1. TRAINEE

BATCH.JAVA

package com.css.corejava.main;

import java.util.Arrays;

public class Batch {

private String batchCode;

private String startDate;

private String endDate;

private Traniee[] traniees;

// constructor

public Batch(String batchCode, String startDate, String endDate, Traniee[] traniee) {

super();

this.batchCode = batchCode;

this.startDate = startDate;

this.endDate = endDate;

this.traniees = traniee;

}

public Batch() {

super();

}

// setters and getters

public String getBatchCode() {

return batchCode;

}

public void setBatchCode(String batchCode) {

this.batchCode = batchCode;

}

public String getStartDate() {

return startDate;

}

public void setStartDate(String startDate) {

this.startDate = startDate;

}

public String getEndDate() {

return endDate;

}

public void setEndDate(String endDate) {

this.endDate = endDate;

}

public Traniee[] getTraniee() {

return traniees;

}

public void setTraniee(Traniee[] traniee) {

this.traniees = traniee;

}

public Traniee[] getTraniee(int age) {

Traniee[] foundTraniees = new Traniee[3];

int i = 0;

for (Traniee traniee : traniees) {

int sirpi= traniee.getAge();

if (sirpi== age) {

foundTraniees[i] = traniee;

i++;

}

}

return foundTraniees;

}

public Traniee[] getTraniee(String gender) {

Traniee[] foundTraniees = new Traniee[3];

int i = 0;

for (Traniee traniee : traniees) {

String gen = traniee.getGender();

if (gen.equals(gender)) {

foundTraniees[i] = traniee;

i++;

}

}

return foundTraniees;

}

@Override

public String toString() {

return "Batch [batchCode=" + batchCode + ", startDate=" + startDate + ", endDate=" + endDate + ", traniee="

+ Arrays.toString(traniees) + "]";

}

}

MAIN.JAVA

package com.css.corejava.main;

public class Main {

public static void main(String[] args) {

Traniee[] traniees = new Traniee[3];

traniees[0] = new Traniee(100,"sirpi", 546789,"sirpimahi@gmail.com","female", 2 );

Batch batch = new Batch();

System.out.println(batch.getTraniee(22));

System.out.println(batch.getTraniee("female"));

}

}

TRAINEE.JAVA

package com.css.corejava.main;

public class Traniee {

private int tranieeeId;

private String tranieeName;

private int contactNo;

private String email;

private String gender;

private int age;

// constructor

public Traniee(int tranieeeId, String tranieeName, int contactNo, String email, String gender, int age) {

super();

this.tranieeeId = tranieeeId;

this.tranieeName = tranieeName;

this.contactNo = contactNo;

this.email = email;

this.gender = gender;

this.age = age;

}

public Traniee() {

super();

// TODO Auto-generated constructor stub

}

// getters and setters

public int getTranieeeId() {

return tranieeeId;

}

public void setTranieeeId(int tranieeeId) {

this.tranieeeId = tranieeeId;

}

public String getTranieeName() {

return tranieeName;

}

public void setTranieeName(String tranieeName) {

this.tranieeName = tranieeName;

}

public int getContactNo() {

return contactNo;

}

public void setContactNo(int contactNo) {

this.contactNo = contactNo;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

@Override

public String toString() {

return "Traniee [tranieeeId=" + tranieeeId + ", tranieeName=" + tranieeName + ", contactNo=" + contactNo

+ ", email=" + email + ", gender=" + gender + ", age=" + age + "]";

}

}

2.PROJECT

PROJECT.JAVA

publicclass Project {

private intprojectId;

private String projectName;

private String headOfTheProject;

private int numOfMembers;

//constructor

public Project(int projectId, String projectName, String headOfTheProject, intnumOfMembers) {

super();

this.projectId = projectId;

this.projectName = projectName;

this.headOfTheProject = headOfTheProject;

this.numOfMembers = numOfMembers;

}

//setters and getters

public intgetProjectId() {

return projectId;

}

public void setProjectId(**i**ntprojectId) {

this.projectId = projectId;

