اشکان شکیبا (9931030)

سوال اول

الف)

Wrapper class: کلاس هایی با کاربردی مشابه داده های نوع ابتدایی (Primitive Data Types) که با ایجاد اشیای مشابه این داده ها، امکان استفاده از آن ها در مجموعه هایی همچون ArrayList و HashMap فراهم میسازد. این کلاس ها عناوینی همچون Integer، Byte، Boolean و... داشته و در پکیج java.lang قرار دارند که از این رو نیازی به import ندارند.

Autoboxing، تبدیل داده ای از انواع ابتدایی به کلاس متناظر با آن نوع داده است. برای مثال میتوان به تبدیل داده int به Integer اشاره کرد.

Unboxing، چیزی برعکس Autoboxing است، یعنی تبدیل داده ای از Wrapper class ها به Primitive Type متناظر با آن.

Garbage Collection، زباله جمع کن جاواست که وظیفه حذف اشیای غیر قابل استفاده از حافظه را دارد و با رسیدن مقدار حافظه Heap به حدی معین، این فرایند انجام میشود.

ب)

1) خیر، رفرنس دهی به اشیای غیرقابل استفاده و عدم طراحی بهینه برنامه، میتواند منجر به وضعیتی شود که حتی با وجود Garbage Collection نیز کمبود حافظه پیش بیاید.

2) بنابر سایز مورد نیاز برای کلاس سازنده Object، حافظه لازم در Heap اختصاص داده شده و رفرنسی از حافظه تخصیص داده شده در استک متد در جریان و یا در حافظه مربوط به Object قبلی که در برگیرنده Object جدید است، ذخیره میشود. سپس فیلد های Object مورد نظر با اجرا شدن کانستراکتور مقداردهی میشوند و پس از آن، Object مورد نظر با با آدرس ذخیره شده قابل دسترسی خواهد بود. همچنین با اجرای متد های آن، استک فریم مورد نیاز برای متد آماده میشود.

3) for: با سینتکس for(statement; condition; statement){statements}

while: با سینتکس while(condition){statements}

for-each: با سینتکس for(type variable: collection){statements}

Iterator: با سینتکس

```
Iterator iterator=collection.iterator();
while(iterator.hasNext()){statements on iterator.next()}
```

4) پدیده Stack Overflow زمانی رخ میدهد که داده های اختصاص داده شده به استک از ظرفیت آن بیشتر شده و اصطلاحا سر ریز (Overflow) میشود. برای مثال با اجرای main(args) در متد main، این متد آنقدر اجرا میشود و با هر بار اجرا حافظه استک به خود اختصاص میدهد که حافظه مورد نیاز از میزان در دسترس بیشتر شده و پدیده Overflow رخ میدهد.

5) کلاس HashMap، کلاسی generic است که با تعیین دو نوع کلاس در ابتدای تعریف آن، داده هایی از نوع اول را به نوع دوم مرتبط میسازد. برای مثال در پرسش پنجم همین تمرین، HashMap تعریف شده در کلاس Inventory، هر یک از اشیای کلاس Product را به اشیایی از کلاس Integer که نمایانگر تعداد محصول مورد نظر است، مرتبط میکند.

کلاس HashSet، کلاس generic دیگری ست که مجموعه ای از اشیای ساخته شده از کلاسی تعیین شده را در خود نگه میدارد.

ویژگی مهم و مشترک این دو کالکشن، استفاده آنها از HashCode اشیاست که از ورود اشیای تکراری به مجموعه اشیای ذخیره شده در HashSet و مجموعه کلید های HashMap جلوگیری میکند.

سوال دوم

1) درست

2) نادرست، تنها برای سایر اعضای پکیج قابل دسترسی هستند.

3) نادرست، کلاس ها نمیتوانند private باشند.

4) نادرست، چنین چیزی امکان پذیر نیست و Map اجازه آن را نمیدهد.

5) نادرست، داده های نوع ابتدایی (Primitive Data Types) اینطور نیستند.

