به نام خدا

دستور کار کارگاه برنامه‌نویسی پیشرفته

جلسه اول

مقدمه‌ای بر جاوا و برنامه‌نویسی ساخت‌یافته در آن

مقدمه

مطالبی که در این جلسه مورد بررسی قرار می‌گیرند عبارتند از:

- آشنایی با نحوه اجرای برنامه‌ها در جاوا
- ساختار کلی برنامه‌ها در زبان برنامه‌نویسی جاوا
- کامپایل و اجرای کد در جاوا
- نصب و راه‌اندازی محیط توسعه یکپارچه (IDE[1])
- آشنایی با نحوه ایجاد یک پروژه
- آشنایی با قواعد نحوی زبان جاوا
- مروری بر ساختارهای کنترلی و متغیرها در جاوا

نحوه اجرای برنامه در جاوا

JDK چیست و چه کاری انجام می‌دهد؟

یکی از ویژگی‌های برجسته زبان جاوا، Cross Platform یا Platform Independent بودن آن است؛ به این معنی که برنامه‌های نوشته شده در زبان جاوا می‌توانند روی پلتفرم‌ها و سیستم‌عامل‌های مختلفی مانند ماشین‌های ویندوزی، سیستم‌عامل‌های لینوکس و موارد دیگر اجرا شوند.

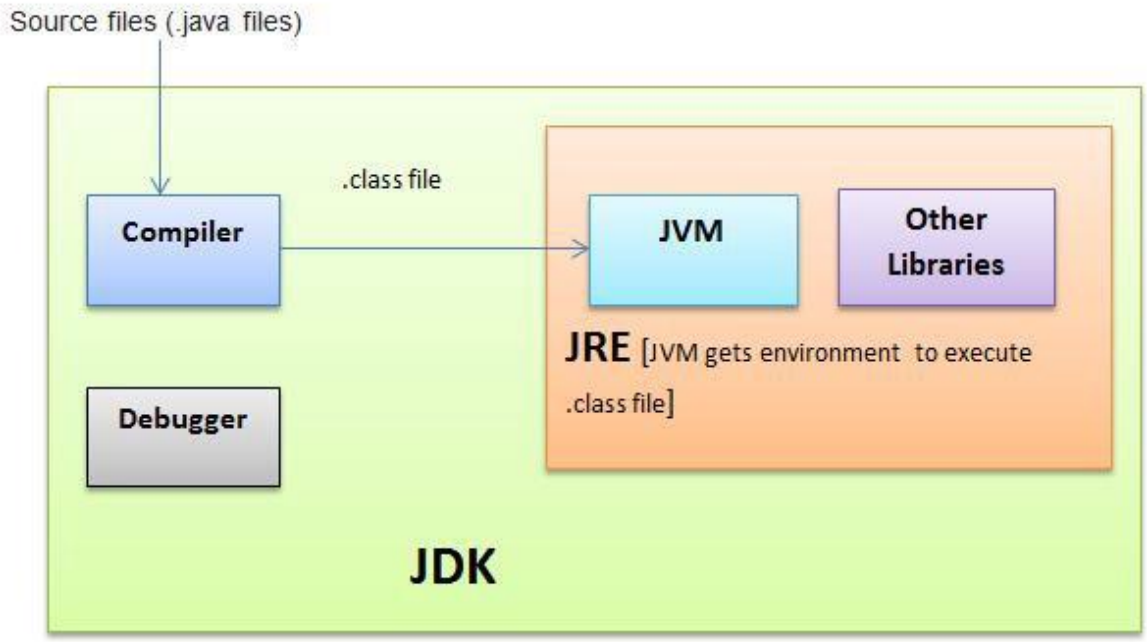
زبان جاوا به دلیل داشتن یک ماشین مجازی به نام JVM[2]، قادر به ارائه چنین ویژگی مهمی است. این ماشین قابلیت اجرای کدهای جاوا را به کاربران بدون درنظر گرفتن نوع سخت‌افزار یا نرم‌افزار را می‌دهد.

1 Integrated Development Environment

2 Java Virtual Machine

از همین رو، آشنایی با ماشین مجازی جاوا و نقش آن در اجرای برنامه‌ها از اهمیت بالایی برخوردار است.

JDK[3] یک مجموعه‌ای است که قابلیت توسعه و اجرا کدهای جاوا را به کاربران می‌دهد. همانطور که در شکل زیر نشان داده شده، یک بخش از JDK مربوط به اجرای کد است که به آن JRE گفته می‌شود. JVM که وظیفه‌ی اجرای کد را دارد قسمتی از JRE است.



