رمزگشایی سلطان

جاسوس های سلطان در کشور های همسایه اطلاعات را به صورت ماتریس هایی رمزنگاری شده به سلطان میرسانند. رمزگشایی دستی این اطلاعات برای سلطان کاری بسی دشوار است بنابراین او از شما خواسته تا اطلاعات را با استفاده از کلید سری جاسوس ها رمز گشایی کنید. کلید سلطان این دفعه چاپ بر اساس قطرهای موازی با قطر فرعی از گوشه بالا سمت چپ است.

ورودی

در خط اول عدد n داده میشود که اندازه ماتریس n*n ما است. در خطوط بعدی اعداد مربوط به اعضا ماتریس داده میشوند.

خروجی

پیام رمزگشایی شده را برای سلطان چاپ کنید

مثال

ورودی نمونه ۱

```
3
1 2 3
4 5 6
7 8 9
```

خروجی نمونه ۱

```
1 2 4 3 5 7 6 8 9
```

ورودی نمونه ۲

```
4
1 2 3 4
```

```
5 6 7 8
9 10 11 12
13 14 15 16
```

خروجی نمونه ۲

```
1 2 5 3 6 9 4 7 10 13 8 11 14 12 15 16
```

دنیای ماتریس

سلطان پس از دیدن فیلم ماتریکس، به واسطه نفوذی که داشت توانست ملاقاتی با مورفیوس داشته باشد و ۲ قرص قرمز و آبی را از او بگیرد. اما با توجه به اینکه سلطان به دستور کسی گوش نمی‌دهد و همواره کار خودش را انجام می‌دهد تصمیم گرفت تا قرص قرمز و آبی را ترکیب کرده و سپس بخورد. این کار باعث شده است تا سلطان به زندانی از ماتریس‌های ریاضیاتی منتقل شود. حال سلطان از شما که برنامه‌نویس معتمد او هستید می‌خواهد تا ماشین حسابی ماتریسی طراحی کنید که به او در فرار از این زندان کمک کند.

کلاس Matrix:

کلاس `Matrix` دارای `property` های زیر می‌باشد :

```
int size;
int [][] data;
```

متغیر `size` اندازه ماتریس `n*n` مورد نظر را نگهداری می‌کند و `data` نشان دهنده اعضای ماتریس است.

کلاس `Matrix` دارای `constructor` زیر میباشد :

```
public Matrix(int size, int[][] data);
```

سازنده بالا با دریافت `size` و `data` ماتریس جدیدی میسازد.

کلاس `Matrix` دارای `method` های زیر می‌باشد :

```
public Matrix add(Matrix m1);
```

تابع بالا ماتریس `m1` را با ماتریس فعلی ما جمع می‌کند و ماتریس `m2` را که حاصل جمع داخل آن ذخیر شده است باز می‌گرداند. اگر 2 ماتریس از یک اندازه نباشند تابع، `null` باز می‌گرداند.

```
public Matrix multiply(Matrix m1);
```

تابع بالا مانند تابع `add` عمل می‌کند اما به جای جمع کردن ۲ ماتریس آن ۲ را در یک‌دیگر ضرب کرده و باز حاصل را در ماتریس `m2` ذخیره و آن را باز می‌گرداند.

```
1 | public String printMatrix();
```

تابع بالا اطلاعات ماتریس را بر می‌گرداند. شکل رشته خروجی در بخش خروجی نمونه نشان داده شده است.

```
1 | public boolean equals(Matrix m);
```

تابع بالا ۲ ماتریس را مقایسه می‌کند در صورتی که برابر بودند `true` و در صورتی که 2 ماتریس برابر نبودند `false` بر می‌گرداند.

```
1 | public static boolean isSymmetric(Matrix m);
```

تابع `static` بالا با دریافت یک ماتریس برسی می‌کند که ماتریس نسبت به قطر فرعی قرینه است یا خیر. اگر قرینه بود `true` و در غیر این صورت `false` بر می‌گرداند. مثلاً ماتریس زیر نسبت به قطر فرعی قرینه است.

```
1 2 3
8 3 2
3 8 1
```

```
1 | public static int det(Matrix m);
```

تابع `static` بالا دترمینان ماتریس `m` را خروجی می‌دهد.

