به نام خدا

دستور کار کارگاه برنامه‌نویسی پیشرفته
جلسه چهارم

آشنایی با کتابخانه‌ها و مدل حافظه در جاوا

مقدمه

در این جلسه قصد داریم با تعدادی از کتابخانه‌های پرکاربرد جاوا و پیداکردن روش استفاده از آن‌ها با استفاده از مستندات جاوا آشنا شویم. پس از آن با بعضی از ساختمان داده‌های موجود در این زبان آشنا می‌شویم و نکاتی را پیرامون مدل حافظه در جاوا مطرح می‌کنیم.

نکات آموزشی

۱. آشنایی با روش استفاده از کتابخانه‌ها در جاوا
۲. آشنایی با تعدادی از کتابخانه‌های معروف و پرکاربرد در جاوا
۳. نکاتی پیرامون مدل حافظه در جاوا

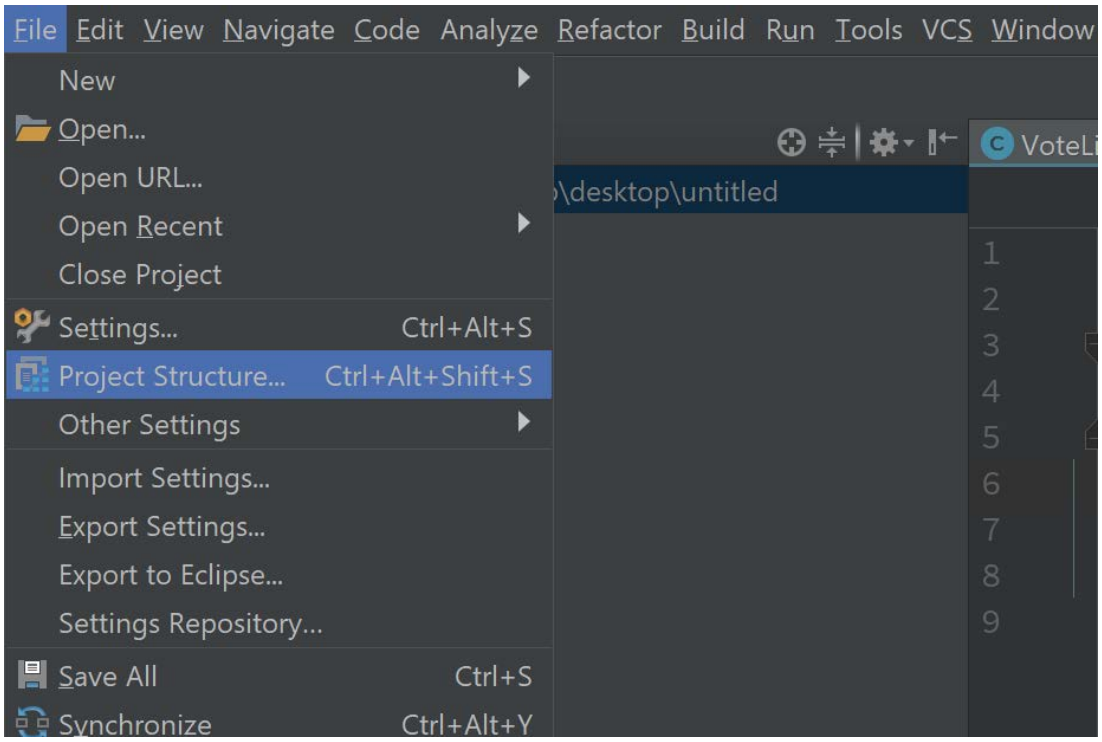
آشنایی با روش استفاده از کتابخانه‌ها در جاوا

