

آموزش زبان برنامه نویسی جاوا

ساخت اولین برنامه جاوا در Eclipse

جلسه چهارم

نویسنده: رحمان زارعی

جاوا را ساده، آسان و شیرین بنوشید!!!!

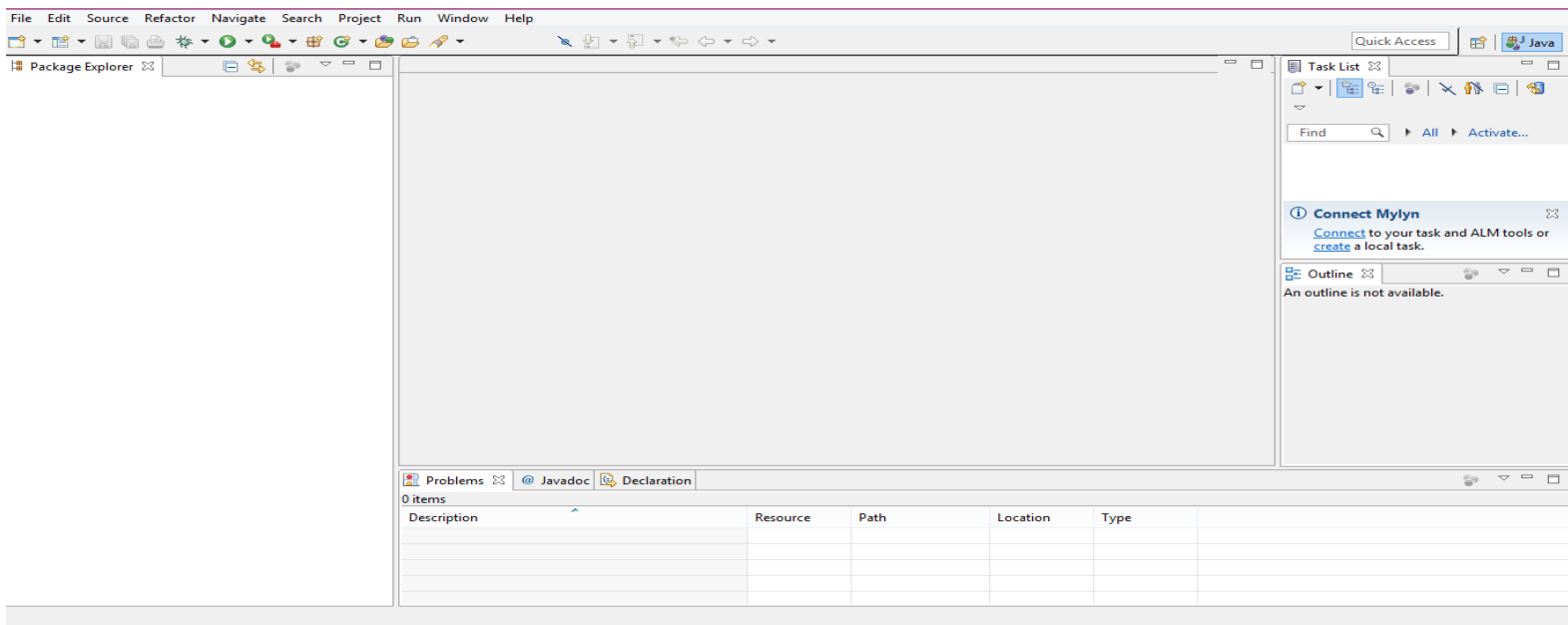


در جلسه دوم در محیط Notepad کد زدیم و بصورت دستی ذخیره، کامپایل و اجرا کردیم. این روش کد زدن خیلی زمان بر و سخت و خشک بود حالا در این جلسه قصد داریم راهی آسان و لذت بخش رو بهتون پیشنهاد بدیم!!! کار با برنامه ایدکیپس (Eclipse)!! اما از این به بعد به راحتی کد های جاوا مون رو در ایدکیپس میزنیم ، کامپایل و اجرا میکنیم.

نوشتن اولین برنامه جاوا در Eclipse:

صورت مسئله: برنامه ای به زبان جاوا بنویسید که بعد از اجرا شدن در خروجی پیام "Hello Iran" را چاپ کند؟

برنامه Eclipse رو باز می کنیم. تصویر (۱)



تصویر (۱)

برای نوشتن برنامه جاوا ابتدا باید پروژه جدید بسازیم. (File>New>Java Project).

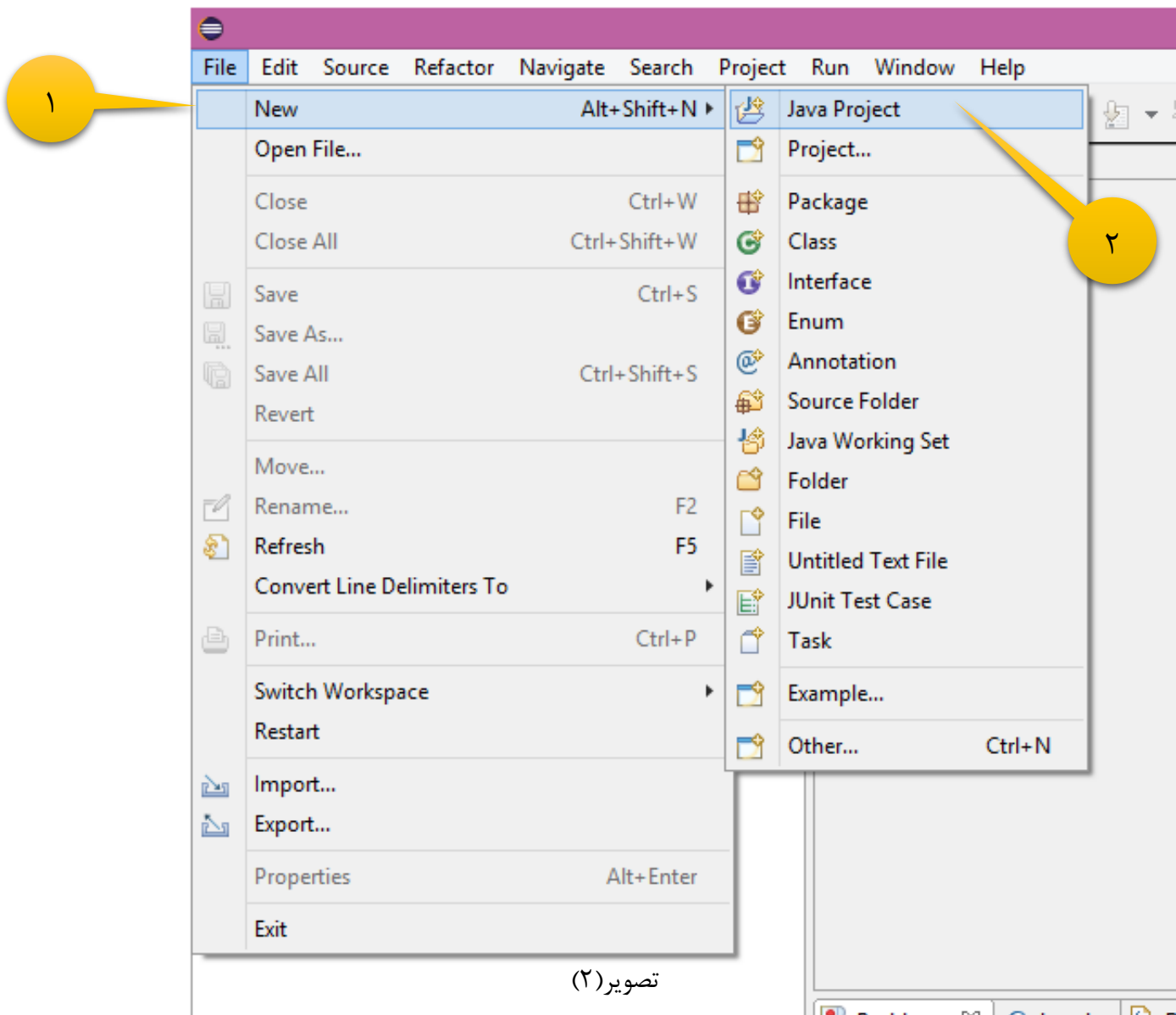
برای ساخت پروژه جدید در Eclipse : تصویر (۲)

۱. انتخاب منوی فایل (File)

