به نام خدا

پاسخنامه تمرین سوم درس برنامه نویسی پیشرفته

نیمسال دوم ۱۳۹۹-۱۴۰۰

فهرست سوالات

سوال اول

الف)

Wrapper class: این کلاس‌ها راهی برای استفاده از primitive type ها به عنوان object هستند. زبان جاوا برای هر primitive type یک wrapper class دارد که با آن مطابقت دارد.

به عنوان مثال:

boolean <-> Boolean

int <-> Int

float <-> Float

Autoboxing and Unboxing: به تبدیل کردن یک نوع داده اولیه (primitive type) به یک object از wrapper class متناظر با آن، Autoboxing در جاوا گفته می‌شود.

به عنوان مثال:

Integer five=5

که جاوا به صورت اتوماتیک Integer Constructor را صدا می‌زند.

به تبدیل کردن یک آبجکت از نوع کلاس Wrapper به نوع داده اولیه‌ای (primitive type) که متناظر آن است، Unboxing گفته می‌شود.

به عنوان مثال:

int six=five + 1

که جاوا به صورت اتوماتیک intValue را برای five صدا می‌زند.

Garbage Collection: در JVM جاوا یک Garbage Collector قرار دارد که وظیفه اش این است که فضاهایی در heap که هیچ پوینتری از stack به آنها اشاره نمی‌کند(unused object) را reclaim می‌کند تا دوباره از آنها در آینده استفاده کند. به این فرایند Garbage Collection گفته می‌شود.

ب)

۱) خیر زیرا ممکن است در یک برنامه تعداد زیادی آبجکت بسازیم که با نگه داشتن reference به آنها، هیچگاه از scope خارج نشده و در نتیجه فرآیند Garbage Collection روی آنها انجام نشود و حافظه پر شود.

۲)

1. Memory is allocated from heap to hold all instance variables and implementation-specific data of the object and its super classes. Implementation-specific data includes pointers to class and method data.

2. The instance variables of the objects are initialized to their default values.

3. The constructor for the most derived class is invoked. The first thing a constructor does is to call the constructor for its super classes. This process continues until the constructor for java.lang.Object is called, as java.lang.Object is the base class for all objects in java.

4. Before the body of the constructor is executed, all instance variable initializers and initialization blocks are executed. Then the body of the constructor is executed. Thus, the constructor for the base class completes first and constructor for the most derived class completes last.

۳)

● Basic for loop:

```
for (int i = 0; i < myList.size(); i++) {
System.out.println(myList.get(i));
}
```

● Enhanced for loop: The enhanced for loop is a simple structure that allows us to visit every element of a list.

```
for (String entry : myList) {
System.out.println(entry);
}
```

● Iterator: An Iterator is a design pattern that offers us a standard interface to traverse a data structure without having to worry about the internal representation.

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

● ListIterator:

A ListIterator allows us to traverse a list of elements in either forward or backward order.

```
ListIterator<String>
listIterator= myList.listIterator();
while(listIterator.hasNext()) {
System.out.println(listIterator.next());
}
```

اگر به جای ListIterator، هرکدام از (forEach.Iterable) یا (forEach.Stream) یا while توضیح داده شده باشند، قابل قبول است.

۴)

A stack overflow is when you've used up more memory for the stack than your program was supposed to use.

The most common causes are:

● unterminated/infinite recursion.

● calling methods from within methods until the stack is exhausted.

● having a vast number of local variables inside a method.

● having cyclic relationships between classes.

● if a class is being instantiated within the same class as an instance variable of that class.

To prevent this:

● Reduce the local variable storage.

● Avoid or strictly limit recursion (detect cycles and recursion that are not intended).

● Don't break your programs up too far into smaller and smaller functions - even without counting local variables each function call consumes as much as 64 bytes on the stack.

(۵

Sr. No.	Key	HashMap	HashSet
1	Implementation	Hashmap is the implementation of Map interface.	Hashset on other hand is the implementation of set interface.
2	Internal implementation	Hashmap internally do not implements hashset or any set for its implementation.	Hashset internally uses Hashmap for its implementation.
3	Storage of elements	HashMap Stores elements in form of key-value pair i.e., each element has its corresponding key which is required for its retrieval during iteration.	HashSet stores only objects no such key value pairs maintained.
4	Index performance	Put method of hash map is used to add element in hashmap.	On other hand add method of hashset is used to add element in hashset.
5	Null Allowed	Hashmap due to its unique key is faster in retrieval of element during its iteration.	HashSet is completely based on object so compared to hashmap is slower.
6	duplicate	Single null key and any number of null values can be inserted in hashmap without any restriction.	On other hand Hashset allows only one null value in its collection, after which no null value is allowed to be added.