برنامه‌نویسان جاوا برای افزایش کارایی و کاهش زمان توسعه برنامه‌های خود نیاز دارند که از کدهای آماده و نوشته‌شده توسط دیگران استفاده کنند. این کدهای آماده اغلب به صورت کتابخانه موجود است و این کتابخانه‌ها معمولا به صورت فایل با پسوند .jar در اختیار آن‌ها قرار می‌گیرد که یا به طور مستقیم دانلود شده و به پروژه اضافه می‌گردد و یا توسط ابزارهایی مانند Maven و Gradle (که به این ابزارها اصطلاحا سیستم‌های خودکارسازی ساخت[1] می‌گویند) در دسترس قرار می‌گیرند. در این دستور کار افزودن مستقیم یک کتابخانه شرح داده می‌شود و سایر ابزارها در جلسات آینده بررسی می‌شوند.
یکی از راه‌های افزودن فایل jar. توسط IDEها است. افزودن این نوع فایل توسط IntelliJ در شکل‌های ۱ تا ۳ آورده شده است. از جمله مجموعه کتابخانه‌های معروف و پرکاربرد موجود می‌توان به Google Guava و Apache Commons اشاره کرد.

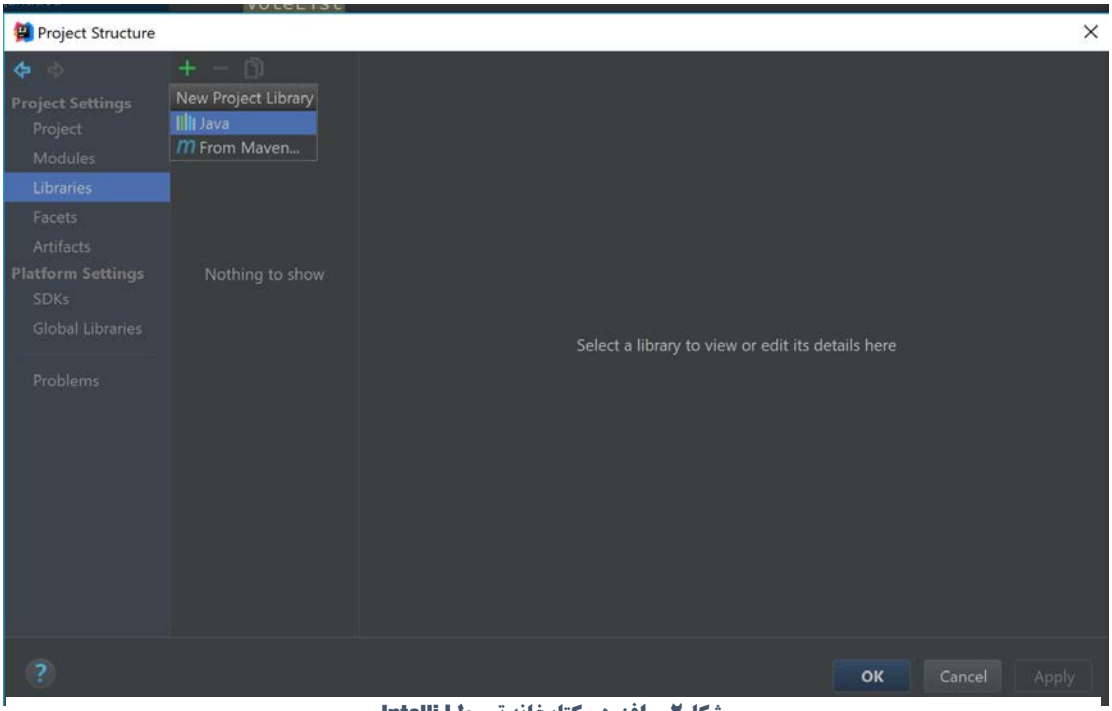
انجام دهید: کتابخانه‌های موجود در این دو مجموعه را جستجو و بررسی کنید.

بعضی از کتابخانه‌های پرکاربرد در JDK به طور پیش‌فرض وجود دارد. برای مثال کتابخانه java.util از این دسته کتابخانه‌ها است. این کتابخانه شامل کلاس‌های کاربردی مانند ساختمان داده‌های مختلف و تولید داده‌های تصادفی است. برای استفاده از این کتابخانه کافیست java.util را در ابتدای فایل خود import کنید. در این جلسه از تعدادی از این کتابخانه‌ها استفاده خواهیم کرد.