اگر کد شما درست باشد با دادن کد ورودی زیر، خروجی زیر باید چاپ شود.

ورودی:

```java
int[][] m1 = {{1,2,3}, {4,5,6}, {7,8,9}};
int[][] m2 = {{-1,2,6}, {3,4,2}, {7,3,-1}};
int[][] m3 = {{1,2,3,4}, {6,7,4,3}, {8,4,7,2}, {4,8,6,1}};
Matrix mm1 = new Matrix(3,m1);
Matrix mm2 = new Matrix(3,m2);
Matrix mm3 = new Matrix(4,m3);
System.out.println(mm1.printMatrix());
System.out.println(mm1.multiply(mm1).printMatrix());
System.out.println(mm1.add(mm2).printMatrix());
System.out.println(Matrix.det(mm3));
System.out.println(Matrix.isSymmetric(mm3));
System.out.println(mm1.equals(mm2));
System.out.println(mm1.add(mm3));
```

خروجی:

```
[{1 ,2 ,3}{4 ,5 ,6}{7 ,8 ,9}]
[{30 ,36 ,42}{66 ,81 ,96}{102 ,126 ,150}]
[{0 ,4 ,9}{7 ,9 ,8}{14 ,11 ,8}]
633
true
false
null
```

اضافه کردن `constructor` و `method` های کمکی برای پیاده‌سازی توابع بالا مانعی ندارد.

استفاده از کتابخانه های جدا برای انجام عملیات های ماتریسی مجاز نیست.

آن‌چه باید آپلود کنید

پس از پیاده‌سازی موارد خواسته‌شده، فایل `Matrix.java` را آپلود کنید.

گلستان کوچک

برای حل این سوال حق استفاده از sort خود زبان را ندارید

سلطان که از آزادی دانشجویان در انتخاب و حذف واحد ها کلافه شده است تصمیم گرفته تا دانشگاهی جدید به وجود آورد که در این دانشگاه دانشجویان هیچ حق انتخابی ندارند و بدون هیچ دسترسی به کلاس ها اضافه و یا از آنها حذف میشوند. ساخت این دانشگاه نیازمند باز طراحی سیستم گلستان است، حال سلطان از شما میخواهد این سیستم جدید را که تحت عنوان گلستان کوچک شناخته میشود طراحی کنید. گلستان کوچک سلطان از کلاس های زیر تشکیل شده است :

کلاس Student:

کلاس `Student` دارای `property` های زیر میباشد :

```
int studentId;
double gradeMean;
Course[] courses;
```

در متغیر `studentId` شماره دانشجویی مربوط به دانشجو ذخیره میشود ، `gradeMean` میانگین نمرات دانشجو را نگهداری میکند و در پایان آرایه `courses` مسئول نگهداری تمام کلاس هایی است که دانشجوی مورد نظر در آنها ثبت نام کرده است. هر دانشجو در حداکثر 5 کلاس مختلف میتواند ثبت نام کند.

کلاس `Student` دارای `constructor` زیر میباشد :

```
public Student(int id, double gradeMean);
```

سازنده بالا با گرفتن `gradeMean` و `studentId` دانشجوی جدیدی میسازد.

کلاس `Student` دارای `method` های زیر میباشد :

```
public String printCourses();
```

تابع بالا نام Course هایی است که دانشجو داخل آنها عضو شده را بر میگرداند است.اگر دانشجو داخل کلاسی عضو نبود این تابع باید there is no course را برگرداند.

برای دیدن شکل رشته بازگردانده شده به بخش مثال مراجعه کنید.

