«باسمه تعالی»

سوال اول:

قسمت 1: کلاس چیست؟ چه تفاوتی با شیء دارد؟

کلاس (class) در برنامه نویسی شئ گرا، نوعی قالب برنامه نویسی قابل گسترش است که برای ایجاد اشیاء، تعیین مقادیر اولیه ی state و مشخص نمودن رفتار آنها مورد استفاده قرار میگیرد. در واقع، کلاس نقشه ی نوعی و مشترک برای گروهی از اشیاء است که ویژگی های مشترکی داشته، و رفتارهای مشترکی از خود نشان میدهند. یعنی کلاسها انواعی هستند که شخص برنامه نویس، خود میتواند، آنها را برای حل مسئله های دنیای واقعی طراحی کند. یک کلاس، یک مفهوم بسط یافته از ساختمان (structure) است که به جای این که، فقط داده ها را نگهداری کند، میتواند هم داده ها و هم توابع را با هم نگهداری کند.کلاس درواقع یک توصیف کلی برای همه ی object هایی که از یک نوع هستند، میباشد. اگر کلاس ها رو معادل (اسم) در نظر بگیریم آنگاه اشیاء معادل (اسم خاص) هستند.

شیء هایی که از روی کلاس ساخته میشوند را یک نمونه(instance) از آن کلاس مینامند. به عنوان مثالی ساده میتوان کلاسی تحت عنوان Circle را در نظر گرفت که خصوصیات و رفتار اشیائی به نام دایره را تعریف میکند. با کمک این کلاس، برنامه نویس میتواند اشیاء (دایره) های مختلفی با خصوصیات مرکز و شعاع متفاوت ایجاد کند و اعمالی نظیر محاسبه شعاع و مساحت یا ترسیم دایره را روی هرکدام انجام دهد در حالی که تعریف و پیاده سازی تمامی این خصوصیات و اعمال یک بار حین تعریف کلاس صورت گرفته است. حال میتوان از روی این کلاس circle که مثل یک نقشه است نمونه های مختلفی ساخت که به هرکدام از آن ها یک شی گفته میشود. ممکن است تمام ویژگی ها ی این اشیاء حتی یکی باشند ولی باز هم منحصر به فردندو مثل دوقلوها.

قسمت2: سازنده کلاس یک تابع است ، این تابع به چه شکل از باقی توابع کلاس تمیز داده میشود؟

این تابع برخلاف دیگر توابع کلاس حتما باید هم نام کلاس خودش باشد. همچنین این تابع در یک کلاس حتماوجود دارد به این صورت که یا خودمان مینویسیم و تعریف میکنیم و در آن فیلدهای کلاس را مقدار دهی اولیه میکنیم یا اینکه اصلا تعریف نمیکنیم و جاوا به صورت پیش فرض یک سازنده را در نظر میگرد و براساس تایپ فیلد ها ،آن ها را به صورت پیش فرض مقدار دهی میکند.

یکی از اصلی ترین نکات این است که متد constructor هیچگونه مقدار برگشتی مشخصی ندارد. یعنی هنگام ساختن آن هیچ مقدار برگشتی را برای آن تعریف نمیکنیم برای متد سازنده حتی void را هم مشخص نمیکنیم.

به صورت خلاصه در زیر داریم:

متد در جاوا	تابع سازنده در Java
یک متد صرفا نشانگر قابلیت و کاری است که یک آبجکت قادر به انجام آن می باشد.	تابع سازنده برای مقداردهی اولیه و پرکردن یک آبجکت با داده بکار می رود.
یک متد خروجی داشته و این باید صراحتا مشخص شود.	تابع سازنده ی خروجی ندارد و نوع خروجی نیز برای آن تعیین نمی شود.
متد از کلاس به صورت صریح فراخوانی می‌شود.	تابع سازنده به صورت ضمنی فراخوانی می شود.
متد را کامپایلر تولید نمی کند.	اگر برنامه نویس متد سازنده تعریف نکند کامپایل جاوا خود یک تابع سازنده ی پیش فرض فراخوانی می کند.
اسم متد ممکن است با اسم کلاس یکسان باشد و ممکن است متفاوت باشد.	اسم تابع سازنده باید با اسم کلاس یکسان باشد.

قسمت3:

خیر- کد نشان داده شده دارای ایراداتی میباشد. اولا که اسم متد constructor باید دقیقا برابر با اسم کلاس باشد پس book غلط بوده و باید Book شود.