Similarities:

Both of these classes do not guarantee that the order of their elements will remain constant over time.

They both provide constant time performance for basic operations such as adding, removing elements etc.

سوال دوم

1. **true**

Information hiding increases the level of independence because the data belonging to one object is hidden from other objects.

2. **false**

Package access members can only be seen and used by the **package** in which it was declared. package-private is the default access modifier and does not have a keyword, because package is used to specify the package for a class or interface.

3. **false**

We **cannot** declare top level class as private. Java allows only public and default modifiers for top level classes in java. Inner classes can be private.

4. **false**

Duplicate keys are **not allowed** in a map but duplicate values are allowed.

5. **false**

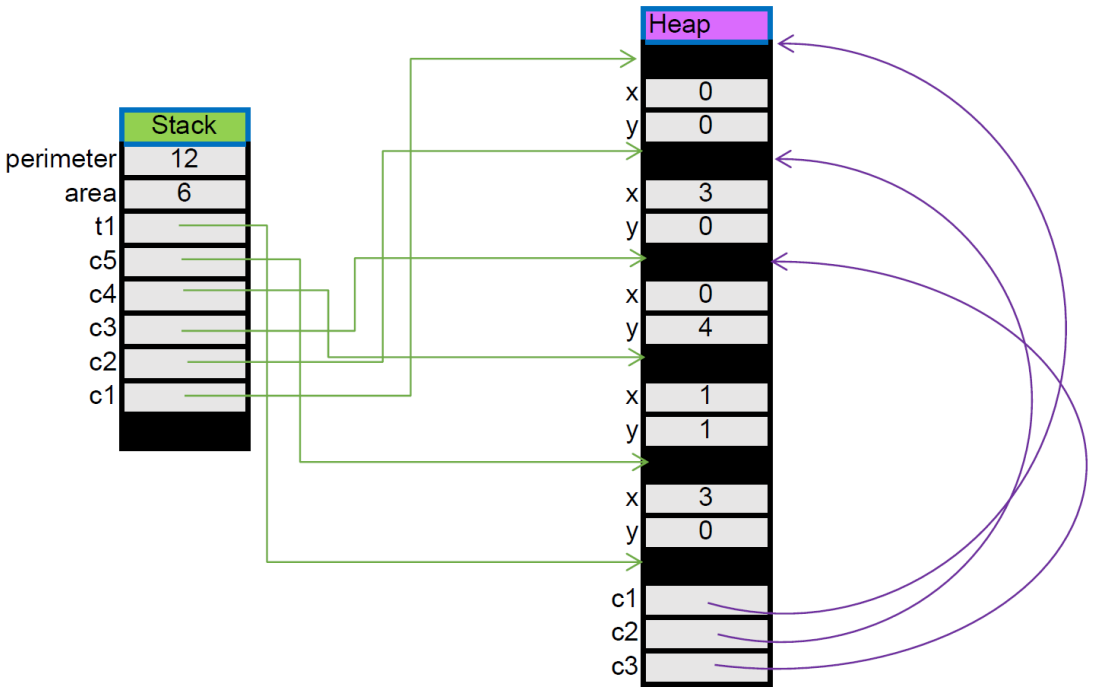
Java always passes parameter variables by value. Object variables in Java always point to the real object in the memory heap. A mutable object's value can be changed when it is passed to a method. “Passing by reference” refers to passing the real reference of the variable in memory.

6. **true**

We can declare a constructor as private. If we declare a constructor as private, we are not able to create an object of a class. We can use this private constructor in the Singleton Design Pattern.

سوال چهارم

الف)



وجود ندارد.

زیرا به ازای تمامی object ها، پوینتری به آدرس آنها وجود دارد و اصطلاحاً آدرس آنها گم نمی‌شود.

ب)

True, True, False, True

توضیح:

در تساوی‌های دوم و چهارم برابری محتوای دو رشته به وسیله equals() چک می‌شود پس جواب درست است.

اما در تساوی‌های اول و سوم برابری آدرس دو رشته چک می‌شود.

در تساوی اول چون رشته از قبل در String Pool موجود است، رشته جدیدی درست نمی‌شود پس آدرس برابری در Heap دارند.

در تساوی سوم چون در Integer.toString(123) از کلمه کلیدی new برای ساخت رشته استفاده می‌شود، آدرس جدیدی به آن اختصاص داده می‌شود پس آدرس‌ها یکسان نیستند.

برای فهم بهتر به لینک زیر مراجعه کنید.

https://www.javatpoint.com/string-pool-in-java