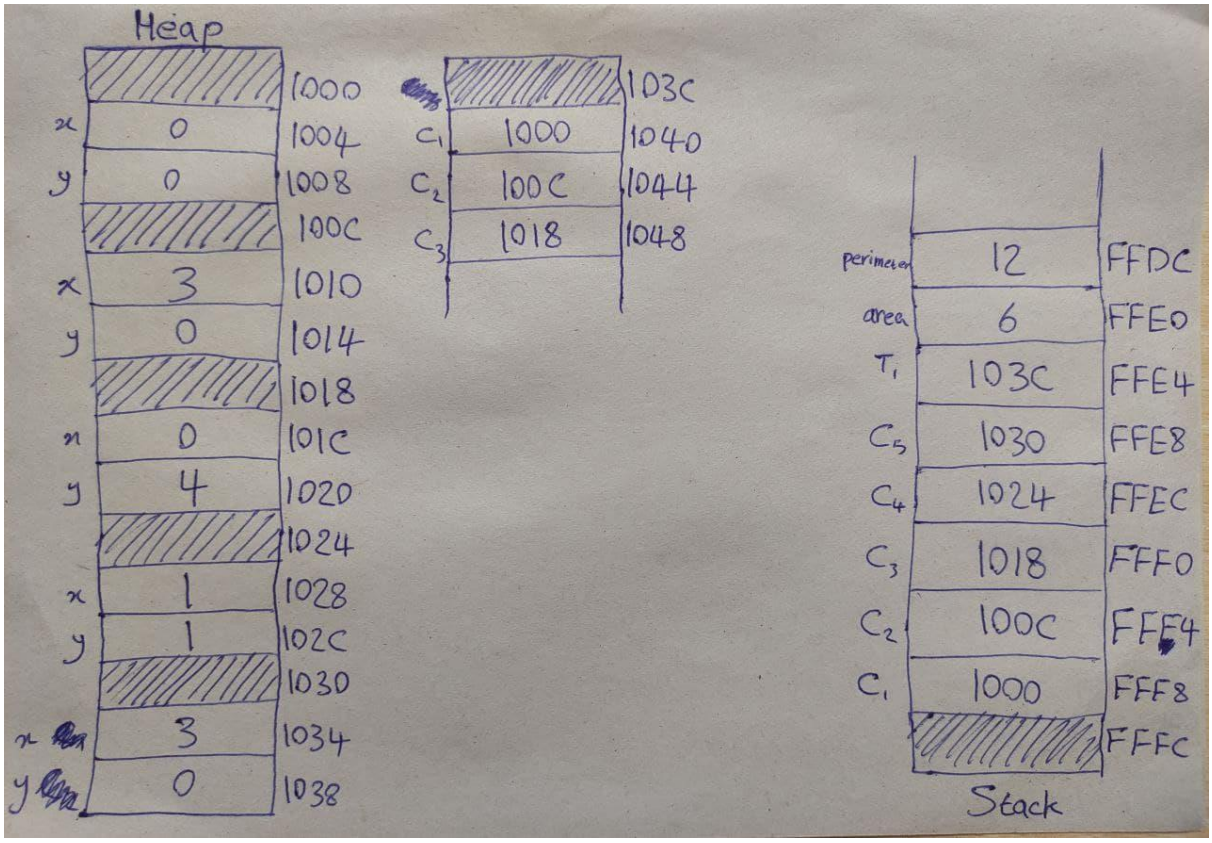
6) درست

سوال سوم

برنامه نوشته شده در پوشه Q3 قرار دارد.

سوال چهارم

الف) وضعیت حافظه بعد از اجرای خط آخر متد run:



Garbage Object نیز وجود ندارد زیرا همگی اشیای تعریف شده در Heap، دارای متغیر های رفرنس میباشند.

ب) خروجی به شکل

true

true

false

true

خواهد بود که true اول به دلیل یکسان بودن identity متغیر ها و true دوم به دلیل یکسان بودن equality آنهاست؛ بررسی یکسان بودن هویت اشیا با آدرس های ذخیره شده در رفرنس آنها صورت میگیرد (که آیا به خانه های یکسانی از حافظه اشاره میکنند) و بررسی مقدار آنها نیز با مقایسه مقادیر HashCode های آنها که بنابر مقادیر ذخیره شده در فیلد ها مقادیری یکتا برمیگرداند. البته به دلیل یکسان بودن مقدار استرینگ ها، پدیده string constant pool رخ میدهد و هر دو متغیر str1 و str2 به خانه های یکسانی از حافظه اشاره میکنند. همچنین در سومین خط از خروجی، به دلیل متفاوت بودن

مکان ذخیره سازی استرینگ ها و در نتیجه متفاوت بودن identity آنها، مقدار false بازگردانی شده و مجددا به دلیل یکسان بودن equality متغیر ها، مقدار true در آخرین چاپ میشود.

سوال پنجم

برنامه نوشته شده در پوشه Q5 قرار دارد.