ثانیا متغیر title به خودش assign میشود در این کد. زیرا مرجع title نزدیک ترین اسکوپ میشود. برای اینکه بگوییم title موجود در پارامتر ورودی constructor را در فیلد tiltle قرار بده، باید به این شکل کار کنیم:

this.title=title

کلیدواژه ی this می تواند برای اشاره به متغیری که خارج از بدنه ی متد و داخل کلاس تعریف شده از نمونه ی جاری کلاس بکار رود. کد تصحیح شده به شکل زیر است:

```
public class Book {
    private String title;
    public Book(String title)
    {
        this.title = title;
    }
}
```

سوال دوم:

درستی یا نادرستی عبارات زیر را مشخص کنید و عبارات غلط را نیز اصلاح کنید.

آ) در صورتی که ما یک سازنده با چند آرگومان ورودی تعریف کنیم، زبان جاوا نیز یک سازنده پیشفرض ، بدون آرگومان برای کلاس، درنظر می گیرد .

نادرست-اگر Constructor در یک کلاس وجود نداشته باشد، کامپایلر جاوا به صورت خودکار یک Constructor پیشفرض با دیتا تایپ های اولیه (Primitive)ایجاد میکند و به صورت پیشفرض آن ها را مقدار دهی میکند.

ب) سطح دسترسی سازنده کلاس، حتما باید از نوع public باشد.

نادرست- سطح دسترسی سازنده کلاس، می تواند از نوع public یا private باشد اما نکته مهم این است که سطح دسترسی مشخص باشد.

پ) اعضای protected در یک کلاس ، تنها برای اعضای همان کلاس قابل دسترسی هستند .

نادرست- اعضای private در یک کلاس تنها برای اعضای همان کلاس قابل دسترسی هستند. اعضای protected برای subclass ها و instanceهای یک کلاس هم قابل دسترسی هستند.

ت)متغیرهایی که در درون توابع تعریف می شوند، به عنوان field کلاس شناخته شده و دیگر توابع می توانند به آن ها دسترسی داشته باشند.

نادرست-متغیرهایی که در کلاس تعریف میشوند به عنوان field کلاس شناخته شده و دیگر توابع می توانند به آن ها دسترسی داشته باشند. متغیرهایی که در توابع تعریف میشوند local variable هستند و دیگر توابع به آن ها دسترسی ندارند.

سوال سوم:

قسمت 1: Primitive types در جاوا چه مواردی شامل می شوند ؟ این تایپ ها شامل چه مقادیری هستند و دامنه تغییرات آن ها چیست ؟

داده های پایه(Primitive Type ها) خود به چهار گروه تقسیم می شوند :

1. اعداد صحیح, که شامل byte, short, int, long می شود که اعداد کامل علامت دار می باشد.
2. اعداد اعشاری با ممیز شناور, که شامل float و double است.
3. کاراکترها, که شامل char است.
4. بولی, این گروه شامل Boolean است که شامل مقادیر بولی مانند True/False است.

حال به توضیح هرکدام میپردازیم:

Byte:

- نوع داده ی byte یک نوع داده ی 8 بیتی می باشد که دوعدد صحیح مکمل اختصاص داده شده است.
- حداقل مقدار آن 128- می باشد(7^2-).
- حداکثر مقدار آن 127 می باشد(1- 7^2).
- مقدار پیش فرض 0 می باشد.
- نوع داده ی byte برای ذخیره ی فضا در ردیف های بزرگ استفاده می شود، عمدتا در محل اعداد صحیح، زیرا یک byte چهار برابر کوچکتر از یک int می باشد.

Short:

- نوع داده ی short یک داده ی 16 بیتی است که 2 مقدار مکمل صحیح می باشد.
- حداقل مقدار آن (15^2-) 32,768- می باشد.
- حداکثر مقدار آن 32,767 (1- 15^2) است.
- نوع داده ی short می تواند برای ذخیره ی حافظه برای ذخیره ی نوع داده ی byte نیز استفاده شود. یک داده ی short دو برابر کوچکتر از int می باشد.
- مقدار پیش فرض آن 0 است.

Int:

- نوع داده ی int یک داده ی 32 بیتی با مقدار مککل صحیح 2 می باشد.
- حداقل مقدار آن (2^31-).2,147,483,648 - می باشد.
- حداکثر مقدار آن 2,147,483,647 (1- 2^31) می باشد.
- به طور کل int به عنوان نوع داده ی پیش فرض برای مقادیر انتگرال استفاده می شود، مگر اینکه در مورد حافظه نگرانی وجود داشته باشد.
- مقدار پیش فرض 0 می باشد.

