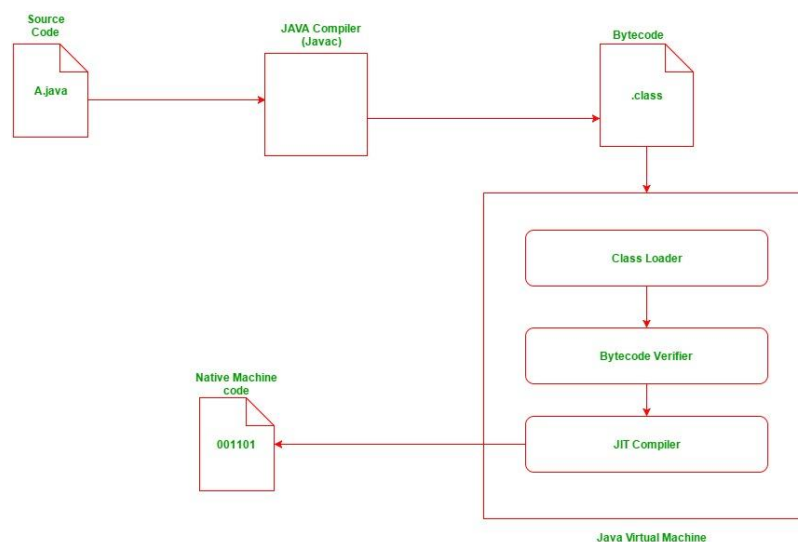


تکلیف شماره ۱

تمرین شماره ۱

الف) اولین تفاوت بین این ۳ اصطلاح، کلمات توصیفی آنها به صورت کامل (مخفف نشده) است. JRE یا Java Runtime Environment و JDK یا Java Development Kit و JVM یا Java Virtual Machine. همانطور که از نام کاملشان پیدا است، بین این ۳، تفاوت های بسیاری وجود دارد، JDK کیت و ابزاری برای توسعه نرم افزار بر پایه جاوا است در حالی که JRE مجموعه ای از نرم افزار ها است که اجازه میدهند یک برنامه جاوا اجرا شود. JVM هم یک محیطیست ساختگی برای اجرای برنامه های جاوا. JRE وابسته به یک پلتفرم خاص هستند (یعنی نسخه های متفاوتی برای پلتفرم های متفاوت دارند) در حالی که این مسئله برای JVM صادق نیست. JDK شامل ابزار هایی برای توسعه، دیباگ و کارهایی از این قبیل است و JRE شامل کتابخانه و کلاس های مختلف است در حالی که JVM هیچ ابزاری برای توسعه نرم افزار و برنامه بر پایه جاوا ندارد.

ب) در ابتدا، فایل java. برنامه به کامپایلر پاس داده میشود، کامپایلر فایل برنامه را به زبان مستقل از ماشین (بایت کد) تبدیل میکند و محتویات هر کلاس در فایل سورس به صورت یک فایل class. تولید میشوند. بعد از آن فایل class اصلی (که در آن متد main وجود دارد) به JVM پاس داده میشود و سپس فایل های class. ای که در فایل اصلی صدا زده شده اند، توسط class loader لود می شوند (منظور از لود شدن تبدیل شدن به بایت کد است). بعد از این مرحله، بایت کد ها به Bytecode Verifier چک میشوند که کار آن چک کردن کد برای جلوگیری از ایجاد آسیب های احتمالی است. چند نمونه از فاکتور هایی که توسط Bytecode Verifier چک میشوند: ۱. درست صدا زده شدن متد ها بر اساس public یا private بودن آنها ۲. اورفلو نکردن استک در حین اجرای برنامه و مرحله بعدی و آخرین مرحله پاس داده شدن Bytecode به JIT Compiler است که مخفف Just-In-Time است که بایت کد های وریفای شده و بدون مشکل را به زبان طبیعی ماشین تبدیل میکند و سپس برنامه اجرا میشود.



ج) داده‌ها در زبان جاوا به دو نوع داده‌های ساده (Primitive data) و داده‌های غیر ساده (Non-Primitive data) تقسیم می‌شوند، داده‌های ساده فقط مقادیری واحدی دارند و قابلیت خاصی ندارند. در کل داده‌های ساده را میتوان داده‌هایی تعریف کرد که نمیتوان آن‌ها را به داده‌های ریز تری تقسیم و تبدیل کرد. مثال‌هایی از این نوع داده در جاوا : `byte, short, int, long, float, double, char, Boolean`.

د) در برنامه‌نویسی ساخت یافته، از تابع‌ها و ترکیب آنها برای نوشتن برنامه استفاده می‌شود. برای مثال یک برنامه به چند تابع تقسیم میشود که هر تابع مسئول انجام قسمتی از پردازش اطلاعات است و در قسمت‌های مختلف برنامه استفاده می‌شوند. اما در برنامه‌نویسی شیء گرا برنامه به چند بخش مخلف تقسیم شده که متشکل از کلاس‌ها و متدهاست. در این روش اجزای اصلی اجرای برنامه نمونه‌هایی از کلاس‌های تعریف شده می‌باشند که در ادامه آن بخش‌های فرعی اعم از پردازش اطلاعات با استفاده از متدهای هر شیء ساخته شده انجام می‌شود. برای مثال برای انجام یک پردازش روی یک استرینگ میتوان از متد تعریف شده در کلاس اصلی استرینگ استفاده کرد و نیازی به نوشتن تابع اضافی یا صدا زدن تابع خارجی برای آن نیست. و از برنامه‌نویسی تابعی برای محاسبه عملیات‌های ریاضی و دوری از داده‌های قابل تغییر استفاده می‌شود.

ه) کلاس: به طور کلی، به مجموعه کدی گویند که برای یک هدف نوشته شده اند و در کنار یکدیگر قرار گرفته اند. آن را میتوان به عنوان الگو یا **Pattern** خاصی گرفت. برای مثال میتوان ماشین را یک کلاس در نظر گرفت.

آبجکت: یک شیء یک نمونه از کلاس است که بر اساس اون کلاس ساخته شده و ویژگی‌های آن کلاس را داراست.

کانستراکتور: سازنده (**constructor**) شبیه به متد است (اما در واقع متد نیست) که هنگام ایجاد شیء فوراً به صورت خودکار فراخوانی می‌شود.

متد: یکی از فیچرهایی که در زبان‌های برنامه‌نویسی به منظور انجام تسک‌های تکراری و کدنویسی اصولی‌تر گنجانده شده است مفهومی تحت عنوان **Method** می‌باشد.

پارامتر: پارامتر داده‌ای است که می‌توان آن را به متد پاس داد و پردازش‌های مورد نظر روی آن انجام شود.

نمونه:

و) متد مانند تابع است که کارهای مشخصی را انجام میدهد، تنها تفاوت آن با تابع این است که متعلق و مرتبط به یک شیء است.