1-> Create 1st class, where a class has one method that takes 2 integers as parameters and returns an integer denoting their sum.Create 2nd class that inherits from a superclass of the first class.Again 3rd class and here print the output of the first class method with the help of 2nd class object.

import java.util.\*;

class Class1 {

void print(){System.out.println("Enter the first number");

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

System.out.println("Enter the second number");

int b=sc.nextInt();

System.out.println("Sum of a and b is : "+ (a+b));

}

}

class Class2 extends Class1{

}

public class makeClass{

public static void main(String[] args) {

Class1 tej=new Class2();

tej.print();

}

}

2-> Create an interface that contains a method, where the method return type is int and the method has one int type parameter. You need to write another class that implements the interface.And your function just takes an integer as input and return the sum of all its divisors. For example divisors of 8 are 1, 2, 4 and 8, so divisor addition should return 15. The value of n will be at most 1000.

interface divisor{

void print(int n);

}

class divisor1 implements divisor{

public void print(int n){

int sum=0;

for(int i=1;i<=n;i++){

if(n%i==0) sum+=i;

}

System.out.println(sum);

}

}

class divisorSum{

public static void main(String[] args) {

divisor1 tej= new divisor1();

int n=9;

tej.print(n);

}

}

3-> Write a program to sort the given array of String in descending order. Your string value should be the same during the output. Like- .12 printed as .12 and 0.12 printed as 0.12

Ex-> String[] str = {“-100”,”50”,”.12”,”0.12”,”0”,”000.000”};

Answer:-

import java.util.\*;

public class sort {

public static void main(String[] arg){

String [] str={"-100","50",".12","0.12","0","000.000"};

String [] str1={"-100","50",".12","0.12","0","000.000"};

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

int k=0;

for(int j=0;j<str.length;j++){

if(Float.parseFloat(str[i]) < Float.parseFloat(str[j])) k++;

}

str1[k]=str[i];

}

System.out.println(Arrays.toString(str1));

}

}

4-> You will be given two integers a and b as input, you have to compute a/b.

If a and b are not 32 bit signed integers or if b is zero,

exception will occur and you have to report it.

Read sample Input/Output to know what to report in case of exceptions.

Sample Input 0:

10

3

Sample Output 0:

3

Sample Input 1:

10

Hello

Sample Output 1:

java.util.InputMismatchException

Sample Input 2:

10

0

Sample Output 2:

java.lang.ArithmeticException: / by zero

Sample Input 3:

23.323

0

Sample Output 3:

java.util.InputMismatchException

Sample Input 4:

2147483648

55

Sample Output 4:

java.util.InputMismatchException

Answer:-

import java.util.InputMismatchException;

import java.util.Scanner;

public class Exception {

public static void main(String[] args) {

try {

System.out.println("Enter a : ");

Scanner sc = new Scanner(System.in);

int a = sc.nextInt();

System.out.println("Enter b : ");

int b = sc.nextInt();

sc.close();

a = a / b;

} catch (ArithmeticException e) {

System.out.println(e);

} catch (InputMismatchException e) {

System.out.println(e);

}

}

}

5->

Write a Java program to create a class known as "SBIBankAccount"

with methods called deposit() and withdraw().

And create opening account methos openAccount.

create sub classes for premiumAccount and jundhanAccount and override the deposit,

withdraw, and openAccount in sub class.

Where if you open premium account should be start from 5K and your jundhanAccount

start from 0 balance.where withdraw() method to prevent withdrawals if the account balance falls below 5K for the premium account and prevent withdrawals if the account balance falls

below 0 for the jundhanAccount.

Answer:-

class sbiBankAcccount {

double balance;

void openAccout(){

System.out.println("Accout created successfully!");

}

void deposit(double amount){

balance+=amount;

System.out.println("Your amount "+ amount +" deposited successfully and current balance is "+ balance);

}

void withdraw(double amount){

if(balance>=amount){

balance-=amount;

System.out.println("Your amount "+ amount +" withdrawn successfully and current balance is "+ balance);

}

else{

System.out.println("Insufficient balance");

}

}

}

class premiumAccount extends sbiBankAcccount{

void openAccout() {

balance=5000;

System.out.println("Accout created successfully!");

}

void deposit(double amount){

balance+=amount;

System.out.println("Your amount "+ amount +" deposited successfully and current balance is "+ balance);

}

void withdraw(double amount){

if((balance-amount)>=5000){

balance-=amount;