}

public String getProjectName() {

return projectName;

}

public void setProjectName(String projectName) {

this.projectName = projectName;

}

public String getHeadOfTheProject() {

returnheadOfTheProject;

}

public void setHeadOfTheProject(String headOfTheProject) {

this.headOfTheProject = headOfTheProject;

}

public int getNumOfMembers() {

returnnumOfMembers;

}

public void setNumOfMembers(intnumOfMembers) {

this.numOfMembers = numOfMembers;

}

}

MAIN.JAVA

public class Main {

public static void main(String[] args) {

Project myProject = newProject(1673,"Bank Application form", "sirpi", 22);

System.out.println("Project ID is " + myProject.getProjectId());

System.*out*.println("The name of the project is " + myProject.getProjectName());

System.*out*.println("The head of the project is " + myProject.getHeadOfTheProject());

System.*out*.println("There are " + myProject.getNumOfMembers() + " members in the project");

}

}

3.STRING SERVICE PROVIDER

1.

package com.css.corejava.stringservieproviderreverse;

public class Stringserviceprovider{

public static void main(String[] args) {

StringServiceProvider s=new StringServiceProvider;

String rev = s.strRev("hello world");

System.out.println(rev);

String replace=s.replace(Str,replacement);

System.out.println(replace);

}

}

2.LINEARSEARCH:

package com.css.corejava.stringservieproviderreverse;

public class StringLinearSearch{

public static int linearSearch(int[] arr, int key){

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

if(arr[i] == key){

return i;

}

}

return -1;

}

public static void main(String a[]){

int[] a1= {1,2,3,4,5,6,7,8};

int key = 20;

System.out.println(key+" is found at index: "+linearSearch(a1, key));

}

}

3.

package com.css.corejava.stringservieproviderreverse;

public class StringServiceProviderReverse {

public static void main(String[] args) {

String name = "sirpiya";

int len = 0, a = 0;

try {

while (true) {

if (name.charAt(a) != '\0')

len++;

a++;

}

} catch (Exception e) {

System.out.println(name);

System.out.println(len);

}

for (int i = len - 1; i >= 0; i--) {

System.out.print(name.charAt(j));

}

}

}

4,5,6.BANK APPLICATION

4.Bankdetail.java

package com.css.corejava.main;

import com.css.corejava.main.IBankServiceProvider;

import com.css.corejava.exception.InsufficientBalanceException;

import com.css.corejava.exception.InvalidAccountNoException;

import java.util.Arrays;

public class Bankdetail implements IBankServiceProvider {

private BankAccount[] bankAccount;

private String IFSCCode;

private String bankName;

// using constructor

public Bank(BankAccount[] bankAccount, String IFSCCode, String bankName) {

super();

this.bankAccount = bankAccount;

this.IFSCCode = ifsccode;

this.bankName = bankName;

}

//override method

}

@Override

public BankAccount CheckAccount(String accountNo)throws InvalidAccountNoException {

BankAccount foundacc= null;

for(BankAccount account:bankAccount) {

if(account.getAccountNo().equals(accountNo)) {

foundacc=account;

break;

}

}

if(foundacc==null) {

throw new InvalidAccountNoException();

}

return foundacc;

}

@Override

public double getBalance(String accountNo)throws InvalidAccountNoException{

double balance=0.0;

BankAccount foundacc=CheckAccount(accountNo);

balance=foundacc.getBalance();

return balance;

}

@Override

public boolean depositMoney(String accountNo, double amount) throws InvalidAccountNoException{

boolean flagDeposit = false;

BankAccount foundacc= CheckAccount(accountNo);

if(foundacc!=null) {

foundacc.setBalance(foundAccount.getBalance()+amount);

FlagDeposit=true;

}

return true;

}

@Override

public boolean withdrawMoney(String accountNo, double amount)throws InvalidAccountNoException, InsufficientBalanceException{

boolean flagWithdraw = false;

BankAccount foundacc=CheckAccount(accountNo);

if(foundacc!=null) {

if(foundacc.getBalance()>amount) {

foundAccount.setBalance(foundacc.getBalance()-amount);