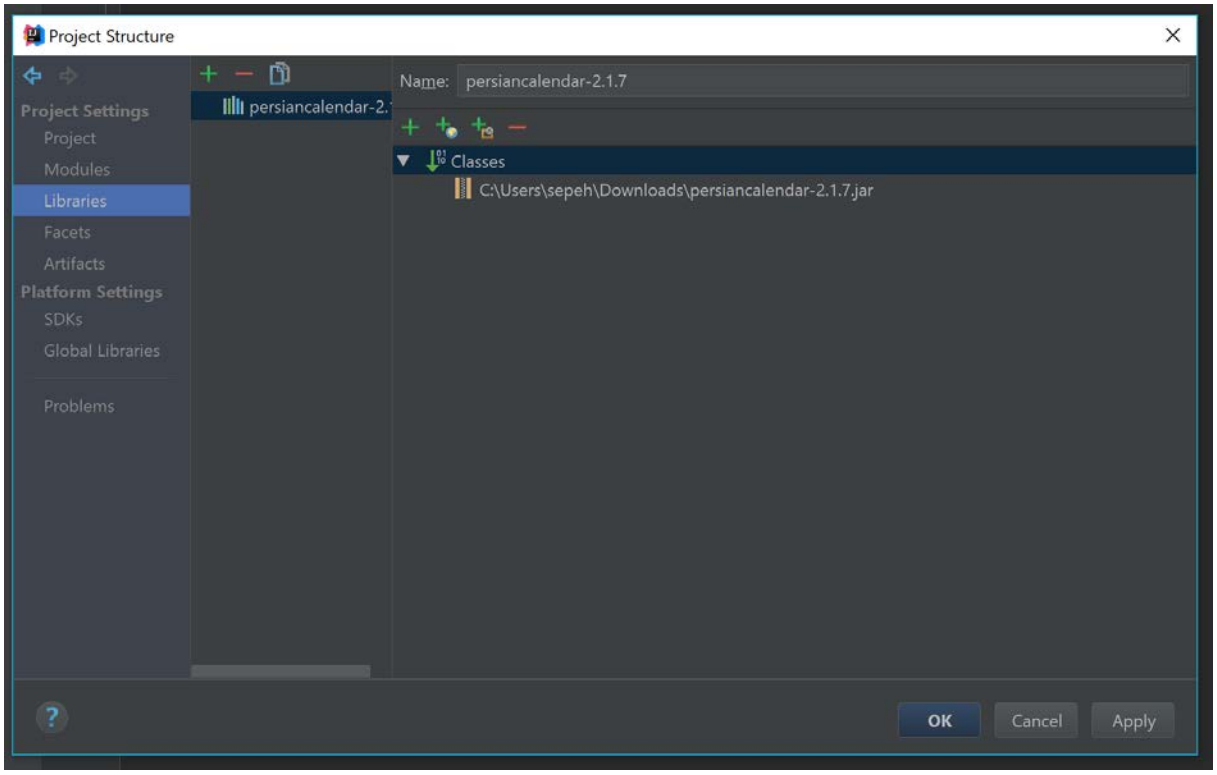
[1] Build Automation System



شکل ۱ – افزودن کتابخانه توسط IntelliJ



شکل۲ – افزودن کتابخانه توسط IntelliJ



شکل ۳ – افزودن کتابخانه توسط IntelliJ

انجام دهید: پیاده‌سازی یک نرم‌افزار رای‌گیری

از شما خواسته شده است تا یک نرم‌افزار رای‌گیری با استفاده از زبان جاوا بنویسید. در این نرم‌افزار، فرد می‌تواند یک رای‌گیری ایجاد کرده و پس از ساختن رای‌گیری، سایرین می‌توانند آرای خود را ثبت نمایند. رای‌گیری می‌تواند دارای دو مدل باشد:

۱- هر فرد تنها بتواند یک رای بدهد.
۲- هر فرد بتواند چندین رای بدهد.