کلاس Course:

کلاس Course دارای property های زیر میباشد :

```
String name;
Student[] students;
int numStudents;
int maxStudents;
```

متغیر name نام Course ما را ذخیره میکند متغیر students مسئول نگهداری لیست **مرتب شده بر اساس نمره** از دانشجویان کلاس است و در پایان numStudents نشان دهنده تعداد دانشجویان حاضر در کلاس و maxStudents نشان دهنده ظرفیت کل کلاس است.

کلاس Course دارای constructor زیر میباشد :

```
public Course(String name);
```

سازنده بالا با گرفتن name یک دوره جدیدی میسازد.

کلاس Course دارای method های زیر میباشد :

```
public boolean addStudent(Student s);
```

تابع بالا دانشجوی جدید را به لیست دانشجو های دوره اضافه میکند. طرفیت لیست دانشجو های دوره در ابتدا 5 است اما هر سری در صورت پر شدن لیست، طرفیت 2 برابر خواهد شد. همچنین این تابع دانشجو ها را به صورت صعودی و بر اساس میانگین نمراتشان در لیست قرار میدهد. بعد از اضافه کردن یک دانشجو به دوره ، دوره مورد نظر داخل لیست دوره های دانشجو قرار خواهد گرفت. در صورتی که دانشجو قبلا به دوره افزوده شده باشد و یا دانشجو تعداد مجاز درس را قبلا اخذ کرده باشد تابع بالا false بر میگرداند و در صورت موفق بودن عملیات true بر میگرداند.

```
public Student deleteStudent(int studentID);
```

تابع بالا دانشجو به شماره دانشجویی `studentId` را از لیست دانشجو های دوره حذف میکند و دانشجوی حذف شده را بر میگرداند. در صورتی که شماره دانشجویی مورد نظر در دوره وجود نداشته باشه تابع بالا `null` باز میگرداند. بعد از عملیات بالا دوره هم باید از لیست دوره های دانشجو حذف شود.

```
public String  printStudents();
```

تابع بالا لیست دانشجو های موجود در دوره را بر میگرداند. در صورتی که دوره دانشجویی نداشته باشد پیام there is no student برگردانده میشود.

برای دیدن شکل رشته برگردانده شده به بخش مثال مراجعه کنید.

```
public String findBest();
```

تابع بالا شماره دانشجویی و میانگین نمره دانشجویی که بالاترین میانگین نمره را دارد بر میگرداند. در صورتی که دوره دانشجویی نداشته باشد پیام there is no student in the class برگردانده میشود.

```
public int size();
```

تابع بالا تعداد دانشجو های حاضر در دوره را بر میگرداند.

```
public Student findId(int id)
```

تابع بالا دانشجو با شماره دانشجویی `id` را بر میگرداند. اگر دانشجو با شماره دانشجویی `id` در دوره موجود نبود `null` برگردانده میشود.

```
public int getMaxStudents();
```

تابع بالا مقدار `maxStudents` را بازمیگرداند.

اضافه کردن `method` های کمکی برای پیاده سازی `method` های اصلی مانعی ندارد. در صورت درست بودن کد شما با اجرای قطعه کد ورودی باید خروجی زیر چاپ شود.

ورودی

```
Course ap = new Course("ap");
Course math = new Course("math");
Course ds = new Course("ds");
Student s1 = new Student(1 ,12.4);
Student s2 = new Student(2 , 7.8);
Student s3 = new Student(3 , 19);
ap.addStudent(s1);
math.addStudent(s1);
ds.addStudent(s1);
ap.addStudent(s2);
math.addStudent(s2);
ap.addStudent(s3);
ds.addStudent(s3);
System.out.println(ds.printStudents());
System.out.println(ap.printStudents());
ap.deleteStudent(1);
System.out.println(ap.printStudents());
System.out.println(ap.deleteStudent(1));
System.out.println(ap.size());
System.out.println(ap.getMaxStudents());
System.out.println(s1.printCourses());
System.out.println(math.findBest());
System.out.println(math.findId(1).studentId);
System.out.println(math.findId(3));
ap.deleteStudent(2);
ap.deleteStudent(3);
System.out.println(ap.printStudents());
System.out.println(ap.findBest());
```

خروجی

```
1 3
2 1 3
2 3
```

```
null
2
5
math ds
id : 1 grade : 12.4
1
null
there is no student
there is no student in the class
```

در پایان فایل های Course.java و Student.java را زیپ کرده و آپلود کنید.

