgentOrder;

    }

    public String getCustName() {

        return custName;

    }

    public double getDeposit() {

        return deposit;

    }

    public boolean isUrgentOrder() {

        return urgentOrder;

    }

    public String toString() {

        return "Customer name: " + custName + "\nDeposit: RM" + deposit + "\nUrgent order: " + urgentOrder;

    }

    public abstract double calcPayment();

}

SOURCE CODE 1.3: DigitalDesign Class

public class DigitalDesign extends DesignService{

    private int digitalType;

    public DigitalDesign(String custName, double deposit, boolean urgentOrder, int digitalType) {

        super(custName, deposit, urgentOrder);

        this.digitalType = digitalType;

    }

    public int getDigitalType() {

        return digitalType;

    }

    public String toString() {

        return super.toString() + "\nDigital type: " + digitalType;

    }

    public double calcPayment() {

        return 0;

    }

}

SOURCE CODE 1.4: PhysicalDesign Class

public class PhysicalDesign extends DesignService {

    private int physicalType;

    private boolean printing;

    private int numCopy;

    public PhysicalDesign(String custName, double deposit, boolean urgentOrder, int physicalType, boolean printing, int numCopy) {

        super(custName, deposit, urgentOrder);

        this.physicalType = physicalType;

        this.printing = printing;

        this.numCopy = numCopy;

    }

    public int getPhysicalType() {

        return physicalType;

    }

    public boolean isPrinting() {

        return printing;

    }

    public int getNumCopy() {

        return numCopy;

    }

    public String toString() {

        return super.toString() + "\nPhysical type: " + physicalType + "\nPrinting: " + printing;

    }

    public double calcPayment() {

        double charge = 0;

        double addCharge = 0;

        switch (physicalType) {

            case 1:

                charge = 250;

                addCharge = 40;

                break;

            case 2:

                charge = 250;

                addCharge = 10;

                break;

            case 3:

                charge = 150;

                addCharge = 35;

                break;

            case 4:

                charge = 100;

                addCharge = 3;

                break;

            default:

                System.out.println("Invalid physical type");

                break;

        }

        if (printing) {

            charge += addCharge \* numCopy;

        }

        if (urgentOrder) {

            charge += 50;

        }

        return charge - deposit;

    }

}