شکل 1 - اجزای JDK

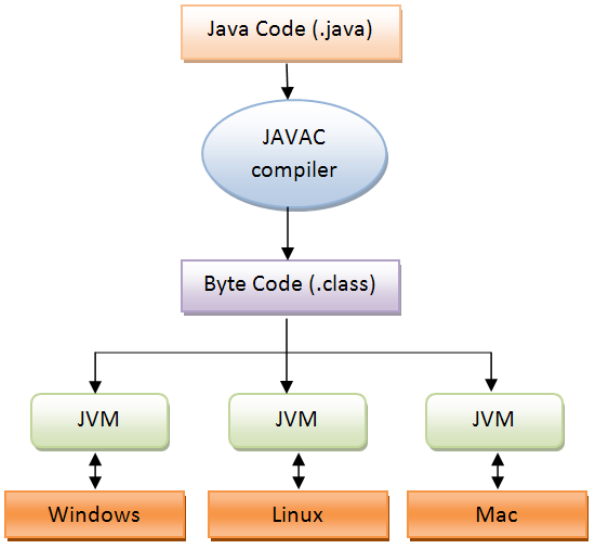
فرایند کامپایل و اجرای برنامه‌ها در جاوا با زبان C که قبلا آموخته‌اید تفاوت دارد. در زبان C برنامه ابتدا نوشته می‌شد و سپس به کامپایلر داده می‌شد و کامپایلر آنرا به دستورات قابل فهم توسط ماشین تبدیل می‌کرد. این فرایند در جاوا بگونه‌ی دیگری است. فرایند کامپایل این برنامه‌ها از دو بخش تشکیل شده است:

- کامپایل‌کردن کد منبع برنامه به یک زبان میانی (bytecode)
- اجرای فایل برنامه تبدیل‌شده به زبان میانی توسط کامپایلر/مفسر در محیط زمان اجرای جاوا (JRE[4])

3 Java Development Kit

4 Java Runtime Environment

طی این فرایند، فایل‌های با پسوند .java که فایل‌های متنی ساده‌ای هستند و فقط متن اصلی برنامه را ذخیره می‌نمایند، به کامپایلر جاوا داده می‌شوند. کامپایلر با دریافت این فایل‌ها، فایل‌هایی به یک زبان میانی تولید می‌نماید که برای مفسرها/کامپایلرهای محیط زمان اجرای جاوا قابل فهم هستند. این فایل‌های زبان میانی با پسوند .class ذخیره می‌شوند. زبان میانی جاوا برای تمام مفسرها/کامپایلرهای جاوا روی هر پلتفرمی قابل فهم است. کافیست این فایل‌های میانی، به مفسر مربوطه روی یک پلتفرم داده شود تا مفسر بتواند آن را اجرا نماید. به این ترتیب، نیازی به دریافت کامپایلر جاوا برای تمام پلتفرم‌ها و کامپایل کردن کد اصلی برای تمام پلتفرم‌ها به طور جداگانه وجود ندارد. شکل 2، این فرایند را به طور کامل نمایش می‌دهد.



شکل 2 – Cross Platform بودن جاوا

اجرای برنامه‌ها از طریق CLI (Command-Line Interface)

در بخش قبل، با نحوه اجرای برنامه در محیط توسعه یکپارچه آشنا شدیم. در این قسمت می‌خواهیم فرایند اجرای برنامه در محیط ترمینال یا دستوری را بررسی نماییم. پس از نصب JDK و برای استفاده از آن در سیستم‌عامل ویندوز کافی است متغیر محیطی[5] PATH را به درستی تنظیم کنید تا به آن دسترسی داشته باشید.

مراحل انجام کار

نصب JDK

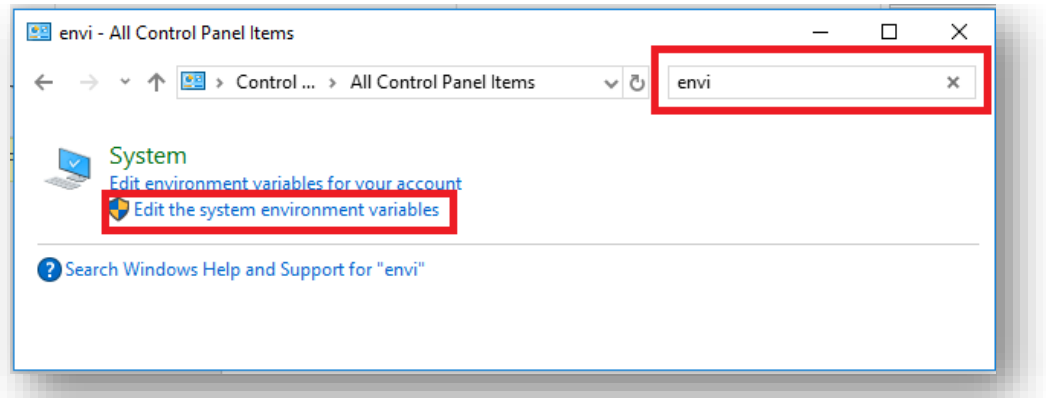
5 Environment Variable

برنامه‌های نوشته‌شده به زبان جاوا، برای کامپایل، نیازمند JDK[6] (یا Java SDK[7]) هستند. برای دریافت JDK مناسب با سیستم‌عامل خود می‌توانید به آدرس زیر مراجعه نمایید.