در ابتدا کلاس‌ها و متدهای مربوط به این نرم‌افزار را طراحی کرده و سپس آن‌ها را پیاده‌سازی می‌کنیم.

در این نرم‌افزار به یک کلاس Voting نیاز است که در آن حالت رای‌گیری (تک رای و چند رای)، پرسش رای‌گیری، گزینه‌های رای‌گیری، مجموعه رای‌دهنده‌ها و آرای اخذشده نگه‌داری می‌شود. این کلاس باید شامل متدهایی باشد که تعداد آرای اخذشده، نتیجه تا این لحظه و افراد رای‌دهنده را بازگرداند.

علاوه بر این، یک کلاس VotingSystem نیاز است که در آن رای‌گیری‌های ساخته و ذخیره می‌شوند. این کلاس شامل لیستی از رای‌گیری‌های فعال است و باید متدهایی داشته باشد که با آن‌ها بتوان یک رای‌گیری را ایجاد و حذف کرد.

کاربر با انتخاب یکی از رای‌گیری‌ها و واردکردن اسم خود، وارد گزینه‌های مربوط به رای‌گیری شده و می‌تواند رای خود را ثبت کند. اگر رای‌گیری از حالت چند رای باشد، کاربر چند گزینه انتخاب می‌کند ولی اگر از حالت تک رای باشد، فقط یک گزینه می‌تواند انتخاب کند. در هر دو مدل رای‌گیری، کاربر پس از ثبت رای دوباره نمی‌تواند رای بدهد. پس باید بررسی کنید که اسم فرد رای‌دهنده قبلا وجود نداشته باشد.

برای ذخیره‌سازی رای‌ها لازم است از Collectionهایی مانند ArrayList، HashMap، HashSet و کتابخانه Random استفاده کنید.
در این قسمت کتابخانه‌های گفته‌شده را بیشتر توضیح می‌دهیم:

- ArrayList: این کتابخانه به شما کمک می‌کند تا بتوانید اشیا و مقادیر مختلف را در یک لیست (آرایه) ذخیره کنید. این کتابخانه امکانات زیادی برای جستجو و به‌روزرسانی مقادیر درون آن به شما می‌دهد. برای دانستن روش استفاده از این کتابخانه به این لینک مراجعه کنید. **از این کتابخانه برای ذخیره‌سازی رای‌گیری‌ها استفاده می‌کنیم.**
- HashMap: این کتابخانه به شما کمک می‌کند تا بتوانید نگاشتی از یک شی به شی دیگری را نگه دارید. برای دانستن روش استفاده از این کتابخانه به این لینک مراجعه کنید. **برای نگاشت هر گزینه به رای داده‌شده توسط رای‌دهنده از این کتابخانه استفاده می‌کنیم.**
- HashSet: این کتابخانه به شما امکان پیاده‌سازی یک مجموعه از اشیا را می‌دهد، به نحوی که امکان اضافه‌کردن شی تکراری به آن نیست. برای دانستن روش استفاده از این کتابخانه به این لینک مراجعه کنید. **برای جلوگیری از رای‌دادن چندباره کاربران از این کتابخانه استفاده می‌کنیم.**
- Random: **در رای‌گیری‌های تک رای کاربران می‌توانند با انتخاب گزینه "انتخاب تصادفی" یک گزینه را به طور تصادفی انتخاب نمایند. برای ایجاد گزینه تصادفی، از کتابخانه Random استفاده کنید.**