System.out.println("Your amount "+ amount +" withdrawn successfully and current balance is "+ balance);

}

else{

System.out.println("Insufficient balance");

}

}

}

class jundhanAccount extends sbiBankAcccount{

void openAccout() {

balance=0;

System.out.println("Accout created successfully!");

}

void deposit(double amount){

balance+=amount;

System.out.println("Your amount "+ amount +" deposited successfully and current balance is "+ balance);

}

void withdraw(double amount){

if((balance-amount)>=0){

balance-=amount;

System.out.println("Your amount "+ amount +" withdrawn successfully and current balance is "+ balance);

}

else{

System.out.println("Insufficient balance");

}

}

}

public class Bank{

public static void main(String[] args) {

premiumAccount preAcc = new premiumAccount();

preAcc.openAccout();

preAcc.deposit(10000);

preAcc.withdraw(1000);

preAcc.withdraw(2000);

System.out.println("Premium Account balance is remaining : " + preAcc.balance);

jundhanAccount junAcc = new jundhanAccount();

junAcc.openAccout();

junAcc.deposit(10000);

junAcc.withdraw(1000);

junAcc.withdraw(2000);

System.out.println("Jundhan Account balance is remaining : " + junAcc.balance);

}

}

6- Write a Java programming to create a banking system with three classes -

Bank, Account, SavingsAccount, and CurrentAccount. The bank should have a list of

accounts and methods for adding them. Accounts should be an interface with methods

to deposit, withdraw, calculate interest, and view balances. SavingsAccount and

CurrentAccount should implement the Account interface and have their own unique

methods.

Answer:-

import java.util.\*;

class Bank {

List<Account> accounts;

Bank() {

accounts = new ArrayList<>();

}

void addAccount(Account acc) {

accounts.add(acc);

}

}

interface Account {

void deposit(double amount);

void withdraw(double amount);

double calculateInterest();

double viewBalance();

}

class SavingsAccount implements Account {

double balance;

double interestRate;

SavingsAccount(double amount, double Rate) {

balance = amount;

interestRate = Rate;

}

public void deposit(double amount) {

balance += amount;

System.out.println("Your Amount " + amount + " deposited successfully to Savings Account and your current balance is " +balance);

}

public void withdraw(double amount) {

if (balance >= amount) {

balance -= amount;

System.out.println("Your Amount " + amount + " withdrawn successfully from Savings Account and your current balance is : " + balance);

} else {

System.out.println("Insufficient balance.");

}

}

public double calculateInterest() {

return balance \* interestRate;

}

public double viewBalance() {

return balance;

}

public void addInterest() {

double interest = calculateInterest();

balance += interest;

System.out.println("Interest added to Savings Account is: " + interest);

}

}

class CurrentAccount implements Account {

double balance;

double overdraftLimit;

public CurrentAccount(double amount, double Limit) {

balance = amount;

overdraftLimit = Limit;

}

public void deposit(double amount) {

balance += amount;

System.out.println("Your Amount " + amount + " deposited successfully to Current Account and current balance is : " +balance);

}

public void withdraw(double amount) {

if (balance + overdraftLimit >= amount) {

balance -= amount;

System.out.println(" Your Amount " + amount + " withdrawn successfully from Current Account and your current balance is : " + balance);

} else {

System.out.println("Insufficient balance.");

}

}

public double calculateInterest() {

return 0;

}

public double viewBalance() {

return balance;

}

public void setOverdraftLimit(double Limit) {

overdraftLimit = Limit;

System.out.println("Overdraft limit updated: " + Limit);

}

}

public class BankAcc{

public static void main(String[] args) {

Bank bank = new Bank();

SavingsAccount savingsAccount = new SavingsAccount(2000, 0.01);

bank.addAccount(savingsAccount);

CurrentAccount currentAccount = new CurrentAccount(2000, 10000);

bank.addAccount(currentAccount);

savingsAccount.deposit(1500);

savingsAccount.withdraw(1000);

double savingsBalance = savingsAccount.viewBalance();

System.out.println(" Your Savings Account Balance is : " + savingsBalance);

currentAccount.deposit(1500);

currentAccount.withdraw(1000);

double currentBalance = currentAccount.viewBalance();

System.out.println("Your Current Account Balance is : " + currentBalance);

savingsAccount.addInterest();

double updatedSavingsBalance = savingsAccount.viewBalance();

System.out.println("Your Savings Account Balance: " + updatedSavingsBalance);

currentAccount.setOverdraftLimit(10000);

}

}