http://ceit.aut.ac.ir/~ghaffarian/download.html

نحوه تنظیم متغیرهای محیطی در Windows 10

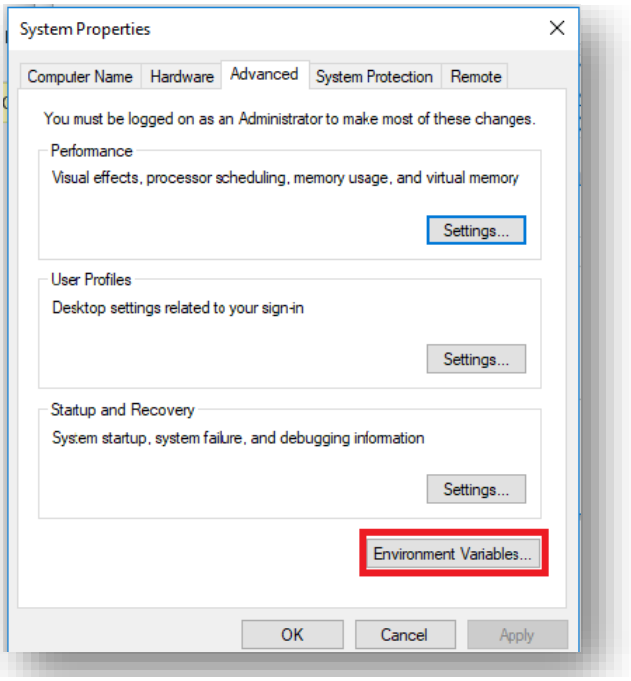
برای این کار کافی است تا از طریق Control Panel و در قسمت Search، عبارت environment را تایپ کنید. سپس گزینه Edit the System Environment variables را انتخاب کرده و از پنجره باز شده عبارت Environments را انتخاب کنید. سپس از پنجره باز شده روی PATH کلیک کرده و گزینه Edit را فشار دهید. از داخل پنجره جدید گزینه New را انتخاب کرده و آدرس پوشه bin موجود در JDK نصب شده بر روی سیستم تان را به انتهای مقادیر موجود اضافه نمایید (آدرسها با ; از هم جدا می شوند).



شکل 3 – جست و جو در Control Panel

[6] Java Development Kit

[7] Software Development Kit



شکل 4 – System Properties

اکنون سیستم‌عامل ویندوز شما برای اجرای برنامه‌های مبتنی بر جاوا پیکربندی شده است (برای محیط‌های لینوکسی و Mac تنظیمات به نحو دیگری است که با جستجو می توانید آنها را پیدا کنید). بعد از دانلود و نصب JDK، برنامه‌ی Command Prompt در ویندوز یا Terminal در لینوکس را باز کنید. دستور java –version را وارد کنید و درصورتی که دستور تعریف شده بود دستورکار را ادامه دهید. در غیر اینصورت سیستم‌عامل شما هنوز برنامه‌ی جاوا را نشناخته و در مراحل نصب آن مشکلی بوجود آمده است.

```
C:\Users\Alireza>java -version
java version "1.8.0_144"
Java(TM) SE Runtime Environment (build 1.8.0_144-b01)
Java HotSpot(TM) 64-Bit Server VM (build 25.144-b01, mixed mode)
```

شکل5 - خروجی دستور java -version

سپس یک پوشه درست کرده و در آن یک فایل به نام HelloWorld و با پسوند java. بسازید. فایل ساخته شده را با یک نرم افزار editor مانند Notepad باز کنید و قطعه کد زیر را داخل آن بنویسید.

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

سپس آن‌را ذخیره کرده و به محیط cmd یا terminal بازگردید و به directory بروید که فایل جاوا را در آن ساخته‌اید. سپس از دستور زیر برای کامپایل کردن کدها استفاده کنید. ورودی دستور javac باید نام فایلی باشد که می‌خواهید آنرا کامپایل کنید.

javac HelloWorld.java

سپس با اجرای دستور زیر JVM را فراخوانی کنید تا byte code های شما را تفسیر[8] کند.

java HelloWorld

یکی از راه‌های گرفتن ورودی از کاربر آن است که ورودی‌ها را قبل از اجرای برنامه به آن بدهیم (به نظر شما دلیل آن چه می‌تواند باشد؟) .در این حالت جاوا به این نوع ورودی‌ها، آرگومان[9] می‌گویند. کد زیر را در فایل Main.java کپی کنید.

```
public class ArgumentPassing {
    public static void main(String[] args) {
        String firstArgument = args[0];
        System.out.println(firstArgument);
    }
}
```

سپس دوباره کد را با دستور javac کامپایل کنید. سپس دستور زیر را برای فراخوانی JVM استفاده کنید.

java ArgumentPassing Great!

انجام دهید

برنامه‌ای بنویسید که با استفاده از ساختارهای حلقه، تمامی آرگومان‌های ارسال‌شده به تابع main را چاپ نماید. این برنامه را از طریق CLI کامپایل و اجرا نمایید.