Long:

- نوع داده ی long یک داده ی 64 بیتی است با مقدار مکمل صحیح 2.
- حداقل مقدار (2^63-).9,223,372,036,854,775,808- می باشد.
- حداکثر مقدار 9,223,372,036,854,775,807 (1- 2^63) می باشد.
- این نوع هنگامی مورد استفاده قرار می گیرد که یک گستره ای وسیع تر از int مورد نیاز است.
- مقدار پیش فرض 0L می باشد.

Float:

- نوع داده ی Float یک داده ی 32 بیتی IEEE 754 می باشد.
- Float اساسا برای ذخیره ی حافظه در ردیف های بزرگ از تعداد ممیزهای شناور، استفاده می شود.
- مقدار پیش فرض 0.0f می باشد.
- نوع داده ی float هرگز برای مقادیر دقیق مانند ارز استفاده نمی شود.

Double:

- نوع داده ی double دقت مضاعف 64-bit IEEE 754 floating point می باشد.
- این نوع داده به طور کل به عنوان نوع داده ی پیش فرض برای مقادیر اعشاری استفاده می شود، و به طور کل یک انتخاب پیش فرض می باشد.
- نوع داده ی double هرگز نباید برای مقادیر دقیق مانند ارز استفاده شود.
- مقدار پیش فرض 0.0d می باشد.

Boolean:

- نوع داده ی Boolean یک بیت اطلاعات را نمایش می دهد.
- تنها دو مقدار ممکن وجود دارد: true، false
- این نوع داده برای پرچم های ساده ای استفاده می شود که شرایط true/false را دنبال می کند.
- مقدار پیش فرض false می باشد.

Char:

- نوع داده ی char یک کاراکتر مجزای 16-bit Unicode می باشد.
- حداقل مقدار (0 or) '\u0000' می باشد.
- حداکثر مقدار (65,535 inclusive or) '\uffff' می باشد.
- نوع داده ی char برای ذخیره ی هر نوع کاراکتر استفاده می شود.

موارد گفته شده به صورت خلاصه در شکل روبه رو آمده:

Primitive Types

Type	Bit Depth	Value Range
boolean and char		
boolean	(JVM-specific)	***true*** or ***false***
char	16 bits	0 to 65535
numeric (all are signed)		
integer		
byte	8 bits	-128 to 127
short	16 bits	-32768 to 32767
int	32 bits	-2147483648 to 2147483647
long	64 bits	-huge to huge
floating point		
float	32 bits	varies
double	64 bits	varies

قسمت 2: مفموم overloading در توابع به چه معناست ؟

overloading تابع، قابلیتی است که به یک کلاس اجازه میدهد، تعداد دو یا بیشتر تابع با نام یکسان داشته باشد.

در overload یک تابع آرگومان های ورودی، دارای تفاوت های به شرح زیر هستند:

1. تعداد پارامترها.
2. نوع داده پارامترها.
3. ترتیب نوع داده پارامترها.

overloadingتابع، با عنوان پلی مورفیسم ایستا (static) نیز شناخته میشود.

باید حواسمان باشد که حتما در یکی از موارد گفته شده توابع با هم تفاوت داشته باشند در غیر این صورت کامپایلر خطا میدهد.

قسمت3 Getter و Setter در کلاس ها چه هستند و چه کاربردی دارند؟

به متد های getter اصطلاحاAccessor و به متدهای setter اصطلاحاMutator گفته میشود.

getter ها وsetter ها، توابع یا متدهایی هستند که برای دریافت کردن و تنظیم کردن مقادیر متغیرها استفاده می‌شوند.متدهای getter و setter متدهایی هستند که معمولا به صورت عمومی یا publicتعریف می‌شوند و ما به وسیله‌ی آن‌ها می‌توانیم به فیلدهایی که به صورت private تعریف شده‌اند دسترسی پیدا کنیم.

متدهای getter باید از جنس فیلد مد نظر باشند. به عنوان مثال فیلد name از جنس کلاس String است، بنابراین متد getName هم لازم است که از جنس کلاس String باشد و باید یک مقدار String برگرداند که در اینجا مقدار nameرا با استفاده از کلیدواژه‌ی returnبر می‌گرداند. اما متد setter باید از جنس voidباشد و مقداری را بر نگرداند و یک پارامتر دریافت کند و مقدار آن پارامتر را به فیلد مورد نظر انتساب دهد .یعنی با استفاده از متد setter می‌توانیم مقادیر فیلدها را تغییر دهیم.