FlagWithdraw= true;

}else {

throw new InvalidAccountNoException();

}

}

return FlagWithdraw;

}

@Override

public boolean transferMoney(String fromAccountNo,String toAccountNo,double amount)throws InvalidAccountNoException,InsufficientBalanceException

{

boolean transferflag = false;

boolean flagWithdraw= withdrawMoney(fromAccountNo, amount);

boolean flagDeposit=depositMoney(toAccountNo, amount);

if(flagWithdraw) {

flagDeposit=depositMoney(toAccountNo,amount);

}

if(flagDeposit)

transferflag=true;

else

depositMoney(AccountNo, amount);

return transferflag;

}

//getters and setters

public BankAccount[] getBankAccount() {

return bankAccount;

}

public void setBankAccount(BankAccount[] bankAccount) {

this.bankAccount = bankAccount;

}

public String getIFSCCode() {

return IFSCCode;

}

public void setIFSCCode(String ifsccode) {

IFSCCode = ifsccode;

}

public String getBankName() {

return bankName;

}

public void setBankName(String bankName) {

this.bankName = bankName;

}

@Override

public String toString() {

return "Bank [bankAccount=" + Arrays.toString(bankAccount) + ", IFSCCode=" + IFSCCode + ", bankName=" + bankName

+ "]";

}

@Override

public BankAccount checkAccount(String accountNo) throws InvalidAccountNoException {

// TODO Auto-generated method stub

return null;

}

}

BankAcc.java

package com.css.corejava.main;

public class BankAcc {

private static int assignedNo;

static {

assignedNo=1000;

}

private String accountNo;

private String accountName;

private double balance=0;

//constructor

public Bankacc(String accountNo, String accountName, double balance) {

super();

this.accountNo = ""+assignedNo++;

this.accountName = accountName;

this.balance = balance;

}

public Bankacc(String accountNo, String accountName) {

super();

this.accountNo = accountNo;

this.accountName = accountName;

this.balance=1000.00;

}

public Bankacc() {

super();

// TODO Auto-generated constructor stub

}

//getters and setters

public static int getassignedNo() {

return assignedNo;

}

public static void setassignedNo(int assignedNo) {

BankAccount.assignedNo = assignedNo;

}

public String getAccountNo() {

return accountNo;

}

public void setAccountNo(String accountNo) {

this.accountNo = accountNo;

}

public String getAccountName() {

return accountName;

}

public void setAccountName(String accountName) {

this.accountName = accountName;

}

public double getBalance() {

return balance;

}

public void setBalance(double balance) {

this.balance = balance;

}

@Override

public String toString() {

return "BankAccount [accountNo=" + accountNo + ", accountName=" + accountName + ", balance=" + balance + "]";

}

}

6.INVALIDACCOUNTNOEXCEPTION.JAVA

package com.css.corejava.main;

public class InvalidAccountNoException {

private String invalidmessage= "Invalid account number";

public InvalidAccountNoException() {

super();

// TODO Auto-generated constructor stub

}

public InvalidAccountNoException(String invalidmessage) {

super();

this.invalidmessage = invalidmessage;

}

@Override

public String getMessage() {

return this.invalidmessage;

}

}

6.INSUFFICIENTBALANCEEXCEPTIO.JAVA

package com.css.corejava.main;

public class InsufficientBalanceException {

private String invalidmessage= "Insufficient balance";

public InsufficientBalanceException() {

super();

// TODO Auto-generated constructor stub

}

public InsufficientBalanceException(String invalidmessage) {

super();

this.invalidmessage = invaidmessage;

}

@Override

public String getMessage() {

return this.invalidmessage;

