# به نام خدا

## تمرین اول درس برنامه نویسی پیشرفته

نام و نام خانوادگی: امیرحسام ادیبی نیا

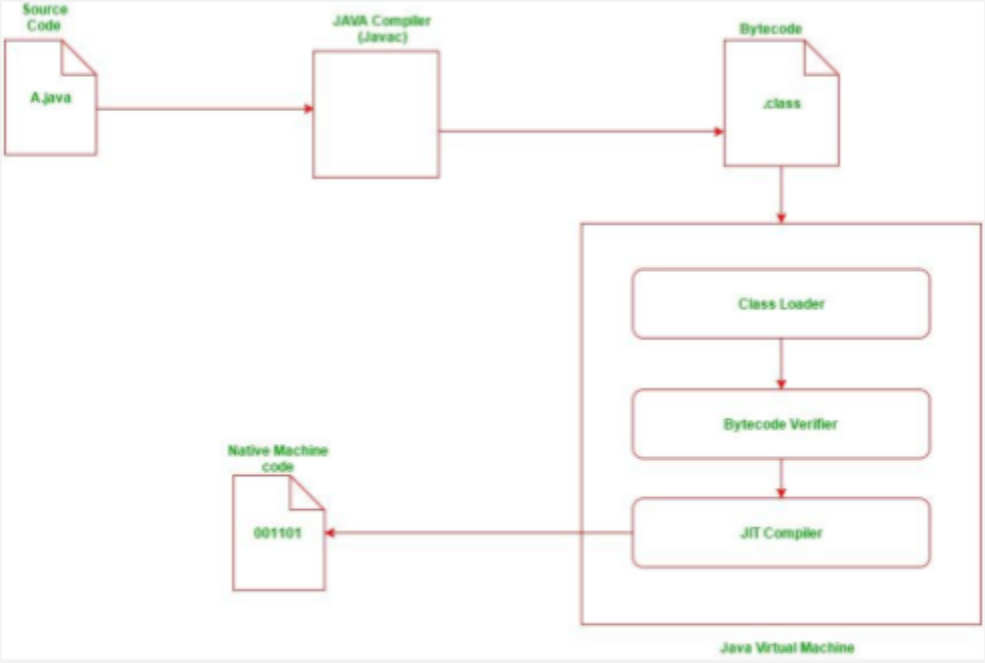
شماره دانشجویی: ۹۹۳۱۰۸۷

ترم زمستان ۰۰ - ۹۹

# سوال اول

- **الف)**
  - JDK[1] پکیجی است شامل نرم‌افزارهای دیگری از جمله JRE، مفسر، کامپایلر و ... که به منظور توسعه‌ی یک برنامه‌ی جاوا ساخته شده است و برای اجرا و کامپایل کردن یک فایل جاوا به این پکیج نیاز است.
  - JRE[2]، همان‌طور که از اسمش معلوم است، وظیفه‌ی اصلی آن ساخت و فراهم کردن یک محیط مجازی برای JVM به منظور اجرای یک برنامه‌ی جاوا است. به طور دقیق‌تر، JRE تمام کلاس‌های مورد نیاز برای اجرای برنامه را بارگذاری کرده و همچنین مدیریت دسترسی برنامه به حافظه را تأمین می‌کند. درست مانند یک سیستم‌عامل برای نرم‌افزارها.
  - JVM[3] یا ماشین مجازی جاوا، وظیفه‌ی اجرا کردن بایت‌کدهای ترجمه شده توسط JDK جاوا را بر عهده دارد. هر سیستم عامل یک JVM مختص به خود را دارد که با ساختن یک ماشین مجازی، امکان اجرای بایت‌کدهای را به آن سیستم‌عامل می‌دهد.
- **ب)**
  - فرآیند کامپایل تا اجرا شدن یک برنامه‌ی جاوا را می‌توان به دو مرحله‌ی اصلی تقسیم نمود: کامپایل شدن آن به بایت‌کدها، اجرا شدن توسط ماشین مجازی. (شکل ۱)