۲. انتخاب گزینه New

۳. انتخاب گزینه Java project

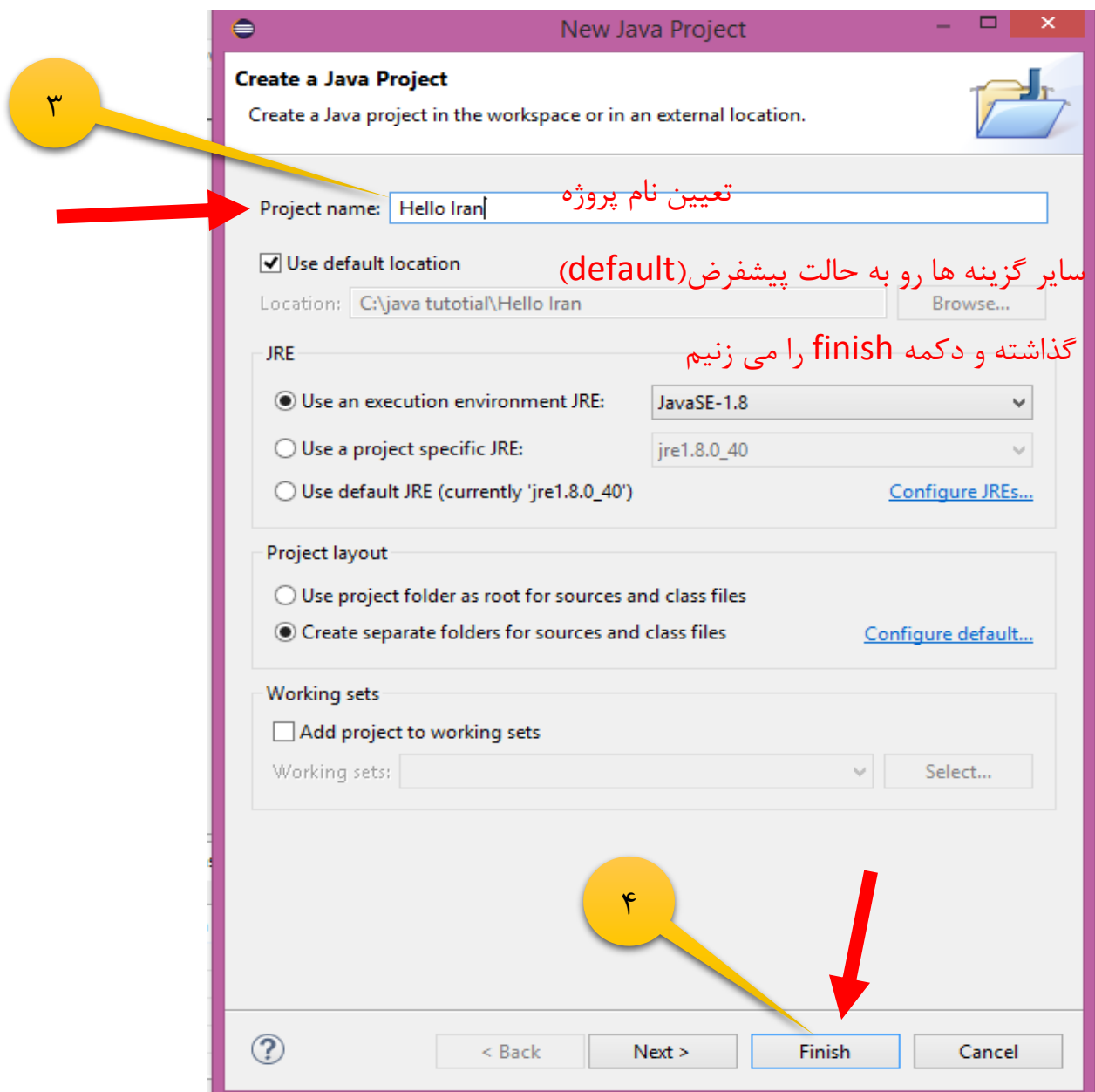
- نیاز نیست هر بار برای نوشتن برنامه پروژه جدید تشکیل بدیم می توانیم با ساخت ی new project هر چندتا برنامه خواستیم بنویسیم.



نام گذاری پرای پروژه:

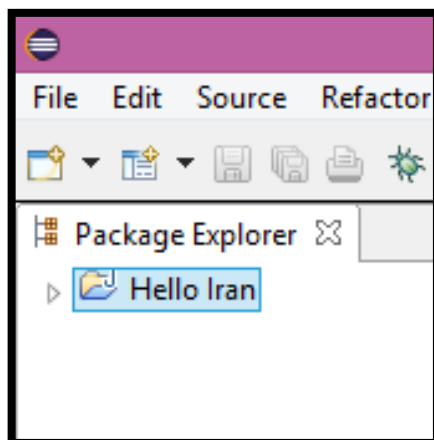
در بخش **New Java Project** کارهای زیر رو انجام میدیم:

تعیین نام مورد نظرمون در قسمت **Project Name** که در این جا نام "Hello Iran" رو انتخاب کردیم. (نام پروژه میتونه با حرف بزرگ یا کوچک شروع شود) بقیه گزینه ها رو به حالت پیشفرض (default) گذاشته و دکمه **Finish** رو میزنیم. تصویر (۳)



تصویر (۳)

در بخش Package Explorer فولدر پروژه مون که در اینجا نامش "Hello Iran" است ایجاد میشود. تصویر (۴)



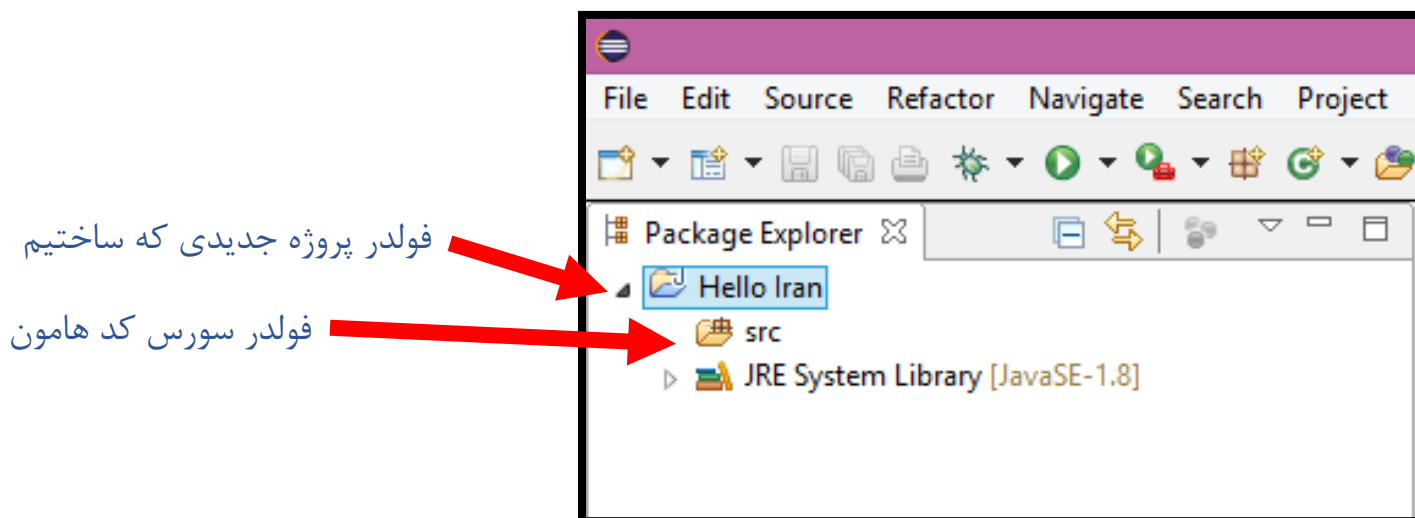
تصویر (۴)

خب تا اینجا اولین گام در جهت برنامه نویسی جاوا در Eclipse برداشتیم.

در بخش Package Explorer: تصویر (۵)

فولدر پروژه مون قرار دارد.

فولدر SRC که فولدر سورس کد هامون هست (کدهایی که می نویسیم درون این فولدر می باشد) درون فولدر پروژه مون قرار دارد.