نصب و راه‌اندازی محیط توسعه یکپارچه

[8] Interpret

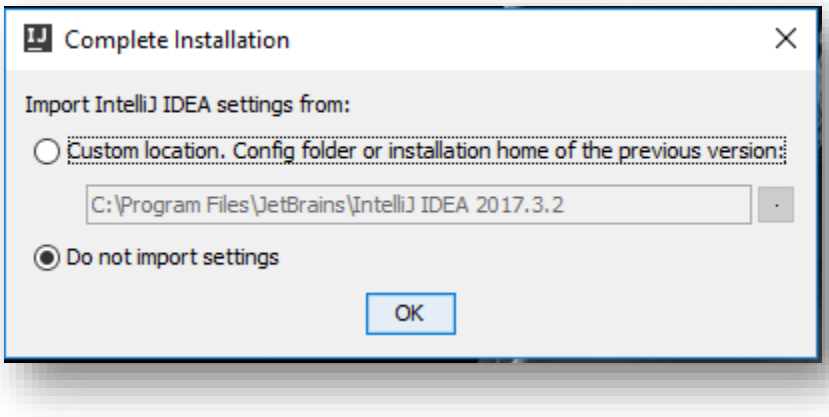
[9] Argument

در این دوره از کارگاه‌های برنامه‌نویسی پیشرفته، از نرم‌افزار IntelliJ به عنوان محیط توسعه یکپارچه استفاده می‌نماییم[10]. این نرم‌افزار، یکی از محصولات شرکت JetBrains است که دارای دو نسخه‌ی Community و Ultimate است. برای شروع می‌توانید از نسخه Community استفاده کنید ولی در ادامه نسخه Ultimate آنرا نصب کنید.

https://www.jetbrains.com/idea/download

مراحل نصب IntelliJ IDEA

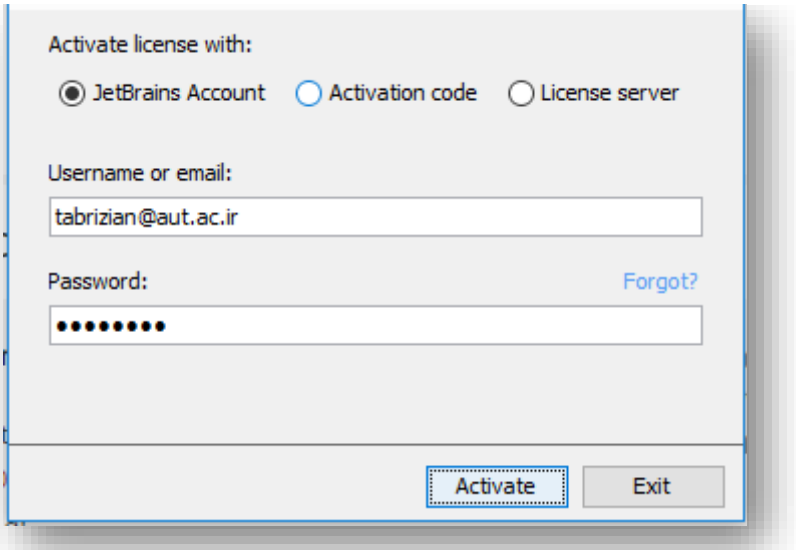
با اجرای فایل نصب نرم‌افزار، پنجره شکل 5 نمایش داده می‌شود. در صورتی‌که قبلاً از نسخه دیگری از این نرم‌افزار استفاده می‌کردید، می‌توانید با انتخاب گزینه اول، تمامی تنظیمات اعمال‌شده روی نسخه قبلی را برای نسخه جدید نیز استفاده نمایید. در غیر این صورت گزینه دوم را انتخاب نموده و به مرحله بعدی بروید.



شکل 5 – صفحه اول پس از اجرای نصب

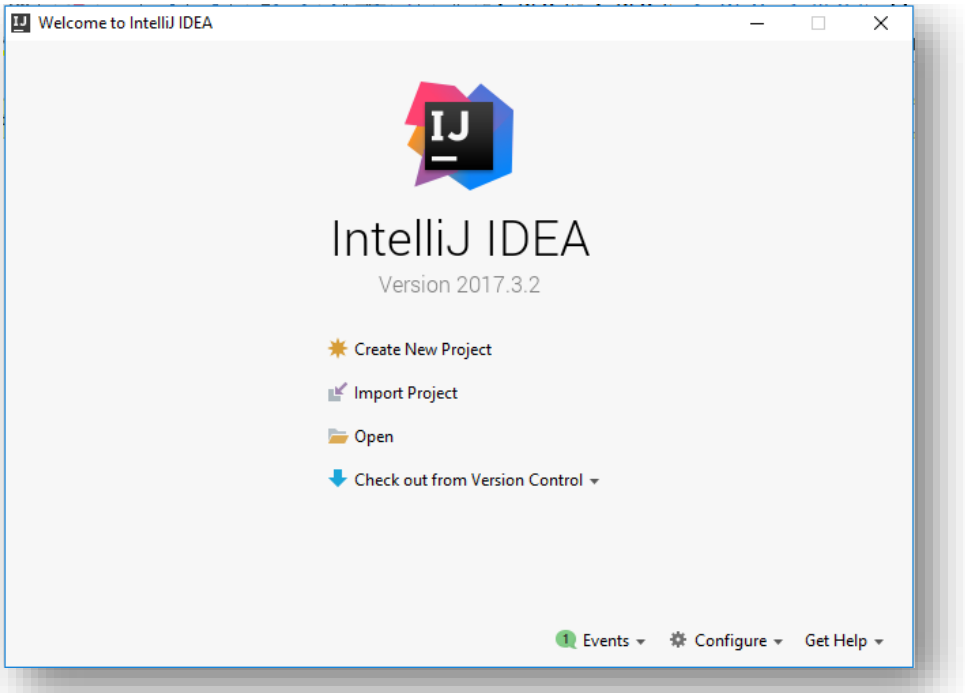
در مرحله بعدی و مطابق شکل 6، پنجره‌ای برای فعال‌سازی نرم‌افزار نمایش داده می‌شود. در صورتی که در سایت jetbrains حساب کاربری دارید، در این پنجره می‌توانید با انتخاب گزینه فعال‌سازی از طریق حساب کاربری و وارد نمودن نام و رمز عبور حساب کاربری خود، نرم‌افزار خود را فعال نمایید.

