1. Write an application that accept two numbers, divides the first number with the second number and display the result. Hint: You need to handle **ArithmeticException** which is thrown when there is an attempt to divide a number by zero.

Ans **import** java.util.Scanner;

**public** **class** Test {

**public** **static** **void** main(String arg[]) {

**int** a,b;

Scanner sc=**new** Scanner(System.***in***);

System.***out***.println("Enter dividend");

a=sc.nextInt();

System.***out***.println("Enter divisor");

b=sc.nextInt();

**try** {

**int** c=a/b;

**if**(b==0) {

System.***out***.println("This line will not be executed");

}

**else** {

System.***out***.println("Output is" +c);

}

}

**catch**(ArithmeticException e) {

System.***out***.println(e);

}

System.***out***.println("After Execution is handled");

}

}

1. Carrying forward with the above problem, handle the **ArithmeticException** by raising **UnsupportedOperationException** as a solution

Ans **import** java.util.Scanner;

**public** **class** Test {

**public** **static** **void** main(String arg[]) {

**int** a,b;

Scanner sc=**new** Scanner(System.***in***);

System.***out***.println("Enter dividend");

a=sc.nextInt();

System.***out***.println("Enter divisor");

b=sc.nextInt();

**int** c=a/b;

**if**(b==0) {

**throw** **new** UnsupportedOperationException("Unsupported Operation Exception Occur");

}

**else** {

System.***out***.println("Output is" +c);

}

}

}

**class** UnsupportedOperationException **extends** RuntimeException {

UnsupportedOperationException(String name) {

**super**();

}

}

1. Write an application to perform withdraw functionality on a SavingAccount object. Point to note:
2. **RaiseInsufficientBalanceException** if you are trying to withdraw more than balanceor your balance is zero. E.g. if your balance is 2000 and if you are trying to withdraw 2100 or if you balance is 0 and you are trying to withdraw positive value.
3. **RaiseIllegalBankTransactionException** if you are trying to withdraw a negative value from your bank. E.g. if you try to withdraw a negative value savingAccoluntwithdraw(-1000)

Note: SavingAccount

|-long id

|-double balance

|-double withdraw(double amount)

|-double deposit(double amount)

Ans

**import** java.util.Scanner;

**public** **class** SavingAccount {

**public** **static** **void** main(String[] args) {

**double** amount, balance=100000;

Scanner sc=**new** Scanner(System.***in***);

System.***out***.println("Enter amount");

amount=sc.nextDouble();

**try** {

**if** (amount>balance)

{

**throw** **new** RaiseInsufficientBalanceException();

}

**else** {

System.***out***.println("balance is" +balance);

}

}

**catch**(RaiseInsufficientBalanceException e) {

e.printStackTrace();

}

**try** {

**if**(amount<0) {

**throw** **new** RaiseIllegalBanktransactionException();

}

}

**catch**(RaiseIllegalBanktransactionException ex) {

ex.printStackTrace();

}

}

}

**class** RaiseInsufficientBalanceException **extends** RuntimeException{

RaiseInsufficientBalanceException()

{

**super**("You have insufficient balance");

}

RaiseInsufficientBalanceException(String msg){

**super**(msg);

}

}

**class** RaiseIllegalBanktransactionException **extends** RuntimeException{

RaiseIllegalBanktransactionException(){

**super**("negative balance");

}

RaiseIllegalBanktransactionException(String msg){

**super**(msg);

}

}