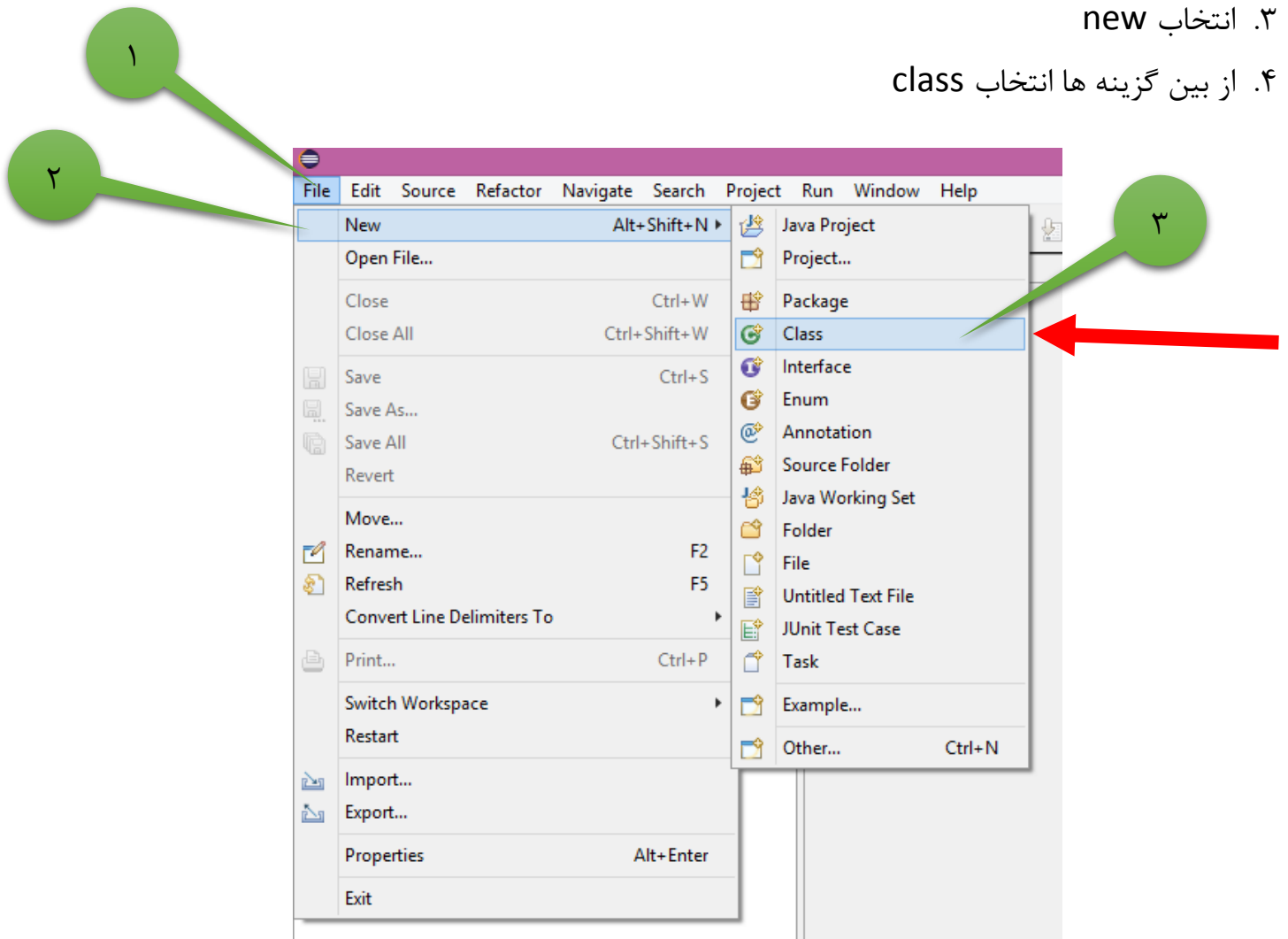
تصویر (۵)

همان طور که می دونید در نوت پد برای نوشتن برنامه جاوا نیاز به تعریف کلاس داشتیم در اینجا هم همین طور هست باید ابتدا یک کلاس بسازیم و درون کلاسمون شروع به کد زدن کنیم.

Eclipse ساختن کلاس رو برای ما راحت کرده!!!! برای ایجاد کلاس در Eclipse بصورت یکی از روش های زیر عمل میکنیم:

روش اول (ایجاد کلاس): تصویر (۶)

۱. فولدر پروژه مورد نظرمون رو انتخاب کرده
۲. رفتن به منو file
۳. انتخاب new
۴. از بین گزینه ها انتخاب class



تصویر (۶)

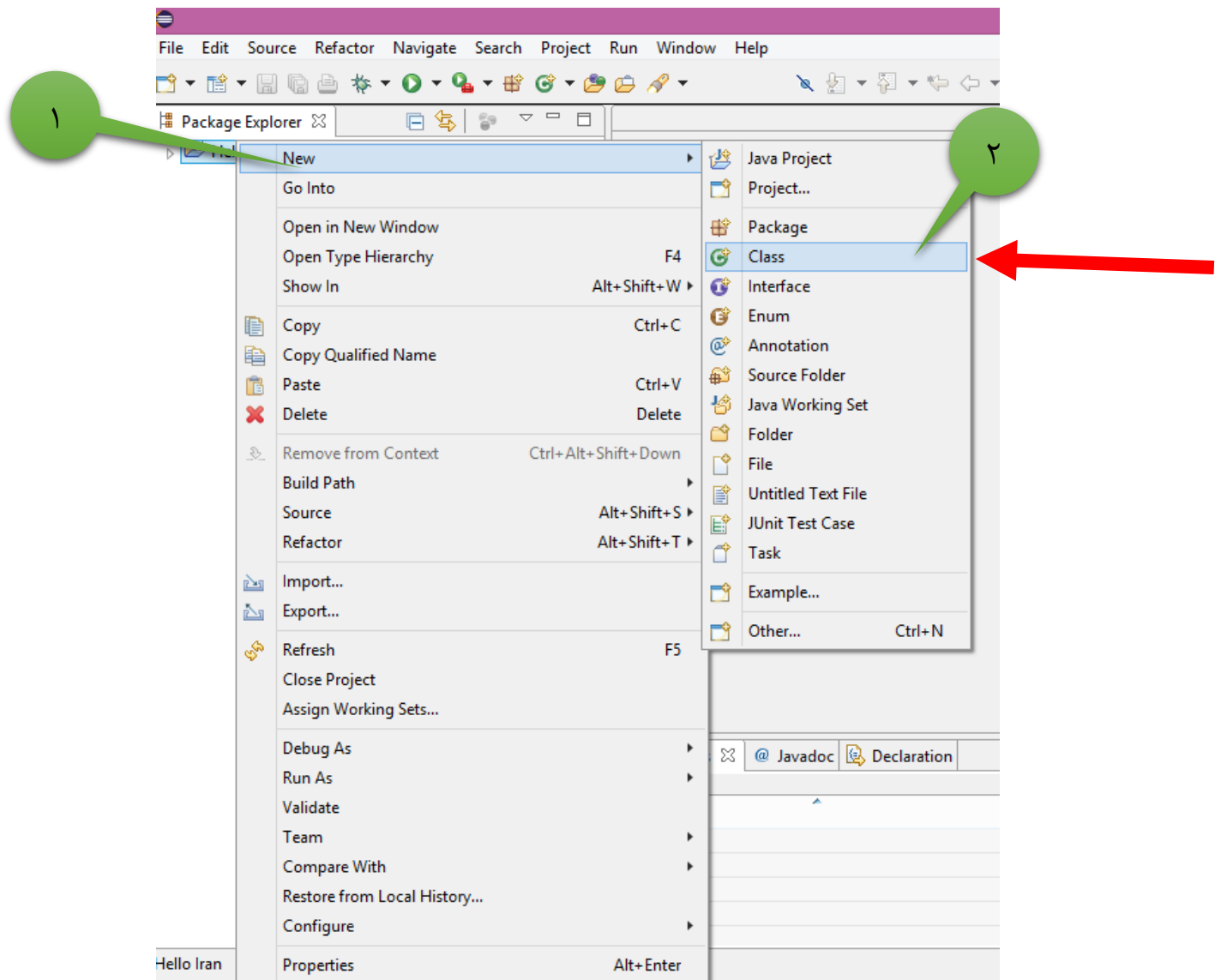
روش دوم (ایجاد کلاس): تصویر (۷)