شبکه اجتماعی

در این سوال می‌خواهیم یک شبکه اجتماعی بسیار ساده پیاده سازی کنیم.

پروژه اولیه را از این لینک دانلود کنید. می‌توانید به کد propertyها و method ها بر اساس نیاز اضافه کنید.

کلاس User

این کلاس شامل نام فرد، پست‌ها، فالوورها و فالووینگ‌های او است.

- سازنده این کلاس شامل نام فرد است.
- متد follow یک user می‌گیرد اگر فرد آن را قبلاً فالو نکرده باشد، این متد مانند فالو عمل می‌کند و در غیر این صورت مانند آنفالو عمل می‌کند.
- متد post، محتوای پست را می‌گیرد و یک پست جدید درست می‌کند و آن را به پست‌های کاربر اضافه می‌کند.
- متد like، یک پست می‌گیرد و آن را لایک می‌کند. کارکرد آن شبیه متد فالو است. (یعنی اگر پست قبلا توسط فرد لایک شده باشد، با اجرای متد لایک، پست آنلایک می‌شود!)
- متد comment، یک رشته و یک پست می‌گیرد و برای آن پست کامنتی که محتوای آن رشته مذکور است می‌گذارد.
- متد های getFollowings ،()getFollowers ،()getPosts(),()getName به ترتیب از راست به چپ، آرایه‌ای از فالوییینگ‌ها، فالوورها، پست‌های فرد و نام فرد را برمی‌گرانند.

کلاس Post

این کلاس شامل محتوای پست، اکانت پست کننده، لایک کننده‌های این پست و کامنت‌های آن است.

- سازنده این کلاس شامل محتوای پست (در اینجا محتوا را از نوع رشته در نظر می‌گیریم) و اکانت فرد پست کننده است.
- این کلاس دارای متد addComment است که با گرفتن یک کامنت، آن را به کامنت های پست اضافه می‌کند.

- متدهای getComments() ، getLikers() به ترتیب از راست به چپ، آرایه‌ای از کامنت‌ها و لایک کننده‌های پست را برمی‌گرانند.

کلاس Comment

- سازنده این کلاس شامل محتوای کامنت (در اینجا محتوا را از نوع رشته در نظر می‌گیریم) و اکانتی که کامنت را می‌گذارد، است.

مثال:

```
User u1 = new User("Harry");
User u2 = new User("Liam");
User u3 = new User("Amy");
User u4 = new User("Bob");
u1.follow(u2);
u1.follow(u4);
u2.follow(u3);
u4.follow(u2);
for (int i =0;i<u2.getFollowers().length ;i++ ) {
    System.out.println(u2.getFollowers()[i].getName());
}
u1.follow(u2);
for (int i =0;i<u2.getFollowers().length ;i++ ) {
    System.out.println(u2.getFollowers()[i].getName());
}
u1.post("This is my first Post!");
u3.like(u1.getPosts()[0]);
u2.like(u1.getPosts()[0]);
for (int i =0;i<u1.getPosts()[0].getLikers().length ;i++ ) {
    System.out.println(u1.getPosts()[0].getLikers()[i].getName());
}
u4.comment("welcome!", u1.getPosts()[0]);
u3.comment("nice!",u1.getPosts()[0] );
for (int i =0;i<u1.getPosts()[0].getComments().length ;i++ ) {
    System.out.println(u1.getPosts()[0].getComments()[i].toString());
}
```

```
Harry
Bob
Bob
```

```
Amy
Liam
Bob: wellcome!
Amy: nice!
```

آن‌چه باید آپلود کنید

پس از پیاده‌سازی موارد خواسته‌شده، یک فایل زیپ آپلود کنید که وقتی آن را باز می‌کنیم، با فایل‌های زیر مواجه شویم:

```
Comment.java
User.java
Post.java
```

سنگ‌کاغذقیچی (Git)

صبا و *نیما* مشغول توسعه پروژه بازی سنگ‌کاغذقیچی خود هستند.

