امیرحسین ایروانی‌منش 9931066 airavanimanesh@aut.ac.ir

تکلیف شماره ۳

تمرین شماره ۱

الف)

Wrapper Class: کلاس هایی هستند که هر کدام معادل primitive data types یا داده های اصلی و ساده هستند و یک شئ هستند، به عنوان مثال به دلیل اینکه کالکشن ها در جاوا مانند اررِی لیست و وکتور ها و ... داده های ساده را پذیرا نمی باشند از این نوع کلاس ها استفاده میکنیم. در ادامه جدولی از معادل داده های اصلی و کلاس های wrapper آنها اورده شده:

Primitive Data Type	Wrapper Class
char	Character
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean

Autoboxing & Unboxing: اتوباکسینگ زمانی اتفاق میوفتد که یک داده ساده به کلاس wrapper معادل خود کست میشود برای مثال:

```
char ch = 'a';
Character a = ch;
```

آنباکسینگ علی رغم اتوباکسینگ، زمانی اتفاق میوفتد که یک کلاس wrapper به primitive data type معادل آن کست و تبدیل می شود. مانند کد زیر:

```
Character ch = 'a';
char a = ch;
```

Garbage Collection: در ماشین مجازی جاوا garbage collector وجود دارد که وظیفه ی مدیریت آبجکت ها را در حافظه را دارد و آبجکت های بلااستفاده رو پیدا کرده و فضای اشغال شده توسط آن (مقرر در heap) را آزاد می کند.

ب)

۱. خیر، برای مثال، برنامه نویس میتواند هزاران آبجکت را در اسکوپ اصلی ایجاد کند و در این صورت garbage collector آن شئ ها را، شئ های مورد استفاده شناسایی میکند و اقدامی به پاک کردن آنها نمی کند.

۲. وقتی یک شئ تعریف می شود یک مرجع یا رفرنس با نام متغیر مورد نظر در stack تشکیل می شود و یک فریم در هیپ (heap) تشکیل شده که متشکل از اعضای آن کلاس است و به همین ترتیب مقدار دهی می شوند ...

۳. یکی از روش های پیمایش لیست در جاوا استفاده از حلقه for معمولی است که از I = 0 تا list.size ادامه پیدا میکند و به این ترتیب پیمایش انجام می شود.

روش دیگر پیمایش لیست در جاوا، استفاده از forEach هست که برای مثال اگر یک لیست از استرینگ داشته باشیم:

```
for (String sample : samplelist)
```

روش دیگر پیمایش استفاده از کلاس Iterator هست که با تعریف یک ایتریتور و با دستور های hasnext و next لیست را پیمایش میکنیم

```
Iterator<String> iterator = list.iterator();
while (iterator.hasNext()) {
   System.out.println(iterator.next());
}
```

دو روش دیگر هم میتوان گفت که یکی listIterator است که بسیار شبیه به iterator هست و دیگر هم استفاده از یک increment ولی به جای for از while استفاده شود.

۴. پدیده stackoverflow زمانی رخ میدهد که میزان فضای مورد نیاز برای اجرای برنامه و ذخیره متغیر ها و شئ ها و ... بیش از اندازه مموری و حافظه تعیین شده (allocated memory) بشود. برای جلوگیری از آن باید کد برنامه را تغییر داد، برای مثال باید توابع ریکرسیو و بازگشتی در برنامه را به خوبی هندل کرد تا دچار stackoverflow نشود. از تغییرات دیگری که میتوان برای جلوگیری از آن اعمال کرد، افزایش سایز stack برنامه است.

۵. hashset از اینترفیس ست استفاده کرده و hashmap از اینترفیس map. hashset ورود مقادیر یکسان و مشابه را نمی پذیرد ولیhashmap از ورود key های مشابه جلوگیری میکند و شرط مشابه نبودن را در فیلد value چک نمی کند و اگرkey مشابه وارد شود، hashmap مقدار آن key را overwrite می کند. ساختمان داده hashset کند تر از ساختمان داده hashmap عمل می کند. در ادامه یک مثال از ساختار هر دو نوع آمده:

hashset: {1,2,3,4,5}

hashmap: { a->1, b->2, c->3 }

تمرین شماره ۲

الف) صحیح. پنهان کردن اطلاعات، باعث کاهش سطح دسترسی و در نتیجه باعث می شود از سطح مستقل بودن آن برنامه بکاهد. در جاوا نیز اسطلاحی به نام کپسوله سازی وجود دارد (Encapsulation) که در آن هر کپسول یک کلاس است و مقادیر و متد ها اجزای تشکیل دهنده آن هستند.

ب) غلط. فقط کلاس های داخل و زیرمجموعه ی پکیج مورد نظر به کلاس های package access آن دسترسی دارن و دسترسی به این کلاس ها از بیرون این پکیج امکان پذیر نیست.

ج) غلط. در این مورد باید به این نکته اشاره کرد که فقط کلاس های داخلی (Inner) میتوانند به صورت private تعریف شوند و کلاس های outer و خارجی نمیتوانند private باشند چون در صورت private بودن غیرقابل استفاده هستند.

د) غلط. درmap نمیتوان کلید مشابه را چندین بار داشت، و اگر سعی کنیم که یک key مشابه را وارد و duplicate کنیم، برنامه به صورت خودکار مقدار و value کلید اولیه را overwrite می کند و مقدار جدید را جایگزین میکند.