10 بدیهی است با توجه به محدودیت‌های موجود، امکان بررسی تمامی ابعاد و ویژگی‌های این محصول در این دوره وجود ندارد. مطلوب است دانشجویان مطالعات بیشتری در رابطه با محیط توسعه مورد استفاده، ابعاد، ویژگی‌ها و امکانات آن انجام دهند.



شکل 6 – فعال‌سازی نرم‌افزار

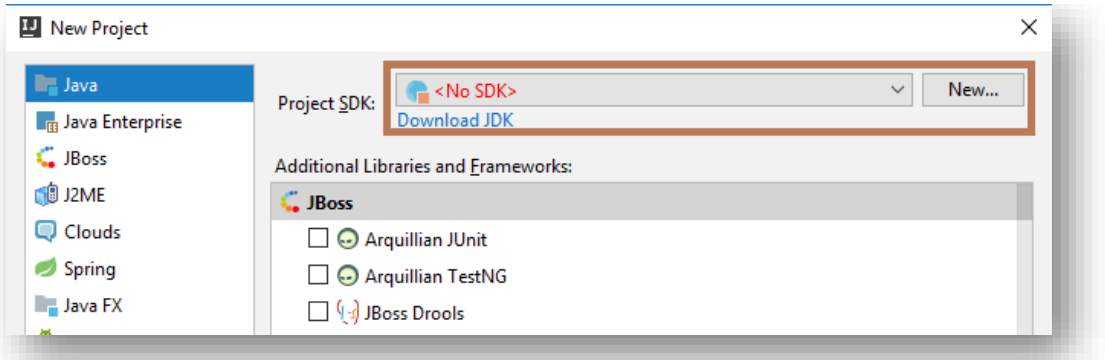
با اتمام فرایند نصب نرم‌افزار، پنجره شکل 7 نمایش داده می‌شود. در این مرحله، فرایند نصب نرم‌افزار تکمیل شده است و از این پس می‌توانید شروع به برنامه‌نویسی نمایید.



شکل 7 – صفحه ایجاد پروژه جدید

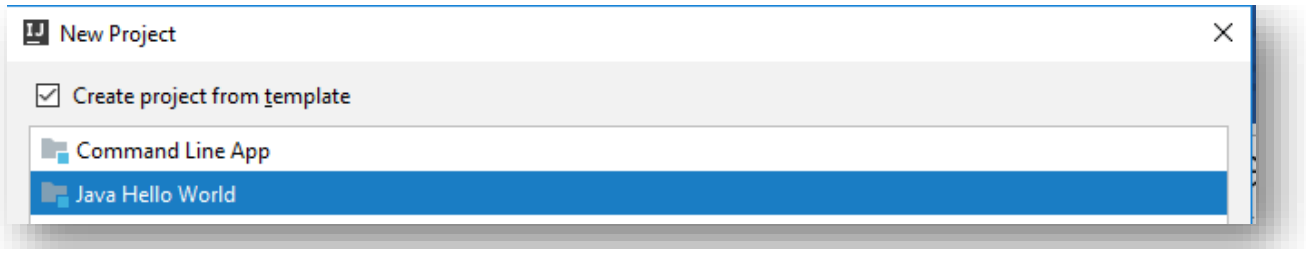
ایجاد پروژه جدید

با انتخاب گزینه Create New Project در پنجره شکل 7، می‌توانید یک پروژه جاوای جدید ایجاد نمایید. با انتخاب این گزینه، پنجره‌ای مشابه شکل 9 نمایش داده می‌شود. در منوی سمت چپ می‌توانید انواع پروژه‌های قابل ایجاد را مشاهده نمایید. در این منو، روی اولین گزینه، پروژه Java کلیک نمایید. در منوی سمت راست، باید SDK مورد استفاده در این پروژه را به محیط توسعه یکپارچه معرفی نمایید. با انتخاب گزینه New، مسیر نصب SDK را مشخص نموده و مراحل را تا انتهای کار دنبال نمایید.



