Name of the Course : Learning Java 9 - Object Oriented Programming

Level : Moderate

Tool Stack : Java9 and Junit4

Problem Statement : Provide a code solution to generate unix user id based on his/her role as i.e. ordinary or super user ,using Java 9 solution of Stream API and forEach() method.

Description : ***Moon Info System*** is an IT company for Unix based high security applications. The organization creates separate user id which starts from 1001 for each Unix user apart from their employee id. Unix system has two kinds of users - Super and Ordinary. It is decided that Super users id will be prime number and ordinary users id will be non prime. You need develop this kind of Java OOPS based application where each user’s information will be accepted in a comma(,) separate format e.g: “72562,Mike,Super” where 1st one is the employee id,2nd is the name and 3rd is the type of user, then and user id will be generated based on type of user i.e. primitive for Super and non-primitive for ordinary user starting from 1001.Number should be sequential series in both the cases. All the users will be maintained in collections.

You need create

1. class UnixUser with private member data

String userId,

String employeeId,

String username,

String userType,

Create getter/setter methods and constructors.

override toString() in String.format("%-10s %-10s %-20s %-10s").

Override the hashCode and equals methods.

1. class Main with methods
2. public static boolean checkPrime(int no ) : It will check where parameterized value is a prime number then return true else return false.

b. method public static void main(String [] arg): It will accept number of unix-users id to generate, then accepts all users details namely employee id, name and type of unixuser to create, in already given format. Two employees cannot have same employee id for that reason you need to override equals and hashCode methods of UnixUser class. After creation each user id (which starts from 1001), you need to display all unix users details including user id in a tabular format .

Code:

Junit Testing

**import** java.io.File;

**import** java.io.FileWriter;

**import** java.io.IOException;

**public** **class** TestUtils {

**public** **static** File *businessTestFile*;

**public** **static** File *boundaryTestFile*;

**public** **static** File *exceptionTestFile*;

**static** {

*businessTestFile* = **new** File("./output\_revised.txt");

*businessTestFile*.delete();

*boundaryTestFile* = **new** File("./output\_boundary\_revised.txt");

*boundaryTestFile*.delete();

*exceptionTestFile* = **new** File("./output\_exception\_revised.txt");

*exceptionTestFile*.delete();

}

**public** **static** **void** yakshaAssert(String testName, Object result, File file) **throws** IOException {

System.***out***.println("\n" + testName + "=" + result);

FileWriter writer = **new** FileWriter(file,**true**);

writer.append("\n" + testName + "=" + result);

writer.flush();

writer.close();

}

**public** **static** String currentTest() {

**return** Thread.*currentThread*().getStackTrace()[2].getMethodName();

}

}

**import** **static** org.junit.Assert.\*;

**import** org.junit.Test;

**import** **static** java9.mod.modapp2.TestUtils.\*;

**public** **class** MainTest {

@Test

**public** **void** testCheckPrime() **throws** Exception {

*yakshaAssert*(*currentTest*(),(Main.*checkPrime*(2)?"true":"false"),*businessTestFile*);

*yakshaAssert*(*currentTest*(),(Main.*checkPrime*(9)?"true":"false"),*businessTestFile*);

}

}

Test Data1

**Enter number of users id to create:**

**7**

**Enter all users details one line at a time**

**2145,Jon,Ordinary**

**9613,Mary,Super**

**9451,Ann,Super**

**8741,Eric,Ordinary**

**8546,Julia,Ordinary**

**4512,Leo,Super**

**7418,Eve,Ordinary**

**User Id Employee Id User Name User Type**

**1009 4512 Leo Super**

**1013 9451 Ann Super**

**1019 9613 Mary Super**

**1001 2145 Jon Ordinary**

**1002 8546 Julia Ordinary**

**1003 8741 Eric Ordinary**

**1004 7418 Eve Ordinary**

Learning outcome: Participant could able to use Stream API filter, collection to stream and vice-versa.