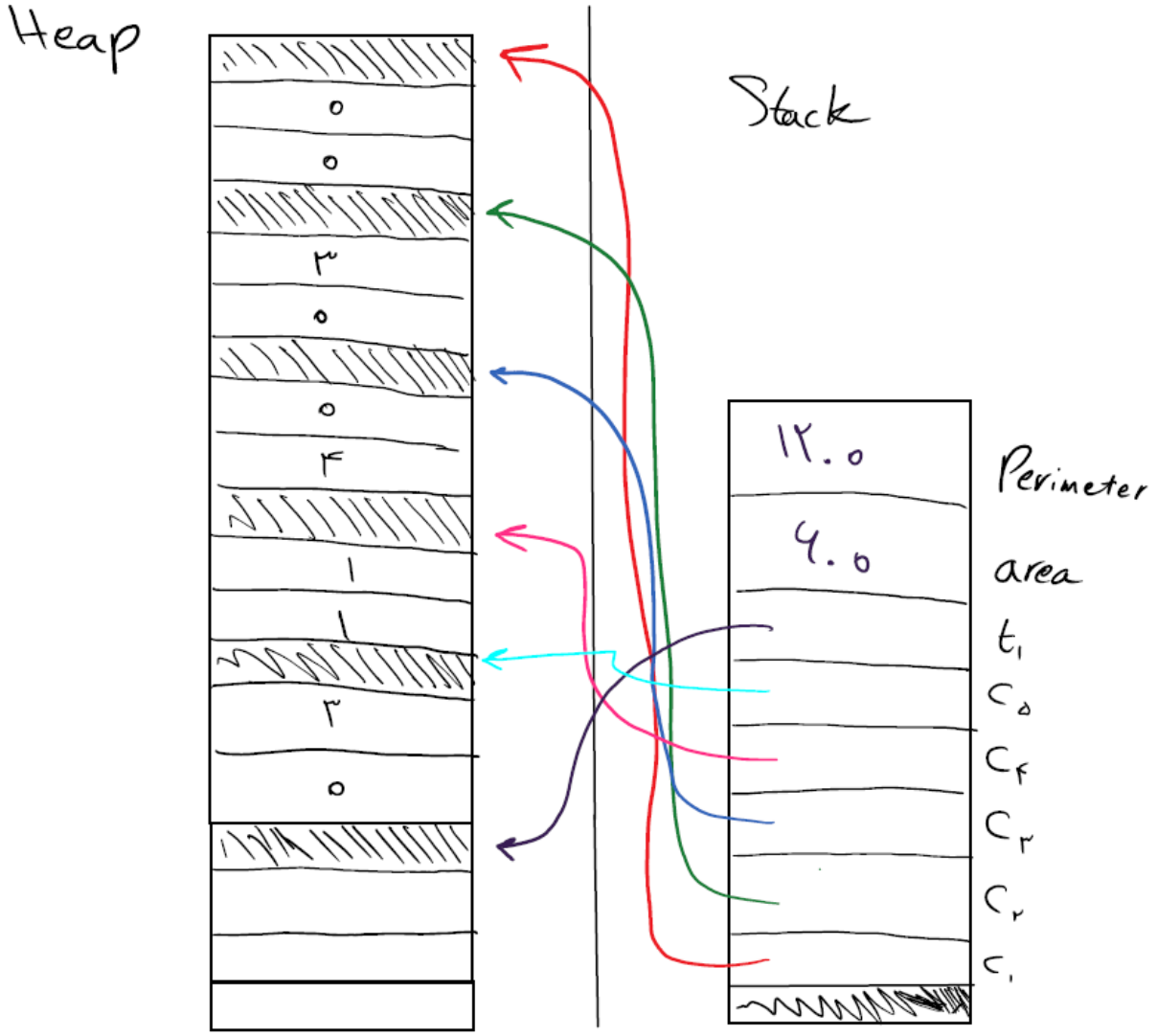
ه) غلط. جاوا همیشه pass by value است و ساز و کارش بر اساس مقدار یا value است و pass by reference نیست.

و) صحیح. بله در جاوا می توان کانستراکتور از نوع private داشت برای مثال میتوان چند کانستراکتور پابلیک داشت که با keyword (this) میتوانند به کانستراکتور پرایوت اشاره کرده و برنامه را انجام دهند.

امیرحسین ایروانی‌منش 9931066 airavanimanesh@aut.ac.ir

تمرین شماره ۴

الف)



در این برنامه garbage object وجود ندارد چون همه ی مقادیر و ابجکت ها در heap مورد استفاده و خانه هایی از استک به آنها اشاره می کنند.

ب) خروجی نهایی کنسول به این صورت است:

false

true

false

true

در خروجی هایی که False هستند به دلیل اینکه هر دو مورد به یک رفرنس اشاره نمیکنند فالس هستند ولی اگر کد به این صورت بود که

String str 1 = “hello”;

String str 2 = “hello”;

خروجی true می شد چون هر دو متغیر به یک رفرنس اشاره میکنند.

در خروجی های true نیز چون محتویات دو مورد مقایسه شده و یکسان هستند، نتیجه true چاپ می شود.