ریپازیتوری پروژه اولیه را از این لینک دانلود کنید.

پروژه آن‌ها ۴ فایل مهم دارد:

۱. فایل `main.py` : این فایل برای مدیریت کلیت بازی‌ست؛ زمان شروع بازی، چاپ نتیجه بازی و... توسط این فایل کنترل می‌شوند.
۲. فایل `user_interface.py` : این فایل برای ورودی گرفتن حرکت کاربر در هر نوبت است.
۳. فایل `computer_choice.py` : این فایل برای تعیین حرکت کامپیوتر در هر نوبت است، حرکت کامپیوتر در تمامی مراحل بازی رندوم است.
۴. فایل `check_results.py` : این فایل برای تعیین نتیجه هر نوبت بازی‌ست.

در پروژه‌ی سنگین‌ِ بازی سنگ‌کاغذقیچی، تقسیم وظایف به این صورت است:

- پیاده‌سازی فایل‌های `main.py` و `check_results.py` وظیفه *صبا* است.
- پیاده‌سازی فایل‌های `computer_choice.py` و `user_interface.py` وظیفه *نیما* است.

صبا فایل `main.py` را از قبل پیاده سازی کرده و روی برنچ `master` قرار داده است.

آن‌ها متوجه شدند که نمی‌توانند جدا از هم روی برنچ `master` کار کنند. برای همین تصمیم گرفتند از این به بعد هر کسی تغییرات خود را روی برنچ خود اعمال کند.

یعنی *صبا* فایل `check_results.py` را در برنچ `feature/check_results` و *نیما* فایل `computer_choice.py` را در برنچ `feature/computer_choice` و فایل `user_interface.py` را در برنچ `feature/user_interface` پیاده‌سازی کند.

اما آن‌ها که از گیت چیزی نمی‌دانند از شما جهت انجام بخش‌های گیتی پروژه درخواست کمک دارند. در جعبه‌های زیر به بررسی جزئیات انجام تمرین می‌پردازیم.

▼ **انتقال فایل `check_results.py` به پروژه**

برای انتقال این فایل روی پروژه باید به صورت زیر عمل کنید:

۱. ابتدا مطمئن شوید روی برنچ `master` قرار دارید و اگر نبودید با کمک `checkout` به آن منتقل شوید.

۲. یک برنچ با نام `feature/check_results` ساخته و روی آن `checkout` کنید.

۳. یک فایل با نام `check_results.py` بسازید و محتوای زیر را درون آن قرار دهید:

```
def check_results(choices, player, computer):
    '''
    function that checks who won.
    returns string
    '''
    if player == computer:
        return 'It\'s a tie'
    elif (player == 0 and computer == len(choices) - 1) or (
            player > computer and not (player == len(choices) - 1 and compu
        return 'Player Won'
    return 'Player Lost'
```

۴. فایل را ادد کنید و کامیت بزنید.

▼ **انتقال فایل `user_interface.py` به پروژه**

برای انتقال این فایل روی پروژه باید به صورت زیر عمل کنید:

۱. ابتدا مطمئن شوید روی برنچ `master` قرار دارید و اگر نبودید با کمک `checkout` به آن منتقل شوید.

۲. یک برنچ با نام `feature/user_interface` ساخته و روی آن `checkout` کنید.