۱. فولدر پروژه مورد نظرمون رو انتخاب کرده

۲. کلیک سمت راست

۳. انتخاب new

۴. از بین گزینه ها انتخاب class

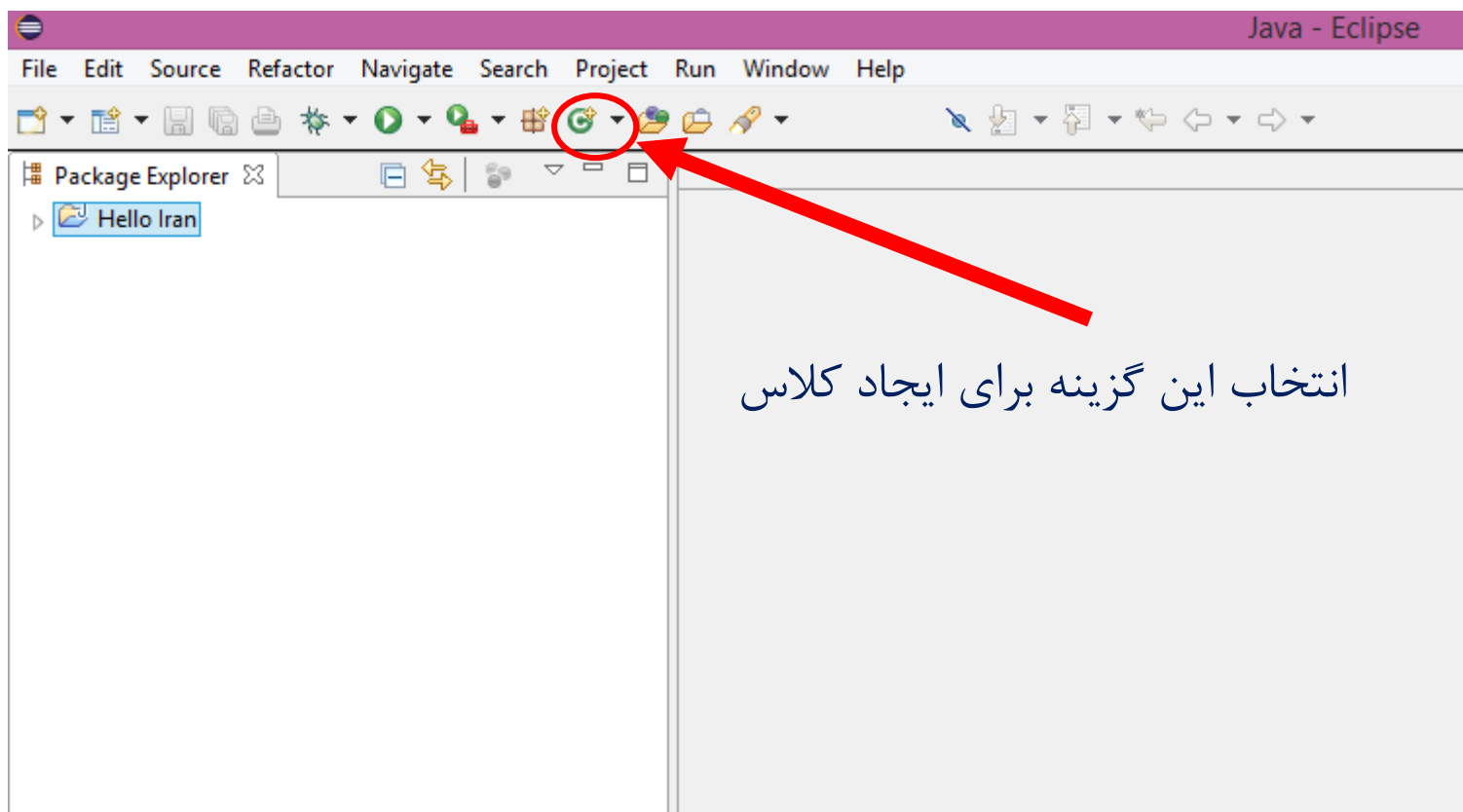


تصویر (۷)

روش سوم (سریع ترین روش ایجاد کلاس): تصویر (۸)

۱. فولدر پروژه مورد نظرمون رو انتخاب کرده

۲. از نوار Toolbar دکمه سبز رنگ شبیه  "new java class" رو انتخاب میکنیم.



تصویر (۸)

بعد از انتخاب کلاس (حالا با هر کدام از روشی که دوست دارید) فرمی تحت عنوان "New Java Class" نمایش داده میشود. که در این فرم به دلخواه برای کلاس مان نامی رو انتخاب میکنیم که در اینجا نام "FirstProgram" رو انتخاب میکنیم.

در این فرم بخش های دیگه ای نظیر package وجود دارد که کلاس هامون درونش قرار میگیرد.

نقش Package :

ما برای نظم بخشیدن و مرتب سازی برنامه مون، کلاس هامون رو پکیج بندی میکنیم .

مثال:فرض کنید ۱۰ کلاس داریم که از این ده تا ۶ تا کلاسش در مورد یک مسئله هست و ۴ تای دیگه درمورد مسئله ای دیگر برای نظم بخشیدن به کلاسامون میتوانیم ۶ کلاس هم موضوع رو در یک پکیج و ۴ تای دیگه در پکیج دیگر قرار می دهیم.

- این رو بدونید که اگر ۱۰ کلاس در یک پکیج قرار بگیرند هیچ مشکلی پیش نمی آید و این کار تنها برای نظم بخشیدن به کلاس هامون هست.

می توانیم نامی به دلخواه برای package مون انتخاب کنیم که باید با حرف کوچک شروع شود.اگر نامی برایش انتخاب نکنیم، Eclipse کلاس ما را درون پکیج پیشفرض(Default package)قرار می دهد.

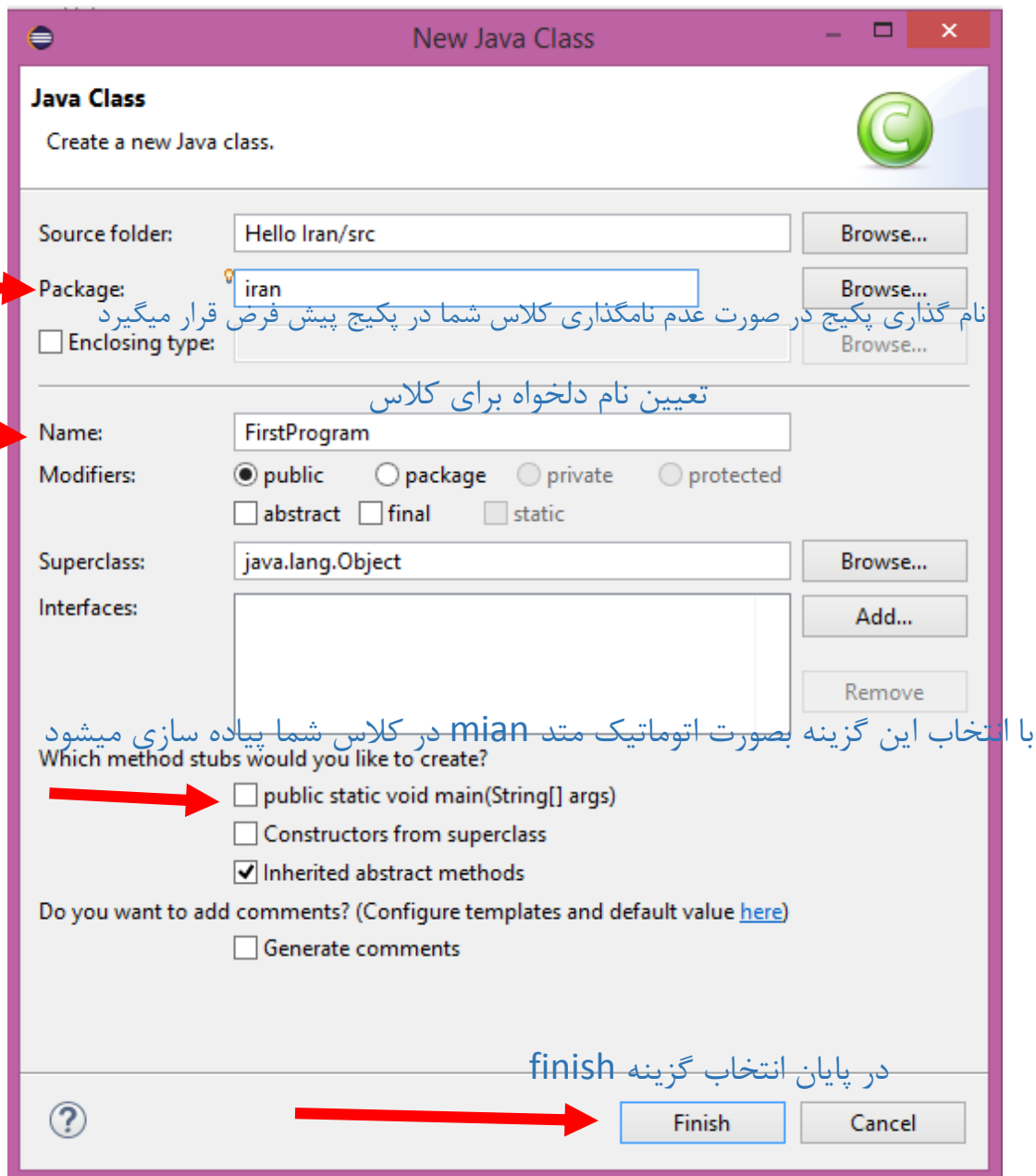
اسم پکیج مون رو "iran" می گذاریم.

