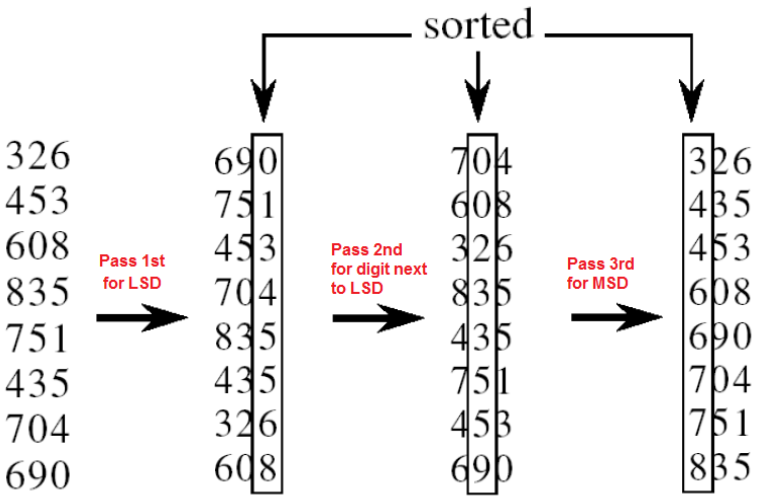
بسمه تعالی

دانشگاه فردوسی مشهد

سوال Radix Sort (نسخه‌ی جاوا):

در این تمرین الگوریتم مرتب‌سازی مبنایی پیاده‌سازی می‌شود. در الگوریتم مرتب‌سازی مبنایی فرض می‌کنیم که داده‌ها دارای کلیدی عددی هستند و در مبنای مشخصی (مثلا ۲، ۱۰، ۱۶ و ...) نمایش داده شده‌اند. الگوریتم به این شکل عمل می‌کند که از رقم کم‌ارزش شروع می‌کند و دادگان را با الگوریتم مرتب‌سازی شمارشی (counting sort)، که الگوریتمی پایدار است، رقم به رقم مرتب می‌کند. در شکل زیر مثالی از اجرای این الگوریتم را مشاهده می‌کنید.



در اینجا نسخه‌ای از الگوریتم مرتب‌سازی مبنایی را در نظر می‌گیریم که در آن هر r رقم دهدهی معادل یک رقم در مرتب‌سازی مبنایی در نظر گرفته می‌شود. به عبارت دیگر مرتب‌سازی مبنایی در مبنای 10^r انجام می‌شود. برای نمایش اعداد دهدهی با طول دلخواه، کلاس UnboundedInteger تعریف شده است. در هر نمونه از این کلاس یک عدد صحیح دهدهی به صورت یک رشته‌ی متنی ذخیره می‌شود.

اولین متدی که لازم است پیاده‌سازی شود متد getDigit از کلاس UnboundedInteger است.

متد getDigit در فایل UnboudedInteger.java:

```
public int getDigit (int r, int j){
    //Write your code here.
}
```

این متد با فرض اینکه هر r رقم دهدهی یک رقم در مبنای 10^r در نظر گرفته می‌شوند، رقم j ام را (در مبنای 10^r) برای یک شیء از نوع UnboudedInteger برمی‌گرداند. دقت کنید که شمارش j از ۱ شروع می‌شود، یعنی برای به دست آوردن اولین رقم، j=1 است.

با توجه به اینکه در مرتب‌سازی مبنایی کلید باید یک عدد صحیح نامنفی باشد، فرض می‌کنیم که اشیائی که می‌خواهیم مرتب کنیم رابط IntegerKeyType را پیاده‌سازی کرده‌اند. این رابط چنین تعریف شده است:

```
public interface IntegerKeyType {
    public UnboundedInteger getKey();
}
```

همچنین لازم است که متدهای sort، calculateD و countingSort از کلاس RadixSort تکمیل شوند. شرح این متدها چنین است:

```
public void sort(T[] array) {
    //Write your code here.
}
```

این متد یک آرایه از نوع عمومی T را به عنوان ورودی دریافت می‌کند و آن را با روش Radix Sort و با کمک توابع calculateD, calculateR و countingSort به صورت صعودی مرتب می‌کند. T یک نوع عمومی (generic) است که رابط IntegerKeyType را پیاده‌سازی می‌کند. پارامتر های این تابع به شرح زیر است:

array: آرایه داده های ورودی

```
public int calculateD(T[] array, int r){
    //Write your code here.
}
```

در آرایه ورودی، key هر داده حداکثر یک عدد d رقمی است. الگوریتم مرتب سازی شمارشی، به ازای d رقم موجود، d بار اجرا می‌شود. برای محاسبه‌ی d و استفاده از آن در الگوریتم Radix sort ، تابع calculateD را پیاده سازی کنید. پارامتر های این تابع به شرح زیر است:

array: آرایه داده‌های ورودی

r: مبنای در نظر گرفته شده 10^r است. به عبارت دیگر هر r رقم ده‌دهی، یک رقم را در مبنای در نظر گرفته شده برای مرتب‌سازی مبنایی نشان می‌دهد.

```java
public void countingSort(T[] A, int k, int r, int j){
    //Write your code here.
}
```

این متد آرایه‌ی ورودی A را با الگوریتم مرتب‌سازی شمارشی مرتب می‌کند. ورودی های این تابع به شرح زیر است:

A: آرایه ورودی از نوع عمومی T که رابط IntegerKeyType را پیاده‌سازی می‌کند.

k: بیشترین مقدار ممکن برای یک عدد r رقمی در مبنای 10. یعنی $10^r - 1$

r: هر r رقم ده‌دهی یک رقم در نظر گرفته شوند.

j: رقمی که در حال مرتب سازی آن هستیم. به عبارت دیگر مرتب‌سازی شمارشی تنها بر روی رقم j ام آرایه انجام می‌شود.