}

}

5.IBANKSERVICEPROVIDER.JAVA

package com.css.corejava.main;

import com.css.corejava.main.BankAccount;

import com.css.corejava.exception.InsufficientBalanceException;

import com.css.corejava.exception.InvalidAccountNoException;

public interface IBankServiceProvider {

BankAccount bankaccount (String accountNo) throws InvalidAccountNoException;

double balance(String accountNo)throws InvalidAccountNoException;

boolean depositMoney(String accountNo, double amount)throws InvalidAccountNoException;

boolean withdrawMoney(String accountNo, double amount)throws InvalidAccountNoException,InsufficientBalanceException;

boolean transferMoney(String fromAccountNo,String toAccountNo, double amount)throws InvalidAccountNoException,InsufficientBalanceException;

}

MAIN.JAVA:

package com.css.corejava.main;

import com.css.corejava.main.Bank;

import com.css.corejava.main.BankAccount;

import com.css.corejava.exception.InsufficientBalanceException;

import com.css.corejava.exception.InvalidAccountNoException;

public class Main {

public static void main(String[] args) {

BankAccount[] accounts = new BankAccount[3];

accounts[0] = new BankAccount("priya");

accounts[1] = new BankAccount("moni");

accounts[2] = new BankAccount("archana" );

Bank hdfc = new Bank(accounts, "adayar", "HDFC");

BankAccount foundacc=null;

double balance=0.0;

try {

balance=hdfc.getBalance("10000");

hdfc.depositMoney("1000",2000.00);

hdfc.withdrawMoney("1000", 100);

hdfc.transferMoney("1000", " 2000.00);

} catch (InvalidAccountNoException | InsufficientBalanceException e) {

e.printStackTrace();

// TODO: handle exception

}

System.out.println(balance);

try {

System.out.println(hdfc.getBalance("1000"));

System.out.println(hdfc.getBalance("102"));

} catch (InvalidAccountNoException e) {

e.printStackTrace();

}

}

}

7.COMPANY

COMPANY.JAVA

package com.css.java.company;

import java.util.Arrays;

import java.util.HashMap;

import java.util.Map;

public class Company {

private String companyID;

private Address empAddress;

private Employee emp;

private HashMap<Integer, Employee> employeeHash;

// constructor

public Company(String companyID, Address empAddress, Employee emp, HashMap<Integer, Employee> employeeHash) {

super();

this.companyID = companyID;

this.empAddress = empAddress;

this.emp = emp;

this.employeeHash = employeeHash;

}

public Company() {

super();

// TODO Auto-generated constructor stub

}

// getters and setters

public String getCompanyID() {

return companyID;

}

public void setCompanyID(String companyID) {

this.companyID = companyID;

}

public Address getEmpAddress() {

return empAddress;

}

public void setEmpAddress(Address empAddress) {

this.empAddress = empAddress;

}

public Employee getEmp() {

return emp;

}

public void setEmp(Employee emp) {

this.emp = emp;

}

public HashMap<Integer, Employee> getEmployeeHash() {

return employeeHash;

}

public void setEmployeeHash(HashMap<Integer, Employee> employeeHash) {

this.employeeHash = employeeHash;

}

@Override

public String toString() {

return "Company [companyID=" + companyID + ", empAddress=" + empAddress + ", emp=" + emp + ", employeeHash="

+ employeeHash + "]";

}

}

ADDRESS.JAVA

package com.css.java.company;

public class Address {

private int doorNo;

private String street;

private String city;

// constructor

public Address(int doorNo, String street, String city, ) {

super();

this.doorNo = doorNo;

this.street = street;

this.city = city;

}

public Address() {

super();

// TODO Auto-generated constructor stub

}

// getters and setters

public int getDoorNo() {

return doorNo;

}

public void setDoorNo(int doorNo) {

this.doorNo = doorNo;

}

public String getStreet() {

return street;

}

public void setStreet(String street) {

this.street = street;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

@Override

public String toString() {

return "Address [doorNo=" + doorNo + ", street=" + street + ", city=" + city + "]";

}

}

EMPLOYEE.JAVA

package com.css.java.company;

public class Employee {

private String name;

private int age;

private int employeeID;

// constructor

public Employee(String name, int age, int employeeID) {

super();

this.name = name;

this.age = age;

this.employeeID = employeeID;