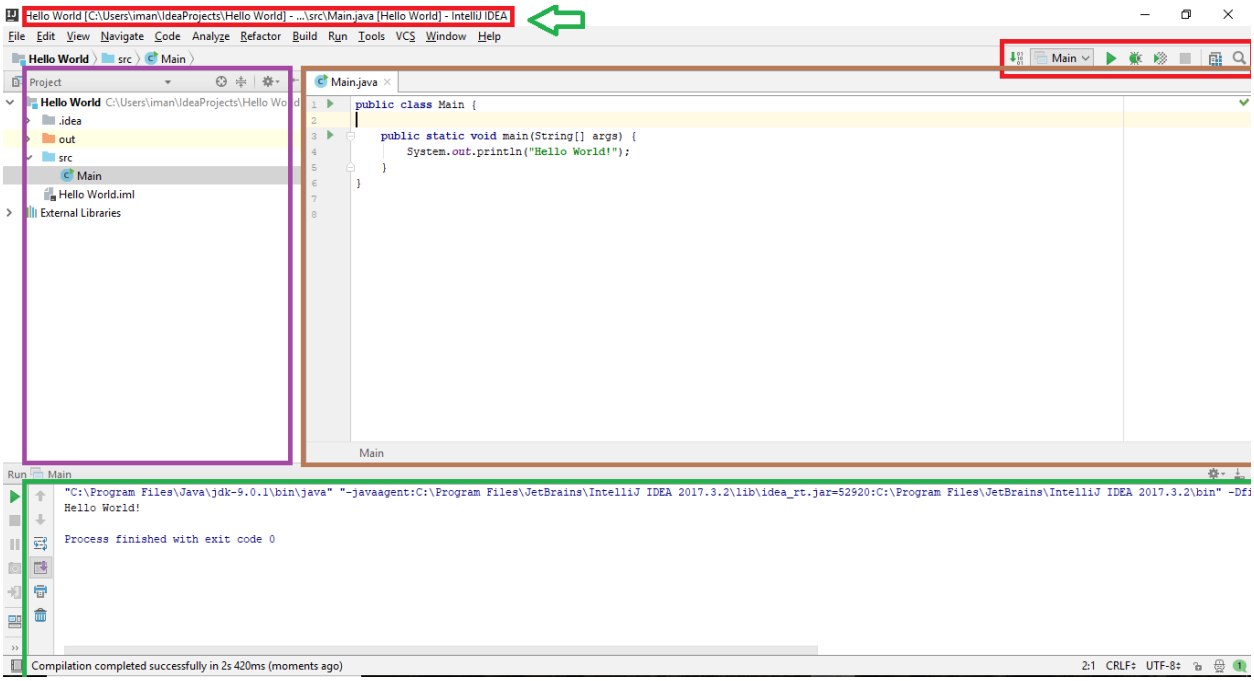
شکل 8 – راه اندازی SDK

پس از نصب SDK، پنجره‌ای مشابه شکل 9 نمایش داده می‌شود. در این پنجره می‌توانید قالب آماده‌ای را برای پروژه خود انتخاب نمایید. روی گزینه Java Hello World کلیک کرده و به مرحله بعدی بروید.



شکل 9 – انتخاب نوع پروژه جدید

با ایجاد پروژه جدید، پنجره اصلی نرم‌افزار، مشابه با شکل 10 نمایش داده می‌شود.



شکل 10 – محیط IntelliJ IDEA

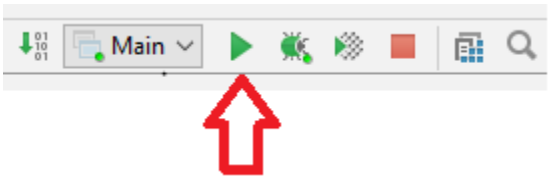
آدرس ریشه پروژه، در نوار عنوان[11] نمایش داده می‌شود. نوار ابزار در زیر نوار عنوان در بالای صفحه قرار دارد. کادر بنفش‌رنگ ساختار فایل پروژه، بسته‌ها و کلاس‌های تعریف‌شده را نمایش می‌دهد. کادر سبزرنگ که در پایین صفحه نشان داده شده است، خروجی برنامه به همراه exit code و اطلاعات مربوط به اجرای برنامه را نشان می‌دهد. کادر قهوه‌ای محیط ویرایش کد را نمایش می‌دهد.

با اتمام فرایند ایجاد پروژه، یک پوشه به نام پروژه در آدرس انتخابی شما ایجاد می‌شود که فایلی به اسم Main.java در آن وجود دارد. با بازکردن این فایل در محیط توسعه می‌توانید کد زیر را در آن مشاهده نمایید.

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

حال برای اجرای برنامه کافی است تا دکمه Run را از منوی زیر اجرا کنید.

11 Title Bar



شکل 11 – منوی اجرای پروژه

انواع داده در جاوا

جدول ۱، انواع داده‌های اولیه[12] مورد استفاده در جاوا را به همراه اطلاعاتی در رابطه با آن‌ها نمایش می‌دهد.