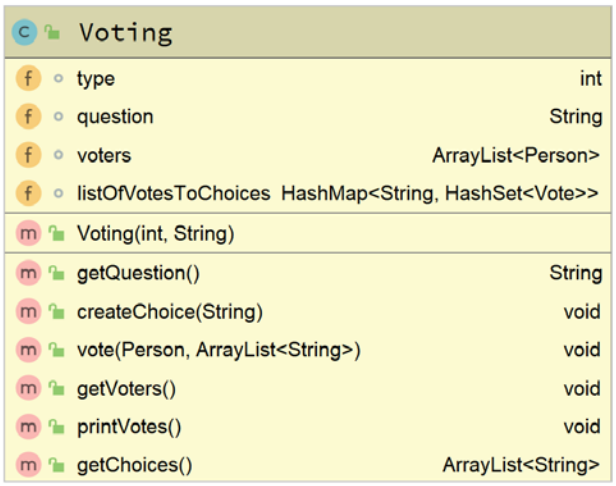
کلاس Random برای شما راهی برای تولید اعداد تصادفی ایجاد می‌کند. شما می‌بایست در ابتدا یک نمونه از این کلاس ساخته و پس از آن، با استفاده از متدهای موجود در این کلاس، که نمونه‌هایی از آن در ادامه آمده است، انواع داده‌های تصادفی را تولید کنید.

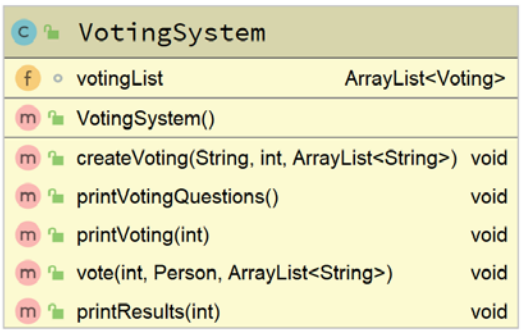
```
Random r = new Random();
r.nextBoolean(); // Generate random boolean
r.nextInt(); // Generate random integer
r.nextInt( bound: 10); // Generate random integer in [0, 10)
```

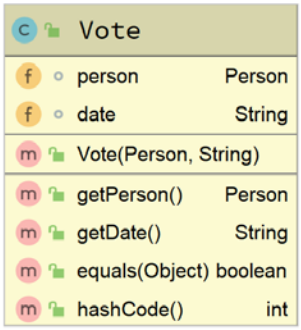
شکل ٤ – مثال‌هایی از متدهای موجود در کلاس Random برای تولید مقدار تصادفی

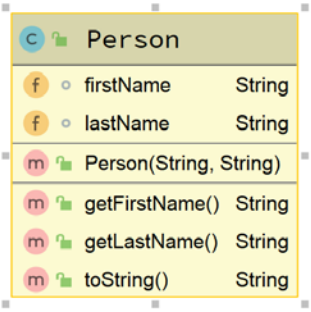
در نرم‌افزار رای‌گیری لازم است که تاریخ رای داده‌شده به هجری خورشیدی ذخیره شود. این نوع تاریخ‌نگاری به طور پیش‌فرض در جاوا وجود ندارد. به همین دلیل کتابخانه مربوط به تاریخ هجری خورشیدی (موجود در پیوست) را به پروژه اضافه کنید و از آن استفاده کنید. استفاده از این کتابخانه بسیار ساده است. برای آشنایی با این کتابخانه به این لینک مراجعه کنید.

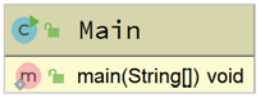
یک مهندس خوب نرم‌افزار، قبل از پیاده‌سازی برنامه، مسئله را به خوبی تحلیل و طراحی می‌کند. از همین رو، قبل از شروع برنامه‌نویسی، کلاس‌ها و اشیای مورد نیاز از آن‌ها، فیلدهای آن‌ها و متدها را طراحی می‌کنیم. شکل زیر نمونه‌ای از تحلیل و طراحی‌ای است که برای این برنامه انجام شده است. با توجه به این طراحی، برنامه را بنویسید.