}

public Employee() {

super();

// TODO Auto-generated constructor stub

}

// getters and setters

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public int getEmployeeID() {

return employeeID;

}

public void setEmployeeID(int employeeID) {

this.employeeID = employeeID;

}

}

MAIN.JAVA

package com.css.java.main;

import java.util.HashMap;

import java.util.Map;

public class Main {

public static void main(String[] args) {

Map<Integer,String> employees=new HashMap<>();

employees.put(100, "sirpi");

employees.put(101, "riya");

employees.put(102, "moni");

System.out.println(employees.get(102));

}

}

11.PRODUCT

PRODUCT.JAVA

package com.test;

import java.util.Comparator;

public class Product implements Comparable<Product>{

    private int productId;

    private String productName;

    private String productDescription;

    private double price;

             public Product()

            {

            }

            public Product(int productId, String productName, String

                             productDescription, double price)

           {

                this.productId = productId;

                this.productName = productName;

                this.productDescription = productDescription;

                this.price = price;

           }

                     public int getProductId()

                     {

                             return productId;

                     }

                     public void setProductId(int productId)

                     {

                           this.productId = productId;

                     }

                     public String getProductName()

                     {

                            return productName;

                     }

    public void setProductName(String productName)

     {

        this.productName = productName;

     }

     public String getProductDescription()

     {

        return productDescription;

     }

     public void setProductDescription(String productDescription)

     {

        this.productDescription = productDescription;

      }

    public double getPrice()

   {

        return price;

    }

    public void setPrice(double price)

    {

        this.price = price;

    }

    @Override

    public int compareTo(Product p)

    {

        return this.productId-p.productId;

    }

    @Override

    public String toString()

    {

        return "price=        " + price + ", productDescription=           "

                + productDescription + ", productId=           " + productId

                + ", productName=              " + productName;

    }

    public static class ProductInnerClass implements Comparator<Product>

    {

        @Override

        public int compare(Product p1, Product p2)

        {

            int i=Double.compare(p1.getPrice(), p2.getPrice());

            if(i==0)

            {

                return p1.getProductId()-p2.getProductId();

            }

            return i;

        }

    }

}

PRODUCTMAIN.JAVA

package com.css.corejava.main;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import com.alok.client.Product;

public class ProductMain {

    public static void main(String[] args) throws NumberFormatException, IOException {

        ArrayList<Product> productList=new ArrayList<Product>();

        productList.add(new Product(1012, "Monitor", "15 inch", 4000.00));

        productList.add(new Product(1002, "Monitor", "17 inch", 5000.00));

        productList.add(new Product(123, "Del", "Laptop", 4000.00));

        productList.add(new Product(234, "Mouse", "Optical Mouse", 200.00));

        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

        int i=1;

        while(i<3)

        {

            System.out.println("1 : Sort by id");

            System.out.println("2 : Sort by name");

            System.out.println("3 : Sort by price");

            int ch=Integer.parseInt(br.readLine());

            switch(ch)

            {

                case 1 : Collections.sort(productList);

                         System.out.println("Sorted product : ");

                         for(Product p : productList)

                         {

                             System.out.println(p);

                         }

                         break;

                case 2 : Collections.sort(productList, new Comparator<Product>()

                         {

                                @Override

                         public int compare(Product p1, Product p2)

                         {

                         int i=p1.getProductName().compareToIgnoreCase

                                                  (p2.getProductName());

                            if(i==0)

                            {

                            return p1.getProductDescription().

                             compareToIgnoreCase(p2.getProductDescription());

                            }

                                       return i;

                         }

                       });

                System.out.println("Sorted product : ");

                 for(Product p : productList)

                 {

                     System.out.println(p);

                 }

                 break;

               case 3 :

             Collections.sort(productList,new Product.ProductInnerClass());

                    System.out.println("Sorted product : ");

                     for(Product p : productList)

                     {

                         System.out.println(p);

                     }

                     break;

                default : System.out.println("Invalid Option");

                            System.exit(0);

            }

        }

    }

}