جدول ۱ – جدول انواع داده

Type	Description	Initial Value	Size	Example Literals
boolean	true or false	false	Platform-dependent	true, false
byte	twos complement integer	0	8 bits	
char	Unicode character	\u0000	16 bits	‘a’, ‘\u0041’, ‘\\’
short	twos complement integer	0	16 bits	
int	twos complement integer	0	32 bits	-2, -1, 0, 1, 2
long	twos complement integer	0	64 bits	-2L, -1L, 0L
float	IEEE 754 floating point	0.0	32 bits	1.23e100f,.3f, 3.14F
double	IEEE 754 floating point	0.0	64 bits	1.23456e300d, 1.234e-3d, 1e1d

تعریف متغیر در جاوا

برای تعریف متغیر در جاوا، ابتدا نوع داده و سپس نام متغیرها را در یک خط مشخص می‌نماییم. قطعه کد زیر، نمونه‌ای از تعریف متغیر در جاوا را نمایش می‌دهد.

```
int a, b;
a = 1234;
b = 100;
int[] phoneNumber = new int[4];
phoneNumber [0] = 6;
phoneNumber [1] = 6;
phoneNumber [2] = 4;
```

12 Primitive Data Types

```
phoneNumber [3] = 1;
```

ساختارهای کنترلی در جاوا

قواعد نحوی ساختار کنترلی if در جاوا، کاملاً مشابه با قواعد زبان C است. کد زیر نمونه‌ای از کاربرد این دستور در جاوا را نمایش می‌دهد. خروجی این قطعه کد چیست؟

```
public class ZeroCheker {
    public static void main(String[] args) {
       int number = 0;
       if (number == 0)
           System.out.println("number is 0");
       else
           System.out.println("number is not 0");
    }
}
```

نمونه‌ای از استفاده از ساختار switch-case در زبان جاوا را نمایش می‌دهد. خروجی این قطعه کد چیست؟

```
public class JavaCondition {
    public static void main(String[] args) {
        int day = 4;
        switch (day) {
            case 0:  System.out.println("Sunday");
               break;
            case 1:  System.out.println("Monday");
               break;
            case 2:  System.out.println("Tuesday");
               break;
            case 3:  System.out.println("Wednesday");
               break;
            case 4:  System.out.println("Thursday");
               break;
            case 5:  System.out.println("Friday");
               break;
            case 6:  System.out.println("Saturday");
               break;
            default: System.out.println("invalid day");
        }
    }
}
```

حلقه‌ها

قواعد نحوی استفاده از ساختار while و for در زبان جاوا، کاملاً مشابه با قواعد این ساختارها در زبان برنامه‌نویسی C است. کدهای زیر نمونه‌هایی از استفاده از این ساختارها را در زبان جاوا مشخص می‌کند. در هر مورد بگویید خروجی قطعه کد نمایش داده شده چیست؟

قطعه کد 1

```
public class LoopInJava {
    public static void main(String[] args) {
        // print out special cases whose ordinal doesn't end in th
        System.out.println("1st Hello");
        System.out.println("2nd Hello");
        System.out.println("3rd Hello");
        int numOfHellos = 4;
        while (numOfHellos <= 10) {
            System.out.println(numOfHellos  + "th Hello");
            numOfHellos  = numOfHellos  + 1;
        }
    }
}
```

و قطعه کد 2

```
public class LoopInJava {
    public static void main(String[] args) {
        // print out special cases whose ordinal doesn't end in th
        System.out.println("1st Hello");
        System.out.println("2nd Hello");
        System.out.println("3rd Hello");
        // count from numOfHellos = 4 to 10
        for (int numOfHellos = 4; numOfHellos <= 11; numOfHellos++) {
            System.out.println(numOfHellos + "th Hello");
        }
    }
}
```

گرفتن ورودی از کاربر به کمک Scanner

برای دریافت ورودی از کاربر، روش‌های مختلفی وجود دارد. از آنجا که استفاده از هر یک از این روش‌ها در جاوا، نیازمند آشنایی با مفاهیم شی‌گرایی است، در این قسمت با یکی از ساده‌ترین روش‌ها آشنا شده و بررسی روش‌های دیگر را به آینده موکول می‌کنیم.

یکی از کلاس‌های موجود در کتابخانه‌های استاندارد جاوا، کلاس Scanner است. این کلاس، مجموعه‌ای از متدهای مورد نیاز و کاربردی را برای ورودی گرفتن را پیاده‌سازی می‌نماید. در این قسمت، سه نمونه از پرکاربردترین متدهای این کلاس را معرفی می‌نماییم. برای کسب اطلاعات بیشتر در مورد متدهای موجود در این کلاس، می‌توانید به https://docs.oracle.com/javase/7/docs/api/java/util/Scanner.html و یا https://www.cs.utexas.edu/users/ndale/Scanner.html مراجعه نمایید.

برای استفاده از کلاس Scanner، ابتدا باید یک نمونه از آن را ایجاد نمایید. ایجاد یک نمونه جدید از این کلاس، با قطعه کد زیر انجام می‌شود. در این قطعه کد، یک نمونه جدید به نام inputStream از کلاس Scanner ایجاد شده است که با توجه به ورودی System.in، ورودی را از کیبورد دریافت می کند.