Voting
type int
question String
voters ArrayList<Person>
listOfVotesToChoices HashMap<String, HashSet<Vote>>
Voting(int, String)
getQuestion() String
createChoice(String) void
vote(Person, ArrayList<String>) void
getVoters() void
printVotes() void
getChoices() ArrayList<String>


VotingSystem
votingList ArrayList<Voting>
VotingSystem()
createVoting(String, int, ArrayList<String>) void
printVotingQuestions() void
printVoting(int) void
vote(int, Person, ArrayList<String>) void
printResults(int) void


Vote
person Person
date String
Vote(Person, String)
getPerson() Person
getDate() String
equals(Object) boolean
hashCode() int


Person
firstName String
lastName String
Person(String, String)
getFirstName() String
getLastName() String
toString() String


Main
main(String[]) void

کلاس	توضیح کلاس	متد / فیلد	توضیحات متد/فیلد
VotingSystem	این کلاس وظیفه مدیریت کل برنامه را دارد. از طریق این کلاس رای‌گیری ساخته و انجام می‌شود.	فیلد votingList	لیست رای‌گیری‌های ساخته‌شده
		متد createVoting	با گرفتن سوال رای‌گیری و حالت رای‌گیری و لیست گزینه ها، یک رای‌گیری جدید می‌سازد.
		متد printVotingQuestions	سوالات رای گیری های ساخته شده را چاپ می‌کند.
		متد printVoting	شماره یک رای‌گیری را گرفته و سوال و گزینه‌های آن را چاپ می‌کند.
		متد vote	شماره یک رای‌گیری، نام رای‌دهنده و گزینه‌هایی که به آن را داده است را گرفته و در رای‌گیری ثبت می‌کند.
		متد printResults	شماره یک رای‌گیری را گرفته و نتیجه رای‌گیری را چاپ می‌کند.
Voting	این کلاس یک رای گیری را مدل می‌کند. شامل یک سوال و تعدادی گزینه است. رای‌های داده‌شده در آن ذخیره می‌شود.	فیلد type	حالت رای‌گیری است: اگر ۰ باشد، رای‌گیری از حالت تک‌رای است. اگر ۱ باشد، رای‌گیری از حالت چندرای است.
		فیلد listOfVotesToChoices	یک HashMap از گزینه به مجموعه (HashSet) رای‌های داده‌شده به آن است.
		فیلد question	سوال مربوط به رای‌گیری است.
		فیلد voters	لیست کسانی که تاکنون رای داده اند.
		متد createChoice	یک گزینه به رای‌گیری اضافه می‌کند.
		متد vote	نام رای‌دهنده و گزینه‌هایی که به آن رای داده است را گرفته و رای را ثبت می‌کند.
		متد getVoters	لیست کسانی که تاکنون رای داده اند را چاپ می‌کند.
		متد getChoices	لیست گزینه های یک رای گیری را برمی‌گرداند.
		متد printVotes	نتیجه رای‌گیری را چاپ می‌کند.