در پایین فرم دستور آشنایی دیده میشه!!!!درسته متد main!!!!در این قسمت با تیک زدن گزینه

`public static void main(String[] args)` برنامه Eclipse بصورت اتوماتیک این دستور رو در کلاس شما پیاده سازی میکند و دیگر نیاز به نوشتنش نیستید!!!!می بینید Eclipse داره تمام تلاشش رو میکنه که شما راحت بتونین برید برنامه خودتون رو بنویسید حالا جلو میریم و میبینید که چقدر امکانات دیگه در اختیار شما میزاره حتی خطایابی برنامه تون و پیشنهاد دادن متدها و متغیر ها و....

پس ما تیک این گزینه رو انتخاب میکنیم تا متد مین برامون پیاده سازی شود.

در پایان گزینه Finish رو انتخاب میکنیم.تصویر (۹)



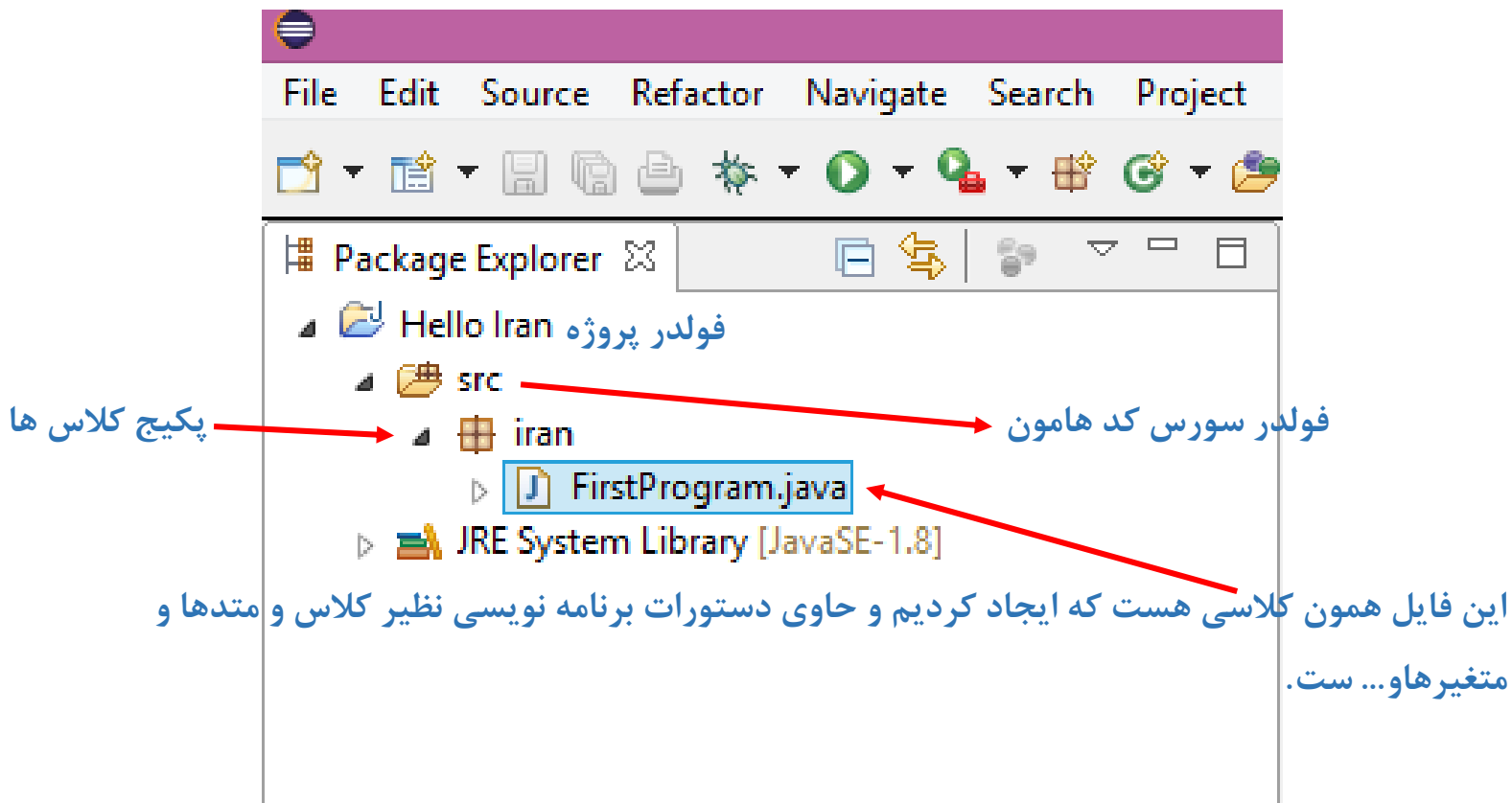
تصویر (۹)

خب بعد از نامگذاری `class` و زدن دکمه `Finish` کلاس ما ایجاد میشود.

کارایی که تا اینجا انجام دادیم: تصویر (۱۰)

۱. ایجاد پروژه جدید به نام `Hello Iran`

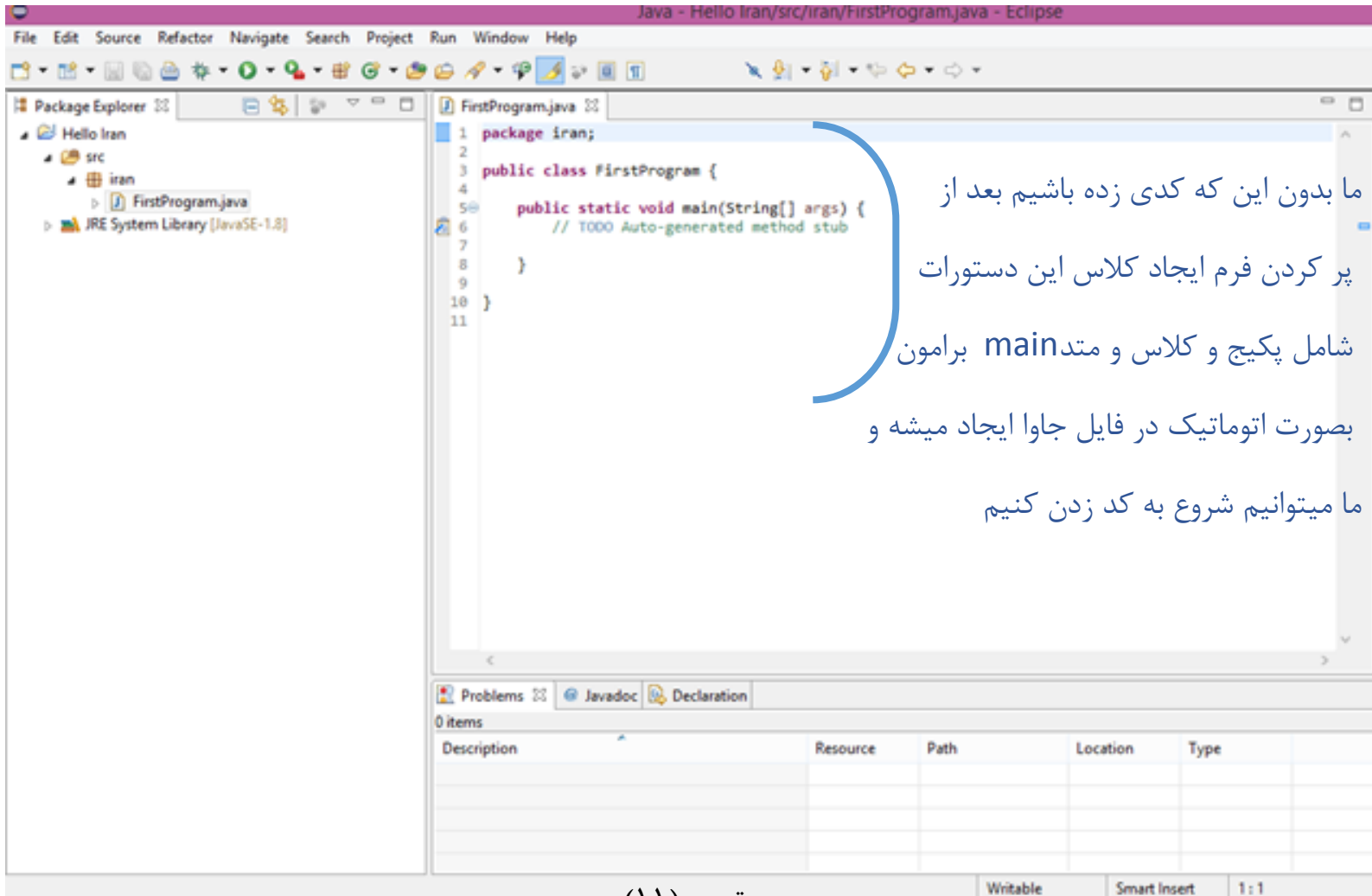
۲. ایجاد یک کلاس به نام `"FirstProgram"` درون پکیجی به نام `"iran"`