۳. یک فایل با نام `user_interface.py` بسازید و محتوای زیر را درون آن قرار دهید:

```
def user_interface(options):
    '''
    function presenting options and asking for player feedback
    returns integer.
    '''
    for index, option in enumerate(options):
        print(f'{index} = {option}')

```

```
    user_input = int(input('What do you choose? '))
    return user_input
```

۴. فایل را `add` و کامیت کنید.

▼ **انتقال فایل `computer_choice.py` به پروژه**

برای انتقال این فایل روی پروژه باید به صورت زیر عمل کنید:

۱. ابتدا مطمئن شوید روی برنچ `master` قرار دارید و اگر نبودید با کمک `checkout` به آن منتقل شوید.

۲. یک برنچ با نام `feature/computer_choice` ساخته و روی آن `checkout` کنید.

۳. یک فایل با نام `computer_choice.py` بسازید و محتوای زیر را درون آن قرار دهید:

```
from random import randint

def computer_choice(content):
    '''
    function that generates a random number based on the available options.
    returns random int
    '''
    computer_chose = randint(0, len(content) - 1)
    return computer_chose
```

۴. فایل را `add` و کامیت کنید.

در نهایت، با کمک `checkout` به برنچ `master` منتقل شوید.

آن‌چه باید آپلود کنید

پس از انجام مراحل خواسته‌شده، محتویات پوشه اصلی ریپازیتوری را زیپ کرده و آپلود کنید.

لیست پیوندی

لیست یکی از ساختمان داده هایی است که برای نگهداری عناصر استفاده میشود. یکی از انواع لیست ، لیست پیوندی یا همان LinkedList معروف است که احتمالا کمابیش با آن آشنا هستید و در ادامه درس با آن بیشتر آشنا خواهید شد.

در این تمرین از شما میخواهیم که چیزی شبیه LinkedList که عدد صحیح نگه می‌دارد را پیاده سازی کنید. داکیومنت‌های کلاس LinkedList را می‌توانید اینجا بخوانید.

شما باید کلاس LinkedList را **خودتان** و با توجه به توضیحات زیر پیاده سازی کنید.

برنامه شما باید شامل همه متد های زیر باشد.

```
/***
* adds the specified element to the end of the list
***/
public boolean add(Integer element) ;

/***
* adds the specified element
* at the specified index of your list
***/
public void add(int index , Integer element);

/***
* append all of the elements
* in the specified linkedlist to the end of this list
***/
public boolean addAll(LinkedList linkedlist);


/**appends all of the elements
* in the specified linkedlist starting at the specified index
***/
public boolean addAll(int index, LinkedList linkedlist);

/***
* inserts a specified element at the beginning of this list
***/
```

```
27  public void addFirst(Integer element);
28
29  /***
30  * appends the specified element at the end of this list
31  ***/
32  public void addLast(Integer element);
33
34  /***
35  * Removes all of the elements from this list. The list will be empty after th
36  ***/
37  public void clear();
38
39  /***
40  * Returns `true` if this list contains the specified element.
41  ***/
42  public boolean contains(Integer i);
43
44  /***
45  * Returns the element at the specified position in this list.
46  ***/
47  public Integer get(int index);
48
49  /***
50  * Returns the index of the first occurrence of the specified element in this
51  ***/
52  public int indexOf(Integer i);
53
54  /***
55  * Retrieves and removes the head (first element) of this list.
56  ***/
57  public Integer remove();
58
59  /***
60  * Removes the element at the specified position in this list.
61  * Shifts any subsequent elements to the left (subtracts one from their indice
62  *  Returns the element that was removed from the list.
63  ***/
64  public Integer remove(int index);
65
66  /***
67  * Returns the number of elements in this list.
68  ***/
69  public int size();
```

```
70
71  /***
72  * Returns an array containing all of the elements in this list in proper sequ
73  ***/
74  public Integer[] toArray();
75
76  /***
77  * Returns true if this list contains no elements.
78  ***/
79  public boolean isEmpty();
```