[1] Java Development Kit
[2] Java Runtime Environment
[3] Java Virtual Machine




شکل ۱

- کامپایل شدن یک برنامه‌ی جاوا، همانند زبان‌های برنامه‌نویسی دیگر، توسط یک کامپایلر انجام می‌گیرد. کامپایلر جاوا، که در پکیج JDK قرار دارد، کدهای نوشته شده به زبان جاوا را به یک زبان غیر وابسته به ماشین[4] تبدیل می‌کند که اصطلاحاً به آن‌ها بایت‌کد[5] گفته می‌شود. به این صورت فایل‌های با پسوند ".java" به فایل‌های با پسوند ".class" تبدیل می‌شوند.
- شروع فرآیند اجرا شدن یک برنامه‌ی جاوا از بعد مرحله‌ی کامپایل شدن آن است. همین مرحله را می‌توان به سه مرحله‌ی دیگر تقسیم نمود: بارگذاری کلاس‌ها، اعتبارسنجی بایت‌کدها و در نهایت JIT کامپایلر.
  - بارگذاری کلاس‌ها: JVM برای اجرای برنامه، با شروع از کلاس اصلی (Main)، کلاس‌های مورد نیاز را بارگذاری می‌کند و به صورت پویا یک بایت‌کد را درست می‌کند.

[4] Machine Independent
[5] Byte Code

- اعتبارسنجی بایت‌کدها: بعد از آنکه بایت‌کدها ساخته شد و کلاس‌ها بارگذاری شد، نوبت به اعتبارسنجی بایت‌کدها می‌رسد. JDK در این مرحله، بررسی می‌کند که بایت‌کدهای ساخته شده، درست باشد و هنگام اجرا به مشکلی بر نمی‌خورد.
- JIT کامپایلر[6]: این مرحله، مرحله‌ی آخر است که بایت‌کدها به کد ماشین تبدیل می‌شوند و توسط سیستم‌عامل قابل اجرا می‌شوند.

- منبع: وب‌سایت گیکس‌فورگیکس

## ج)

- نوع‌های ابتدایی یا Primitive Data Types، ساده‌ترین نوع دیتا موجود در جاوا هستند که به طور پیش‌فرض در جاوا وجود داشته‌اند و نوع‌های دیگر، با استفاده از آنها پیاده‌سازی شده‌اند. در کل ۸ نوع ابتدایی boolean, byte, char, short, int, long, float, double در جاوا وجود دارد که به یک منظور ساخته شده‌اند، ذخیره کردن مقادیر ساده‌ی از یک نوع. این نوع‌ها دارای متد نیستند و نمی‌توان عملیات جدید بر روی آنها تعریف نمود. باقی نوع‌ها، هر کدام یک شئ از یک کلاس هستند و تنها نوع‌های ابتدایی هستند که عضو هیچ کلاسی نیستند.

## د)

- برنامه‌نویسی شئ‌گرا[7]

- در این دیدگاه، هر چیزی را به شکل یک شئ نگاه می‌کنند و برای آن، یک کلاس تعریف می‌کنند. و به ازای هر کلاس، تمام ویژگی و ساختارش را به طور جداگانه تعریف می‌کنند و در نهایت به ازای هر شئ، یک نمونه از کلاس پیاده‌سازی شده می‌سازند. از مزایای این نوع می‌توان به ارث‌بری کلاس‌ها و کپسوله سازی کلاس‌ها[8] اشاره کرد.

[6] Just-In-Time Compiler
[7] Object Oriented Programming
[8] Encapsulation

- برنامه‌نویسی ساختار یافته[9]
  - نوعی از دیدگاهی است که برنامه مانند یک ساختار عمل می‌کند. به این معنا که برنامه دستور به دستور اجرا می‌شود و در کد آن، هیچ دستوری مانند GO TO نخواهیم داشت. در نتیجه دستورات برنامه به نوبت اجرا می‌شوند. از مزایای این نوع می‌توان به خوانا بودن و مفهوم بودن آن و باگ زدایی راحت‌تر آن اشاره نمود.
- برنامه‌نویسی عملکردی[01]
  - در این دیدگاه، هر زیر مسئله را به صورت تابعی در نظر می‌گیرند که تعدادی ورودی می‌گیرد و پاسخ آن زیر مسئله را بر می‌گرداند. از مزایای این نوع می‌توان به این اشاره نمود که به دلیل استقلال نسبی توابع در ترتیب فراخوانی، می‌توان برنامه را به روش‌های گوناگونی پیاده‌سازی کرد.

- منبع: وب‌سایت دانشگاه فلوریدا

- ه)
  - Class
    - در برنامه‌نویسی شئ‌گرا، کلاس‌ها را می‌توان دسته‌ای از ساختارها دانست که ویژگی‌ها و خصوصیات اشیاء را نگه می‌دارد.
  - Object
    - به هر نمونه‌ی ساخته‌شده از هر کلاس، یک شئ می‌گوییم. اشیاء یک کلاس دارای ویژگی‌ها و متد های مشترک هستند.
  - Constructor

[9] Structured Programming
[10] Functional Programming

- کانستراکتورها نوع خاصی از متدها هستند که برای ساخت یک شئ مورد استفاده قرار می‌گیرند.