تصویر (۱۰)

خب تا اینجا چطور بود؟! سریع تر از Notepad نبود؟! جالب بدونید شما بعد از ایجاد پروژه تا زمانی که نیاز پیدا نکردید نمیخواه دوباره پروژه جدید بسازید میتوانید برای ایجاد کلاس جدید اگر هم موضوع با کلاس قبلی بود درون پکیجی که از قبل وجود داره بزارید اگه نه ی پکیج جدید میسازید و کلاس جدید رو درون این پکیج قرار میدید. پس می بینید که با چ سرعتی فقط با ی کلیک و تعیین نام میشه کلاس ساخت و درونش شروع به کد زدن کرد.

- برای دیدن سورس کد (فایلی که کد برنامه نویسمون درونش هست و داخلش کد میزنیم) روی کلاسمون که فرمتش جاوا (.java) هست دابل کلیک میکنیم تا فایل باز شود و کدهامون رو ببینیم. تصویر (۱۱)
- ❖ درون فایل **FirstProgram.java** (فایلی که بعد از ساختن و تعریف کردن کلاس مون ایجاد شد) ما شروع به کد زدن میکنیم و ما چون از قبل نام کلاس و نام پکیج مشخص کردیم و متد **main** رو تیک زدیم که ایجاد شود (همه این کارها زمان ایجاد کلاس انجام دادیم) برنامه Eclipse بصورت خودکار این ها رو درون فایل **FirstProgram.java** برای ما پیاده سازی کرده است و تنها نیاز هست ما درون بدنه کلاس مون شروع به کد زدن کنیم. تصویر (۱۱)



تصویر (۱۱)

❖ دستوراتی که با ایجاد کلاس و پکیج و تیک زدن متد main پیاده سازی میشود بصورت زیر است:

```

1. package iran;
2.
3. public class FirstProgram {
4.
5.     public static void main(String[] args) {
6.     // TODO Auto-generated method stub
7.
8.     }
9.
10.     }

```

- تا اینجا ما هیچ کدی نزدیم و تنها با پر کردن فرم ایجاد کلاس و تیک زدن سری گزینه دستورات بالا بصورت خودکار برای ما پیاده سازی شدند که توضیح هر مورد بصورت زیر است:

دستور خط ۱:

```
package iran;
```

خب ما از قبل در فرم ایجاد کلاس (New Java class) برای پکیج کلاس مون نام "iran" رو انتخاب کردیم.

این دستور میگه کلاس ما در پکیج "iran" قرار دارد.

نام پکیج هر کلاس در ابتدا و بالای دستورات و کلاس برنامون قرار میگیرد.

دستور خط ۳:

```
public class FirstProgram {  
  
}
```

- ما در فرم ایجاد کلاس (New Java class) نام کلاس مون رو "FirstProgram" گذاشتیم . بعد از زدن دکمه **Finish**

بصورت خودکار Eclipse برامون در فایل همنام با کلاس که فرمتش java هست پیاده سازی میکند و ما نیاز نداریم که بصورت دستی کلاس رو تعریف کنیم.

- بین دو آکولاد باز و بسته "{}" دستورات برنامه شامل متدها، متغیرها و ... که در جلسات بعد بیشتر بهش می پردازیم قرار میگیرد.
- نام کلاس همان طور که مشاهده میکنید باید با حرف بزرگ شروع شود.
- قبل از کلمه کلیدی "class" از کلمه کلیدی "public" که معنای عمومی هست استفاده شده است که در جلسات آینده بیشتر در مورد کلمات کلیدی جاوا آشنا می شویم.

دستور خط ۵:

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

- متد `main` رو مشاهده میکنید که درون بدنه کلاس قرار گرفته است. همان طور که در جلسات قبل گفتم هر کلاس برای اجرا باید این متد داخلش پیاده سازی شود.
- هر کلاس تنها می تواند یک متد `main` داشته باشد.
- وقتی ما برنامه مون رو اجرا میکنیم کامپایلر خط به خط دستورات درون متد `main` رو اجرا میکند یعنی هر دستور از کلاس که بخواد اجرا شود با استفاده از این متد (`main`) اجرا میشود.

درون متد `main` عبارت `// TODO Auto-generated method stub` مشاهده میکنید. این عبارت میگه که متد `main` شما به صورت خودکار ایجاد شده است (چون ما تیک ایجاد شدن متد `main` رو در فرم ایجاد کلاس زدیم این عبارت نمایش داده شده است) در کنار این عبارت علامت `/**` مشاهده میکنید، از دو علامت اسلش `/**` برای کامنت گذاری (`comment`) استفاده می شود. کاربرد `comment`:

وقتی بخوایم بخشی از دستور پروژه مون رو توضیح بدیم که بقیه هم کد رو بفهمند یا برای خودمون یا وقتی یک پروژه بصورت گروهی انجام میشود در کنار هر دستور می توانیم کامنت گذاری کنیم که ابتدای توضیح دو علامت اسلش `/**` بکار می بریم. دو نوع روش `comment` گذاری داریم:

۱. ابتدای هر توضیح دو علامت اسلش `/**` می آوریم.

مثال:

```
// TODO Auto-generated method stub
```

۲. توضیح خودمان در مورد کد رو در دو بلوک زیر قرار می دهیم:

```
/*
    بینش توضیح مورد نظر
*/
```

مثال:

```

/*
    TODO Auto-generated method stub
*/

```

• یکی دیگر از کاربرد `comment` گذاری بصورت زیر است:

وقتی بخشی از کد برنامه ای که نوشتیم نیازش نداریم و احتمال داره در آینده بهش نیاز پیدا کنیم یا قصد داریم بصورت موقت ازش استفاده نکنیم اون بخش از کد رو می توانیم `comment` گذاریش کنیم و با این کار وقتی کامپایلر خط به خط برنامه رو اجرا کرد و به اون قسمت `comment` گذاری شده رسید توجهی بهش نمیکنه و بدون این که کد رو اجرا کنه سراغ خط بعدی می رود.

مثال: قصد داریم متد `main` موجود در کلاس `"FirstProgram"` رو با `comment` گذاری کردن غیر قابل اجرا کنیم:
روش اول `comment` گذاری کردن: از بلوک `*/` استفاده میکنیم. در محدوده بلوک هر دستوری باشه غیر قابل اجرا میشود.

```

package iran;

public class FirstProgram {

    /*   public static void main(String[] args) {

        TODO Auto-generated method stub

    }
    */
}

```

روش دوم `comment` گذاری کردن: برای هر خط دستور تک به تک از علامت `"//"` استفاده میکنیم.

```

package iran;

public class FirstProgram {

    //   public static void main(String[] args) {

        // TODO Auto-generated method stub

        //}

    }
}

```

با کامنت گذاری کردن متد `main`، کامپایلر وقتی به این متد میرسد ازش رد میشود و اجراش نمیکند.

• اینقدر از واژه کامپایلر (compiler) استفاده کردم اما ی توضیحی هم در موردش ندادم!!! تنها این رو بدونید که کامپایلر کد برنامه نویسی که زدید رو میخواند و به زبان ماشین تبدیل میکند همان طور که میدونید کامپیوتر زبون آدمی زاد حالیش که نیست!!!! فقط زبون ماشین رو میفهمه!!! جناب کامپایلر هم به ما لطف میکنه و زبان برنامه نویسی ادم رو به زبان ماشین تبدیل میکنه که برای کامپیوتر قابل فهم باشه!!!! من به بصورت خودمونی گفتم خواستید بیشتر بدونید ی سرچ کوچیک تو نت کنید ☺

خب حالا که با چگونگی ساختن کلاس ، پکیج ، متد main و جزیات دیگر آشنا شدیم میریم سراغ اصل مسئله این که در خروجی پیام "Hello Iran" چاپ کنیم .