کد نمونه

```
1   public class Main {
2       public static void main(String[] args) {
3           LinkedList linkedList1 = new LinkedList();
4           linkedList1.add(1);
5           linkedList1.add(3);
6           System.out.println(linkedList1.size()); // 2
7           linkedList1.add(1, 2);
8           Integer list[] = linkedList1.toArray();
9           for (int i = 0; i < list.length; i++) {
10              System.out.println(list[i]); // 1 2 3
11          }
12          System.out.println(linkedList1.contains(4)); // false
13          System.out.println(linkedList1.indexOf(1)); // 0
14          LinkedList linkedList2 = new LinkedList();
15          linkedList2.addAll(linkedList1);
16          System.out.println(linkedList2.size()); // 3
17          linkedList2.clear();
18          System.out.println(linkedList2.isEmpty()); // true
19      }
20  }
```

توجه مهم

این سوال به صورت اتوماتیک و متفاوت با سوالات دیگری که تا کنون دیده‌اید داوری می‌شود. برای داوری از JUnit استفاده شده است. برای داوری صحیح، موارد گفته شده مثل نام متدها و کانستراکتور و غیره را دقیقا در کلاس قرار دهید.

نهایتا یک فایل زیپ اپلود کنید که فقط شامل LinkedList.java باشد.

در صورتی که به عبارت could not run 9 tests. برخوردید به معنی کامپایل ارور و رعایت نکردن موارد فوق است.

محتویات فایل زیر مثل تصویر زیر باشد:

```
.
└── LinkedList.java

0 directories, 1 file
```

اندرومدا (امتیازی)

آیا به نجوم علاقه دارید؟ تا کنون به آسمانِ شب نگاه کرده‌اید؟ تعداد ستاره‌های کهکشان چقدر است؟ تعداد مولکول‌های کهکشان اندرومدا چقدر است؟ آیا این تعداد در integer جا می‌شود؟ در long چطور؟ احتمالا جواب منفی است. برای نگه‌داشتن اعداد بسیار بزرگ، برنامه‌نویسان جاوا از کلاس‌های BigDecimal و یا BigInteger استفاده می‌کنند.

داکیومنت‌های کلاس BigInteger را می‌توانید اینجا بخوانید.

حالا ناسا برای نیازهای خاص و محرمانه خودش، از شما خواسته که کلاس BigInteger را با خواص زیر **خودتان از ابتدا با استفاده از لیست پیوندی** پیاده کنید. استفاده از ۲ کلاس BigDecimal و BigInteger آمادهٔ جاوا و هر کد آماده‌ای برای کار با اعداد بزرگ نمره‌ای ندارد.

چیزهایی که باید رعایت کنید:

- نام کلاس شما باید BigInteger باشد.
- سازندهٔ گفته‌شده
- متدهای گفته شده
- فیلدهای استاتیکی که ذکر می شود
- کلاس باید **immutable** باشد، یعنی بعد از construct شدن، فیلدهایش هیچ تغییری نکند.
- برای پیاده‌سازی فقط از لیست پیوندی استفاده کنید. (یعنی نه آرایه و نه String)
- می‌توانید از کلاس لیست پیوندی جاوا (java.util.LinkedList) استفاده کنید ولی استفاده از لیست پیوندی خودتان نمره امتیازی دارد.

سازندهٔ زیر را عینا پیاده‌سازی کنید.

```
/**
* Translates the decimal String representation of a
* BigInteger into a BigInteger.
*/
public BigInteger(String val)
```

متدهای زیر را دقیقا پیاده‌سازی کنید.

```java
/**
* Returns a BigInteger whose value is the absolute
* value of this  BigInteger.
*/
public BigInteger abs()

/**
* Returns a BigInteger whose value is
* (this + val).
*/
public BigInteger add(BigInteger val)

/**
* Returns a BigInteger whose value is (this - val).
*/
public BigInteger subtract(BigInteger value)

/**
* Returns a BigInteger whose value is (this * val).
*/
public BigInteger multiply(BigInteger value)

/**
* Compares this BigInteger with o for equality.
* if o is not BigInteger, return false
*/
public boolean equals(Object o)

/**
* Returns the decimal String representation
* of this BigInteger.
*/
public String toString()
```