- Method
  - به تابع‌های تعریف شده در یک کلاس، که به منظور اضافه کردن یک کاربرد است، متد می‌گویند.
- Parameter
  - به متغیرهای ورودی یک تابع، آرگومان یا پارامتر می‌گویند.
- Instance
  - در حالت کلی، نمونه یا مثال را همان شئ (Object) در نظر می‌گیرند. اما به طور دقیق‌تر نمونه، یک کپی از یک شئ است.

- و)
  - تابع، تکه کدی است که می‌تواند با ورودی گرفتن تعدادی پارامتر صدا زده شود و پس از اجرای دستوراتی، متغیری را برگرداند. متد اما، تابعی است که درون یک کلاس تعریف شده است و توسط اشیاء آن کلاس صدا زده می‌شود. در واقع می‌توان متد را نوعی تابع در نظر گرفت.

# سوال دوم

- ۱)
  - غلط است، زیرا جاوا یک زبان ایستا[11] است. به این معنا که هر متغیر، پیش از استفاده باید تعریف شده باشد و در غیر این صورت، برنامه با خطا مواجه خواهد شد و اجرا و کامپایل نمی‌شود.
- ۲)
  - صحیح است. فایل‌های با پسوند class. حاوی بایت‌کدها هستند که می‌توانند توسط JVM اجرا شوند.
- ۳)
  - غلط است. به طور کلی، آرایه در جاوا، چند دیتا با جنس یکسان را ذخیره می‌کند. اما با استفاده از کلاس Object، می‌توان آرایه‌ای ساخت که چند دیتا با جنس متفاوت را ذخیره نماید.
- ۴)
  - غلط است. متغیرهای مرجع، در واقع مانند پوینترها در زبان C هستند. در نتیجه، عملگرهای اصلی برای آنها تعریف نمی‌شوند. البته در زبان C، برای پوینترها عملگر منها تعریف می‌شود که استثنا است.
- ۵)
  - غلط است. زبان برنامه‌نویسی‌ای که کامل شئ‌گرا است، ۷ شرط را دارد که جاوا ۲ شرط از آنها را ندارد. برای مثال در یک زبان کامل شئ‌گرا، تمام متغیرها باید خودشان یک شئ باشند و نباید چیزی به نام متغیرهای اولیه وجود داشته باشد. در نتیجه جاوا را نمی‌توان یک زبان کامل شئ‌گرا دانست. (منبع)

[11] Statically-Typed Language

# سوال سوم

- **A)** Java classes contain **Methods** (which implement class behaviors) and **Fields** (which implement class/object data).
- **B)** In Java, the unit of programming is the **Classes**, from which **Objects** are eventually instantiated.
- **C)** Java programmers concentrate on creating their own user-defined types, called **Classes**.
- **D)** The **java** command of JRE executes an application.
- **E)** The **javac** command of JDK compiles a Java program.
- **F)** A Java program file must end with the **.java** file extension.
- **G)** When a Java program is compiled, the file produced by the compiler ends with the **.class** file extension.
- **H)** The file produced by the Java compiler contains **bytecodes** that are interpreted to execute a Java applet or application.

# سوال چهارم

- ابتدا باید ورودی متد printArray را به یک آرایه از اعداد تغییر دهیم. سپس باید به جای علامت , در حلقه‌ی for خط ۱۷، علامت ; را بگذاریم. همچنین علامت ; خط ۱۵ را باید اضافه کنیم. بعد از انجام این مراحل، برنامه کامپایل می‌شود.
- باید شرط حلقه‌ی for خط ۴، را به $i < n$ تغییر دهیم. پس از آن، کلاس sort را به صورت زیر تغییر می‌دهیم:

```
private static void sort(int arr[]) {
        int n = arr.length;
        for (int gap = n / 2; gap >= 0; gap -= 1) {
            for (int i = gap; i < n; i += 1) {
                int temp = arr[i];
                int j;
                for (j = i; j >= gap && arr[j - gap] > temp; j -= gap)
                    arr[j] = arr[j - gap];
                arr[j] = temp;
            }
        }
    }
```

- الگوریتم متد sort به ازای gap=1، همان الگوریتم مرتب‌سازی حبابی[21] است و از آنجا که متغیر gap حتماً برابر با یک می‌شود، پس آرایه مرتب می‌شود.

[12] Bubble Sort