همان طور که گفتیم برای اجرا دستورات برنامه از متد main موجود در کلاس استفاده می کنیم پس دستورات خودمون رو برای اجرا شدن در این متد پیاده سازی میکنیم. برای این کار بصورت زیر عمل میکنیم:

یادآوری :

در جاوا برای چاپ متن و پیام و مقدار در خروجی از دستور زیر استفاده میکنیم:

```
System.out.println();
```

برای چاپ متن با پیام مورد نظرمون رو درون دو نقطه ویرگول "" بصورت زیر قرار بدیم:

```
System.out.println("متن مورد نظر");
```

که در اینجا قصد داریم متن "Hello Iran" رو چاپ کنیم پس بصورت زیر عمل میکنیم:

```
System.out.println("Hello Iran");
```

تفاوت System.out.println() و System.out.print() :

System.out.println() وقتی یک پیام چاپ میکند بعدش به سطر بعد می رود.

وقتی از عبارت "\n" استفاده کنیم باعث میشود وقتی یک متن چاپ شود بعش به سطر بعد برود که System.out.println() این کار رو انجام می دهد.

System.out.print() وقتی یک پیام چاپ میکند به سطر بعد نمی رود.

اگر میخواهید `System.out.print()` شبیه `System.out.println()` عمل کند کافیس که در ورودی `System.out.print()` عبارت `"\n"` قرار بدیم با این کار بعد از چاپ به سطر بعد می رود به شکل زیر:

```
System.out.print("\n")
```

برای سریع تایپ کردن دستور `System.out.println()` کافیسست بصورت زیر عمل کنیم:
تایپ عبارت `Syso` که `S` با حرف بزرگ شروع میشود.

بعد زدن دکمه ترکیبی `ctrl+space`

```
Syso + ctrl+space
```

با دستور چاپ در خروجی آشنا شدیم حالا میریم به سراغ نوشتن ادامه برنامه مون که چاپ پیام "Hello Iran" در خروجی است برای این کار کافی است در بدنه متد `main` دستور `System.out.println();` استفاده میکنیم. درون پرانتز (ورودیش) متن `Hello Iran` که بین دو "" قرار گرفته رو قرار می دهیم. دستور برنامه ما تا این جا بصورت زیر است: تصویر (۱۲)

```
package iran;

public class FirstProgram {

    public static void main(String[] args) {

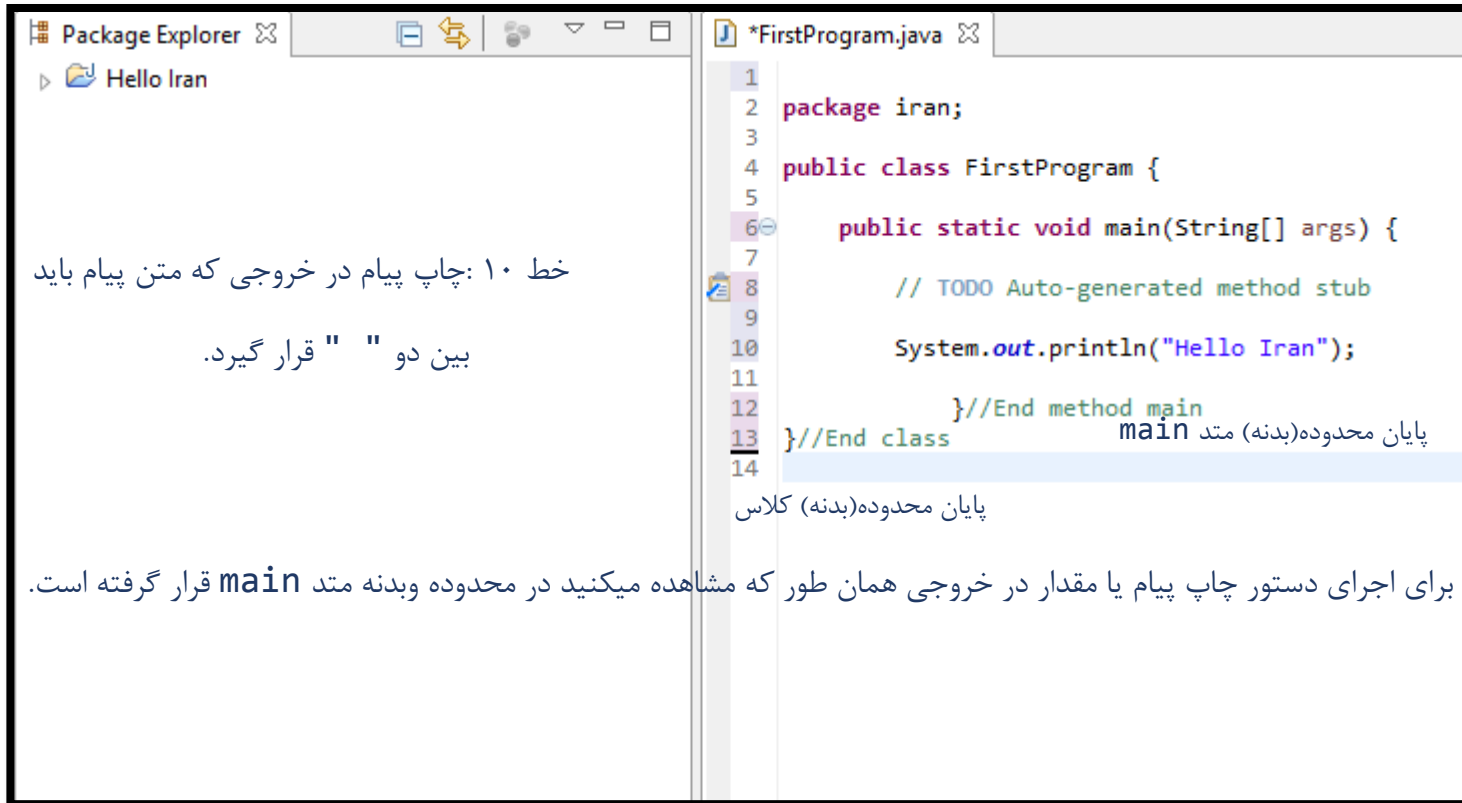
        // TODO Auto-generated method stub

        System.out.println("Hello Iran");

    } //End method main


} //End class
```

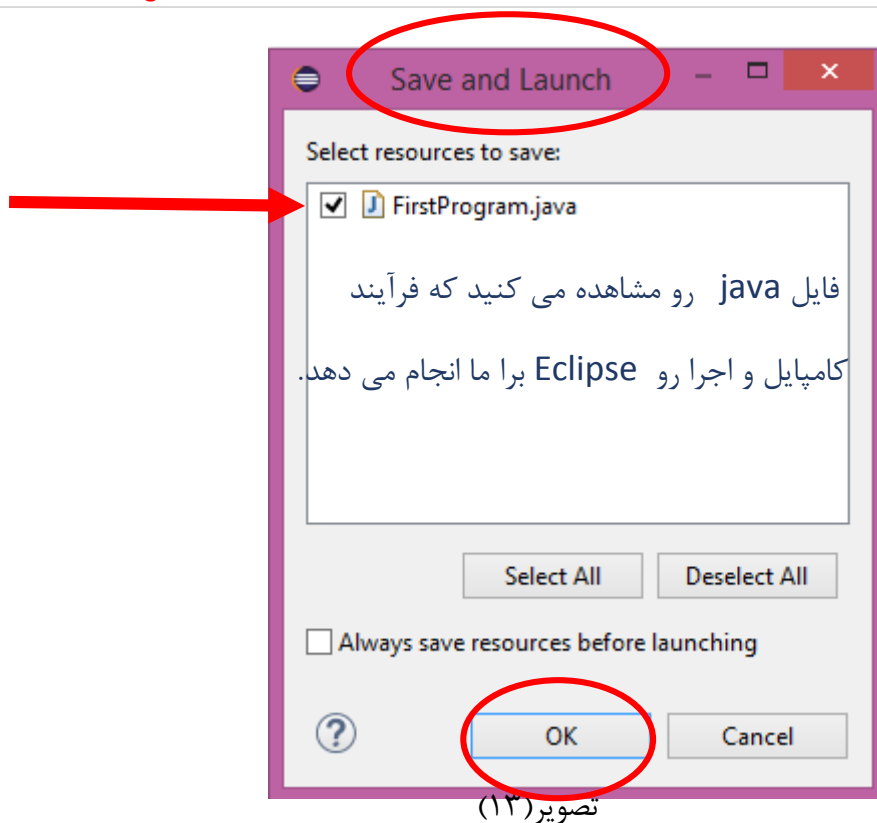
• می توانید دستور بالا رو کپی و مستقیم در ایکلیپس پیست کنید برای خودتون تست کنید.