متد getDigit از کلاس UnboundedInteger می‌تواند اینجا کاربرد داشته باشد.

```java
public int calculateR(int n){
    int r = (int) (Math.log10(n + 1));
    if (r > 4)
        r = 4;
    else if (r < 1)
        r = 1;
    return r;
}
```

مطابق توضیحات درس، در این تابع، مقدار بهینه r با توجه به طول آرایه ورودی، محاسبه می‌شود. با توجه به اینکه مرتب‌سازی شمارشی به آرایه‌ای به طول 10^r نیاز دارد، مقدار r در اینجا به مقادیر ۱ تا ۴ محدود شده است.

در این تمرین نحوه ارزیابی به شرح زیر است:

۱. TestCalculateD: در این تست، تابع calculateD پیاده‌سازی شده توسط دانشجو، مورد ارزیابی قرار می‌گیرد. این تست شامل ٪۵ نمره کل است.
۲. TestGetDigit: در این تست، تابع getDigit بررسی می‌شود. در این تست، مقادیر مختلف ورودی، به تابع داده می‌شود و انتظار می‌رود بر اساس r,j ورودی، بخش مورد نظر از شی UnboundedInteger برگردانده شود. این تست شامل ۱۵٪ نمره است.
۳. TestSortingRandomNumbers: در این تست بررسی می‌شود که آیا عملیات مرتب سازی (sort) به درستی انجام شده است یا خیر. در این تست چندین بار آرایه هایی با طول متفاوت به صورت تصادفی (random) مقداردهی می‌شوند و سپس توسط کد نوشته شده توسط شما مرتب سازی می‌شوند. این تست ٪۱۰ از کل نمره این تمرین را شامل می‌شود.
۴. TestOrder: این تست که ٪۳۰ از کل نمره را به خود اختصاص داده است همانند تست قبل درستی ترتیب آرایه مورد نظر را بررسی می‌کند اما با این تفاوت که تعداد آرایه هایی که باید مرتب سازی شوند و عمل مقایسه ای که بر روی آنها صورت می‌گیرد متفاوت است. هدف از این تست، بررسی مرتبه زمان اجرای کد شماست.
۵. TestRadixSortStability: انتظار می‌رود کد پیاده‌سازی شده توسط شما، پایدار باشد. این تست، خاصیت پایداری مرتب سازی شما را چک می‌کند. ۱۰٪ از نمره نهایی شما مربوط به این تست است.
۶. TestCountingSort: تابع countingSort پیاده‌سازی شده توسط شما، در این تست بررسی می‌شود. این تست شامل ۱۵٪ نمره کل است.
۷. TestGetKeyCount: در این تست، تعداد دفعات فراخوانی مقدار key شی UnboundedInteger در تابع countingSort شمارش می‌شود. این تست ۱۵٪ نمره را به خود اختصاص می‌دهد.

* پیشنیاز قبولی در تست TestRadixSortStability ، تست TestCountingSort است.

* پیشنیاز قبولی در تست TestOrder ، قبولی در تست‌های TestSortingRandomNumbers، TestCountingSort، TestCalculateD، TestGetDigits و TestRadixSortStabilityاست.

* پیشنیاز قبولی در تست TestSortingRandomNumbers قبولی در تست‌های TestCountingSort، TestCalculateD، TestGetDigits و TestRadixSortStabilityاست.

* پیشنیاز قبولی در تست TestGetKeyCount ، قبولی در تست‌های TestSortingRandomNumbers، TestCountingSort، TestCalculateD، TestGetDigits و TestRadixSortStabilityاست.

* تمام کتابخانه‌های مورد نیاز برای حل مسأله در اختیار شما قرار گرفته است. افزودن کتابخانه جدید موجب رخ دادن compile Error در مرحله تصحیح کد می‌شود. بنابراین مجاز به استفاده از کتابخانه‌های متفرقه نیستید.

برای بارگذاری این تمرین گام‌های زیر را دنبال کنید :

۱- ابتدا فایل info.txt را با مشخصات خود پر کنید.

۲- پس از حل تمرین، از پوشه src همه فایل های اضافی که به دلیل کامپایل برنامه بوجود آمده اند را پاک نمایید. (ممکن است IDE شما به طور خودکار، فایل‌هایی را اضافه کند). در نهایت فقط فایل‌هایی که از ابتدا در پوشه src وجود داشته‌اند، همچنان باقی می‌مانند.

۳- پوشه src و فایل info.txt را در کنار این پوشه، زیپ کنید. مطمئن شوید که وقتی فایل zip را باز می کنید پوشه src و همچنین فایل info.txt را می بینید.

٤- دقت کنید که پسوند فایل شما حتما zip باشد و حجم فایل بالای یک مگابایت نباشد.

٥- فایل را در سامانه بارگذاری کنید.

٦- اشکالاتی را که سامانه مشخص کرده است برطرف نمایید و مجددا تمرین را در سامانه بارگذاری کنید.

۷- مرحله قبل را آن قدر ادامه دهید که از صحت عملکرد برنامه خود اطمینان حاصل نمایید.

با آرزوی موفقیت