Exception Handling Assignment-4

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

**package** org.exception.app;

**import** java.util.Scanner;

**public** **class** DivisonbyZero

{

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

{

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

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

**int** a=sc.nextInt();

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

**int** b=sc.nextInt();

**for**(**int** chance=4;chance>=1;chance--)

{

**try**

{

System.***out***.println(a/b);

System.***out***.println("Division is successful");

}

**catch**(ArithmeticException e)

{

**if**(chance==1)

{

System.***out***.println("Last attempt is failed,Division is unscussfull,s0 forcefully program is terminated");

}

**else**

{

System.***out***.println("Denominator cannot be zero");

System.***out***.println("You have "+(chance-1)+" attempts left");

System.***out***.println("Re-enter the denominator");

b=sc.nextInt();

}

}

}

}

}

OUTPUT:

Enter a number:

12

Enter b number:

0

Denominator cannot be zero

You have 3 attempts left

Re-enter the denominator

2

6

Division is successful

6

Division is successful

6

Division is successful

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

**package** org.exception.app;

**import** java.util.Scanner;

**public** **class** ArithException

{

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

{

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

System.***out***.println("Enter number1:");

**int** a=sc.nextInt();

System.***out***.println("Enter number2:");

**int** b=sc.nextInt();

**try**

{

System.***out***.println(a/b);

}

**catch**(ArithmeticException e)

{

System.***out***.println("Exception Handled "+e);

System.***out***.println("UnsupportedOperationException");

}

}

}

OUTPUT:

Enter number1:

12

Enter number2:

0

Exception Handled java.lang.ArithmeticException: / by zero

UnsupportedOperationException

1. Write an application to perform withdraw functionality on a SavingAccount object. Point to note:
2. Raise InsufficientBalanceException if you are trying to withdraw more than balance or when you balance is zero. E.g if you balance is 2000 and if are trying to withdraw 2100 or if you balance is 0 and you are trying to withdraw positive value.

**package** org.exception.app;

**import** java.util.Scanner;

**class** InsufficientBalanceException **extends** RuntimeException

{

}

**public** **class** SavingAccount

{

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

**public** **void** withdrawal(**double** a)

{

System.***out***.println("Enter your Id: ");

**long** id = sc.nextLong();

System.***out***.println("Enter your balance: ");

**double** b = sc.nextDouble();

**try**

{

**if**(a<=b)

{

b = b - a;

System.***out***.println("Balance= " + b);

}

**else**

{

**throw** **new** InsufficientBalanceException();

}

}

**catch** (InsufficientBalanceException e)

{

e.printStackTrace();

}

}

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

{

SavingAccount s = **new** SavingAccount();

s.withdrawal(2000);

}

}

OUTPUT:

Enter your Id:

2345

Enter your balance:

500

org.exception.app.InsufficientBalanceException

at org.exception.app.SavingAccount.withdrawal(SavingAccount.java:26)

at org.exception.app.SavingAccount.main(SavingAccount.java:38)

1. Raise IllegalBankTransactionException if you are trying to withdraw a negative value from your balance. E.g. if you try to withdraw a negative value savingACC withdraw(-1000);

Note: SavingAccount

|-long id

|-double balance

|-double withdraw(double amount)

|-double deposite(double amount)

**package** Account;

**import** java.util.Scanner;

**class** IllegalBankTransactionException **extends** RuntimeException

{

}

**public** **class** SavingsAccount

{

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

**public** **void** withdrawal(**double** a)

{

System.***out***.println("Enter your Id ");

**long** id = sc.nextLong();

System.***out***.println("Enter your balance ");

**double** b = sc.nextDouble();

**try** {

**if**(a>0)

{

System.***out***.println("Balance= " + b);

}

**else**

{

**throw** **new** IllegalBankTransactionException();

}

}

**catch** (IllegalBankTransactionException e)

{

e.printStackTrace();

}

}

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

{

SavingsAccount s = **new** SavingsAccount();

s.withdrawal(-10201);

}

}

**OUTPUT:**

Enter your Id

124563

Enter your balance

100

Account.IllegalBankTransactionException

at Account.SavingsAccount.withdrawal(SavingsAccount.java:21)

at Account.SavingsAccount.main(SavingsAccount.java:32)