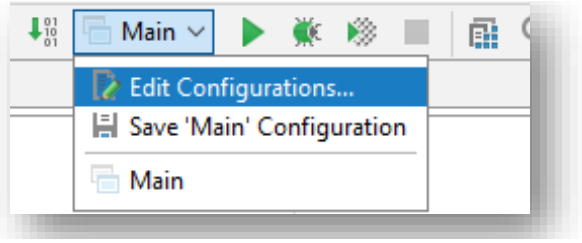
```
Scanner inputStream = new Scanner(System.in);
```

با ایجاد این نمونه جدید می‌توانیم از متدهای آن برای دریافت ورودی از کاربر، از طریق کنسول، استفاده نماییم. سه نمونه از پرکاربردترین متدهای پیاده‌سازی شده در کلاس Scanner متدهای nextInt (دریافت یک عدد صحیح)، nextLine (دریافت یک رشته شامل یک خط از ورودی کنسول) و nextFloat (دریافت یک عدد اعشاری یا ممیز شناور) هستند. قطعه کد زیر، نحوه استفاده از این متدها را نمایش می‌دهد.

```
int number = inputStream.nextInt();
String userInput1 = inputStream.nextLine();
float real = inputStream.nextFloat();
String userInput2 = inputStream.nextLine();
```

پاس دادن Argument به برنامه‌های جاوا در محیط IntelliJ IDEA

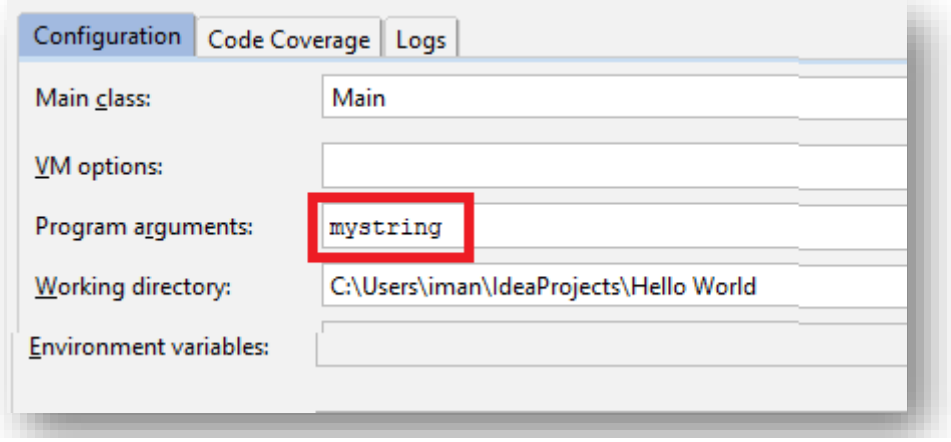
برای مقداردهی این آرایه در محیط IntelliJ لازم است تا عملیات‌های زیر را انجام دهید.



شکل 12 – تغییر پیکربندی اجرای برنامه

ابتدا مطابق با شکل 12، گزینه Edit Configurations را انتخاب می‌کنید تا نحوه اجرای برنامه را تغییر دهید. پس از انجام این کار در پنجره‌ای که مطابق با شکل 13 می‌باشد، عبارت mystring را در Program arguments وارد کنید. این بخش شامل رشته‌هایی می‌شود که در آرایه‌ی String[] args قرار داده می‌شوند. عباراتی که در اینجا قرار می‌دهید با فاصله (space) از یکدیگر جدا می‌شود و به ترتیب در آرایه args قرار می‌گیرد که در ورودی تابع main قابل دسترسی است.

پس از انجام این تغییرات گزینه Ok را بزنید و از این صفحه خارج شوید.



شکل 13 – تعیین Argument

حال برنامه شکل 13 را اجرا کنید و خروجی آن را تفسیر کنید.

انجام دهید

۱) برنامه‌ای بنویسید که دو عدد دریافت کند و بررسی کند آیا این دو عدد نسبت به هم اول هستند یا خیر.

۲) برنامه‌ای بنویسید که یک جدول ضرب ۱۰ در ۱۰ را حساب کند و در کنسول نمایش دهد.

نکاتی درباره برنامه‌نویسی در جاوا

- جاوا یک زبان برنامه‌نویسی حساس به حالت حروف[13] است.
- طبق قرارداد، اسامی متدها (تابع‌ها) باید با حرف کوچک شروع شود. اگر اسم متد از چند کلمه تشکیل شده است، باید اولین حرف کلمه داخلی بزرگ نوشته شود (اصطلاحاً CamelCase). مانند قطعه کد زیر:

13 Case Sensitive

```
public void myMethod()
```

- اسم فایل باید حتما با اسم کلاس public مطابقت داشته باشد.
- برنامه‌های جاوا از متد main با شکل زیر آغاز می‌شوند.

```
public static void main(String[] args)
```

اشکال‌زدایی

۱. در ادامه شما چند قطعه کد مشاهده خواهید کرد و وظیفه شما آن است که اشکالات این قطعه کدها را پیدا کنید.

قطعه کد اول:

```
    public void my_method () {
    }
```

قطعه کد دوم:

```
public class main {
    public void myanothermethod () {
    }
}
```

۲. توضیح دهید که اگر ما بخش‌های زیر را از برنامه Hello World برداریم، چه خطاهایی رخ خواهد داد.

الف) ;

ب)یکی از ”ها

ج) یکی از آکلادها

۳. توضیح دهید چرا قطعه کد زیر اجرا نمی‌شود؟

```
public class Hello {
   public static void main() {
      System.out.println("Doesn't execute");
   }
}
```

۴. جاوا چه نوع زبان برنامه‌نویسی است: مفسری یا کامپایلری؟ توضیح دهید.