تصویر(۱۲)

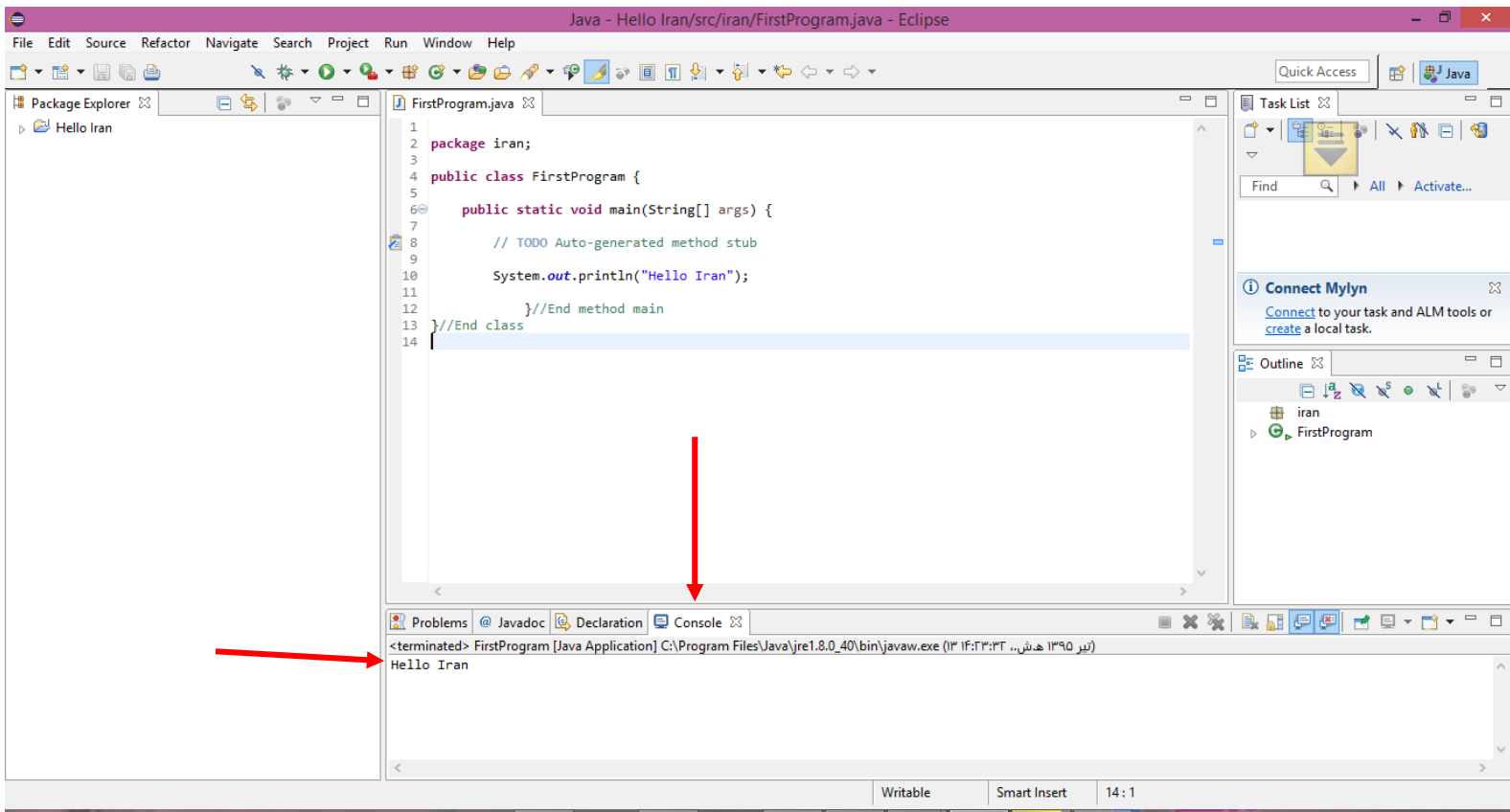
بعد از کد زدن برای اجرای برنامه نیاز هست برنامه رو به اصطلاح Run کنیم برای Run کردن برنامه بصورت زیر در Eclipse عمل میکنیم:

- ۱) در نوار Toolbar انتخاب گزینه Run 
- ۲) بعدش فرمی تحت عنوان Save and launch باز میشود که باز زدن دکمه Ok همزمان برنامه رو ذخیره و اجرا میکند. تصویر(۱۳)



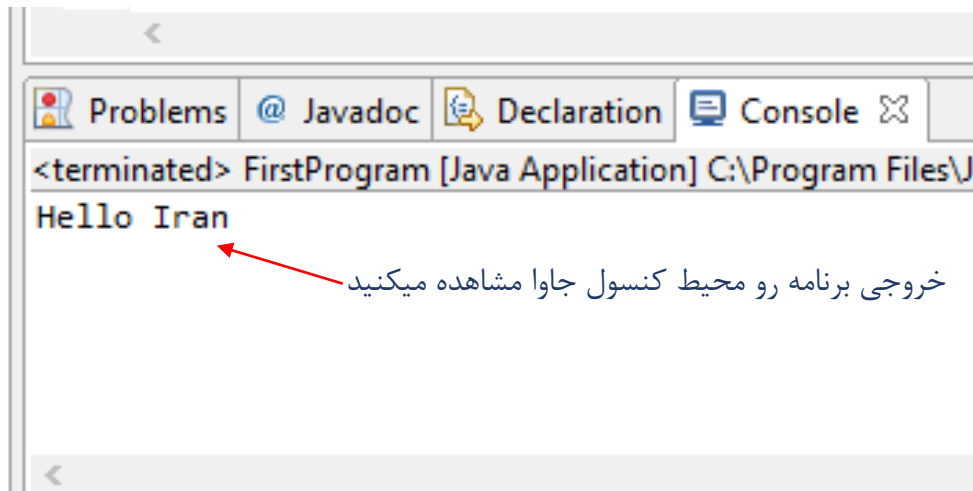
خب دیگه خبری از فرایند سخت و خشک و وقت گیر کامپایل و اجرا برنامه های جاوا نیست!!!! با ی کلیک Eclipse برای ما برنامه رو کامپایل و اجرا میکند.

در پایان خروجی در محیط کنسول جاوا نمایش داده میشود. تصویر (۱۴) و (۱۵)



تصویر (۱۴)

- از این به بعد هر کدی که میزنیم خروجی برنامه در محیط کنسول جاوا که سفید رنگ است نمایش داده میشود. تا این که در آینده به مباحث گرافیک میرسیم و خروجی خود رو بصورت گرافیکی و پنجره ای رو فریم و... خواهید دید.



تصویر (۱۵)

خب تا اینجا با طریقه ساختن کلاس، پکیج بندی کلاس، استفاده از متد main و اجرا اولین برنامه مون آشنا شدیم

شاید در نگاه اول بگید اینم شد برنامه نویسی کد بزنیم که ی خروجی چاپ بشه؟!!!! این همه وقت بزاریم ی برنامه بنویسیم که هیچ کاربردی نداره؟!!!! صبر داشته باشید ما ابتدا سعی داریم که گام به گام و جز به جز جلو برویم که با مفاهیم پایه برای نوشتن برنامه کاربردی و بازی و... آشنا شویم، مثل ورزشکاری که روز اول میره باشگاه باید از وزنه سبک شروع کنه حتی طریقه صحیح وزنه زدن!!!! اما هم باید در برنامه نویسی صبر و حوصله داشته باشیم و با علاقه و انگیزه و اراده خواهید دید که یکی از بهترین برنامه نویس ها شدید!!!! پس با ما همراه شوید 😊

نظرات و پیشنهادات و رفع اشکال و راهنمایی ایمیل یا در کانال تلگرام بخش ادمین پیام بدید

پیروز و موفق باشید

سایت آموزش زبان جاوا به زبان ساده، آسان و شیرین!!!

www.JAVAPRO.ir

آموزش جاوا SE را با تجربه شخصی و به زبان خودمونی یاد بگیرید!!!!

بازدید از کانال

بازدید از سایت

هر روز مفاهیم و مثال های جدید به سایت اضافه می شود برای اطلاع از مطالب جدید روی سایت عضو کانال شوید.