سوال اول

محدودیت RAM برای ذخیره‌سازی آیتم‌ها در ArrayList را چگونه می‌توان حل کرد؟

اگر به عنوان مثال بخواهیم هر خط از یک فایل طولانی متنی را در داخل ArrayList ذخیره کنیم، ممکن است با خطای java.lang.OutOfMemoryError برخورد کنیم. یکی از راه‌های حل این موضوع، افزایش حافظه heap اختصاص‌یافته به JVM است. راه بهتر آن است که قسمتی از ورودی‌ها را پردازش کنیم و آن‌ها را در یک فایل یا دیتابیس ذخیره کنیم، سپس بقیه فایل را پردازش کنیم.

سوال دوم

اگر عدد خارج از محدوده ایندکس‌ها به توابع get و remove دهیم چه اتفاقی می‌افتد؟

با خطای index out of bound در زمان run time مواجه می‌شویم.

سوال سوم

اگر از مجموعه ۲۰تایی عنصر دهم را حذف کنیم، ایندکس آخرین عنصر چند می‌شود؟

در ابتدا ایندکس آخرین عنصر ۱۹ است (ایندکس از صفر شروع می‌شود). پس از آن که عنصر دهم را حذف کنیم، تمامی عناصر بعدی (از عنصر یازدهم تا عنصر آخر) یک شیفت به عقب می‌خورند؛ در نتیجه ایندکس عنصر آخر، ۱۸ می‌شود.

آیا امکان اضافه کردن شی بین اشیای دیگر در ArrayList وجود دارد؟ در این حالت ایندکس آخرین شی چه تغییری می‌کند؟

بله با استفاده از متد add(index, object) می‌توان این کار را انجام داد. با این کار تمامی عناصر بعدی یک شیفت به جلو می‌خورند و به ایندکس عنصر آخر یکی اضافه می‌شود.

اشکال زدایی

اشکالات:

- i تا size – 1 می‌تواند پیشروی کند چون ایندکس از صفر شروع می‌شود.
- به طور کلی نمی‌توان با این شیوه، یک عضو از لیست را حذف کرد چون وقتی عنصری حذف می‌شود، ایندکس تمامی عناصر بعدی یک شیفت به عقب می‌خورد و در هنگام دسترسی به عنصر آخر، با خطای out of bound index مواجه می‌شویم.
- با ارور null pointer مواجه می‌شویم چون در هیچ جای کد، لیست را new نکرده‌ایم. البته در قطعه کد درست هم new نکرده‌ام چون فرض کرده‌ام که داخل سازنده کلاس و موقع فراخوانی new می‌شود. اگر اینطور نیست باید به شکل زیر عمل کرد:

```
private ArrayList<String> = new ArrayList<>();
```

کد درست به صورت زیر است:

```java
import java.util.ArrayList;
import java.util.Iterator;

public static class MusicOrganizer {
        private ArrayList<String> tracks;

        public void removeTrack(String nameLike){
            Iterator<String> it = tracks.iterator();

            while(it.hasNext()){
                String name = it.next();
                if(name.contains(nameLike)){
                    it.remove();
                }
            }
        }
    }
```

بله می‌توان از دستور گفته‌شده استفاده کرد.

پاسخ دهید

۱.تفاوت این دو قطعه کد در چیست؟

در جاوا می‌توان یک شی را بدون آن که نام داشته باشد تعریف کنید (کد دوم). به این شی، anonymous object گفته می‌شود. ما تنها زمانی می‌توانیم شی ناشناس ایجاد کنیم که فقط یک بار بخواهیم از شی استفاده کنیم. ما در جاهای دیگر برنامه نمی‌توانیم از شیئی که به صورت ناشناس ایجاد شده است استفاده کنیم (کد دوم)؛ اما در کد اول، در صورت نیاز می‌توان در جاهای دیگر برنامه از اشیای ایجادشده استفاده کرد.

۲.می‌خواهیم سیستمی برای ذخیره و بازیابی اطلاعات دانشجویان و نمرات آن‌ها در درس‌های مختلف طراحی کنیم که اساتید هر درس به این سیستم دسترسی دارند. برای این سیستم چه کلاس‌هایی تعریف می‌کنید؟

- کلاس Student برای ذخیره‌سازی اطلاعات دانشجویان
- کلاس Course برای ذخیره‌سازی اطلاعات درس (در کلاس Student، لیستی از Course وجود دارد که در هر شی آن، نمره مربوط به آن درس نیز نگهداری می‌شود.)
- کلاس Teacher برای ذخیره‌سازی اطلاعات اساتید
- کلاس ClassRoom و Department و College و University برای نگهداری اطلاعات کلاس درس، گروه آموزشی، دانشکده و دانشگاه. در کلاس ClassRoom لیستی از Student وجود دارد و هر کدام از آن‌ها یک Teacher هم دارد. در کلاس Department لیستی از Lab و لیستی از ClassRoom وجود دارد. در کلاس College، لیستی از Department وجود دارد و در کلاس

College نیز لیستی از Department وجود دارد. در نهایت نیز در کلاس University لیستی از College وجود دارد.

در کلاس ClassRoom، Teacher می‌تواند به اطلاعات دانشجویان کلاس خود دسترسی داشته باشد. البته باید توابع setter و getter به درستی در کلاس‌های Student و Course تعریف شده باشند.

۳.سه نمونه از کلاس‌های جاوا برای دسته‌بندی اشیا به همراه کاربرد آن‌ها ذکر کنید.

کلاس ArrayList:

برای ذخیره‌سازی لیستی از اشیا که تعداد آن می‌تواند نامحدود باشد و یک ایندکس ترتیب‌دار به هر شیء اختصاص می‌یابد (ایندکس شی اول ۰، شی دوم ۱ و ...). به عنوان مثال در اپلیکیشن پخش موسیقی، می‌توان لیستی از آهنگ‌ها را با ArrayList ذخیره‌سازی کرد. در این حالت با استفاده از ایندکس می‌توان به آیتم‌ها دسترسی داشت.

کلاس HashMap:

برای ذخیره‌سازی یک حالت دیکشنری که هر عضو دیکشنری یک key و یک value دارد. این کلید و مقدار می‌توانند از هر نوع شیئی باشند. به عنوان مثال فرض کنید بخواهیم یک دیکشنری داشته باشیم که در آن به هر دانشجو یک کورس را اختصاص دهیم. در این حالت می‌توان یک HashMap<Student, Course> تعریف کرد که Student و Course به ترتیب شی دانشجو و شی کورس هستند. در این حالت با استفاده از key می‌توان به value دسترسی داشت.

کلاس HashSet:

این کالکشن در واقع همان مجموعه در ریاضیات است. اگر چند بار یک شی مشابه را به HashSet اضافه کنیم، تنها یک شی نگهداری می‌شود.

۴.یک کتابخانه جاوا برای خواندن فایل‌های اکسل (با فرمت xlsx.) پیدا کنید.

در JDK چنین کتابخانه‌ای وجود ندارد و باید از کتابخانه‌های خارجی استفاده کنیم. یک کتابخانه برای این کار، کتابخانه Apache POI است. لینک توضیحات