فیلدهای استاتیک زیر باید در کلاس شما وجود داشته باشند و مقداردهی شده باشند.

```java
public static final BigInteger ONE;
public static final BigInteger TEN;
public static final BigInteger ZERO;
```

توجه

- تمامی توابع از جمله جمع، نباید چیزی درون شی را عوض کنند، مثلا a.add(b) باعث می‌شود مقدار a+b برگردانده شود و a و b خودشان هیچ تغییری نمی‌کنند.
- همان‌طور که می‌دانید یکی از مزایای لیست‌پیوندی این است که تمام داده‌ها را پشت‌سر هم داخل رم نگه نمی‌دارد و این برای زمانی که داده‌های زیادی داریم مثل این سوال، مزیت مهمی محسوب می‌شود.
- از لیست‌پیوندی جاوا می‌توانید مثل کد زیر استفاده کنید:

```
/**
* example of using linked list of Integer
* you should import java.util.LinkedList for this
*/
LinkedList<Integer> myList = new LinkedList<Integer>();
 myList.add(1);
 myList.add(2);
 myList.add(3);
 myList.addFirst(0);
 int len = myList.size();
System.out.println(myList.get(len-1)); // print last item:0
```

کد نمونه

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

        /**
            example of construct by String
        */
        BigInteger b1 = new BigInteger("+1234");
        System.out.println(b1.toString()); //1234

        BigInteger b2 = new BigInteger("3333");
        System.out.println(b2.toString()); //3333

        BigInteger b3 = new BigInteger("-10");
        System.out.println(b3.toString()); //-10

```

```
15
16          /**
17          * example of add
18          */
19          BigInteger bz = b3.add(BigInteger.TEN);
20          System.out.println(bz.toString()); // -10 + 10 = 0
21
22          /**
23          * example of other operations
24          */
25          System.out.println( b2.add(b3.abs()).toString() ); //3343
26          //b3.abs() will compute to 10
27          // b2+10 will be 3343
28
29          System.out.println(new BigInteger("7").multiply(new BigInteger("-4"))
30
31          System.out.println(new BigInteger("74").subtract(new BigInteger("-34"
32
33
34          System.out.println(b1.equals(b2)); // b1!=b2 ==> false
35
36          System.out.println(bz.equals(BigInteger.ZERO)); // both zero ==> true
37
38          System.out.println(bz.equals(0)); // return false, because 0 is Integ
39      }
```

کد نمونه و امضای توابع را می‌توانید از این لینک دانلود کنید.

توجه مهم: برای اینکه کد شما کامپایل شود شما باید امضای توابع را حفظ کنید، در صورتی که کوچک‌ترین تغییری بدهید، کد شما کامپایل نشده و نمره‌ای دریافت نمی‌کنید.

همچنین پیشنهاد اکید می‌شود همین فایل BigInteger را دانلود کرده و پیاده‌سازی‌ها را داخلش انجام دهید.

ضمنا پیاده‌سازی تابع equals و toString را هم در اولویت قرار دهید چون تست این سوال بر اساس این ۲ تابع کار می‌کند.

یک فایل زیپ اپلود کنید که فقط شامل BigInteger.java باشد. در صورتی که از لینک لیست خودتان استفاده می‌کنید می‌توانید (اختیاری) یک فایل LinkedList.java هم در فایل زیپ داشته باشید.

در صورتی که به عبارت could not run 12 tests. برخوردید به معنی کامپایل ارور و رعایت نکردن موارد فوق است.

محتویات فایل زیپ به یکی از ۲ حالت زیر باید باشد:

```
.
├── BigInteger.java
└── LinkedList.java

0 directories, 2 files
```

```
.
└── BigInteger.java

0 directories, 1 file
```