نکاتی پیرامون مدل حافظه در جاوا

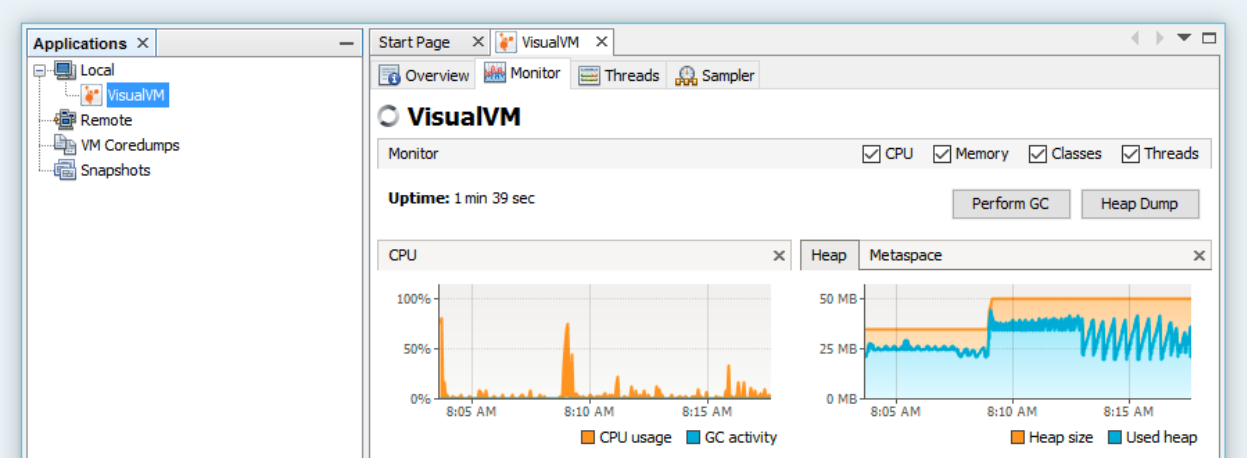
در کلاس با مدل حافظه در جاوا، عملکرد هر قسمت از آن و ارتباط بین متغیرها و نمونه‌ها آشنا شده‌اید. در این جا قصد داریم برخی نکات مرتبط با مبحث حافظه در جاوا را ذکر کنیم:

۱- اشیای تغییرناپذیر (immutable objects): متدهای موجود در یک کلاس اغلب برای تغییر وضعیت یک نمونه ساخته‌شده از آن کلاس طراحی می‌شوند. اگر این متدها، مقادیر فیلدهای یک شی را تغییر دهند، وضعیت و حالت شی تغییر کرده است و این تغییرات، برای سایر اشیایی که به آن شی دسترسی داشته باشند، قابل درک است. در برنامه‌نویسی به زبان جاوا می‌توان اشیایی ایجاد کرد که تغییرناپذیر باشند. متدهای این اشیا، تغییری در مقادیر فیلدها ایجاد نمی‌کنند. یکی از معروف‌ترین مثال‌های این‌گونه اشیا، نمونه‌های ساخته‌شده از کلاس String هستند. اشیایی که از این کلاس ساخته می‌شوند، تغییرناپذیر هستند، به این معنی که فراخوانی متدهای آن‌ها، وضعیت شی را تغییر نمی‌دهد؛ بلکه یک شی جدید می‌سازد و آن را برمی‌گرداند. به مثال زیر توجه کنید:

```
String s = "Hello world";
String sr = s.replace( oldChar: 'H', newChar: 'C');
System.out.println(s); // Hello world
System.out.println(sr); // Cello world
```

انجام دهید: مستندات متدهای concat، toUpperCase، toLowerCase، compareTo، replaceAll، split، subString و trim از کلاس String را مطالعه کنید.

۲- عملکرد Garbage Collector جاوا: همانطور که می‌دانید، هر شی‌ای که ساخته می‌شود، بخشی از حافظه Heap را به خود اختصاص می‌دهد. این حافظه می‌بایست در زمانی که استفاده‌ای از آن شی نداریم، برای استفاده در ادامه برنامه آزاد شود. این کار همان وظیفه‌ی Garbage Collector است و این کار را از طریق شمارش اشاره‌گرهایی که به آن بخش از حافظه تخصیص‌یافته برای یک شی وجود دارند، انجام می‌دهد. زمانی که هیچ اشاره‌گری به یک شی اشاره نکند، آن شی از حافظه پاک می‌شود. برای اینکه بتوانیم این عملیات را از نزدیک ببینیم، از ابزاری با نام VisualVM استفاده می‌کنیم. این ابزار به صورت پیش‌فرض به همراه JDK نصب می‌گردد و محصولی از شرکت Oracle است.



شکل ۸ - نمای نرم افزار VisualVM

انجام دهید: یک‌بار دیگر نرم‌افزار رای‌گیری را اجرا کنید و همزمان تغییرات حافظه را از طریق این ابزار مشاهده کنید.