به نام خدا

آموزش چهل و چهارم

اهداف آموزشی این قسمت عبارتند از:

۱. معرفی کلاس Math در زبان جاوا
۲. معرفی متدهای مرتبط با کلاس Math

در این آموزش و آموزش بعد قصد داریم تا یکی از پرکاربردترین کلاس های ساخته شده در API زبان برنامه نویسی جاوا را مورد بررسی قرار دهیم. به طور خلاصه می توان گفت که اگر بخواهیم از اعمال ریاضیاتی در طراحی برنامه های خود بهره مند شویم، می بایست از کلاس Math و متدهای مختلف مرتبط با این کلاس استفاده کنیم(توجه داشته باشید که حرف اول این کلاس با حرف بزرگ نوشته می شود).

در ابتدا پروژه ای تحت عنوان 44^{th} Session به معنی **"جلسه چهل و چهارم"** ایجاد کرده و کلاسی تحت عنوان MathClass به معنی **"کلاس ریاضیات"** در آن ایجاد می کنیم و پیش از کلیک کردن بر روی دکمه Finish گزینه public static void main را تیک دار می کنیم.

اکنون برای آنکه بتوانیم متدهای مختلف کلاس Math را مورد بررسی قرار داده و خروجی این متدها را مشاهده کنیم، نیاز است تا دستور System.out.println(); را در کد خود اضافه کنیم.

در این مرحله از کار، کد برنامه می بایست به صورت زیر باشد:

```
public class MathClass {
	public static void main(String[] args) {
		System.out.println();

	}
}
```

حال چنانچه هر چیزی را داخل متد println قرار دهیم در پنجره Console به نمایش در خواهد آمد. در این آموزش پیش از آشنایی با کلاس Math به دو رویکرد در Import کردن کلاس های مختلف در پروژه خود خواهیم پرداخت سپس با متدهای abs و ceil و floor آشنا خواهیم شد.

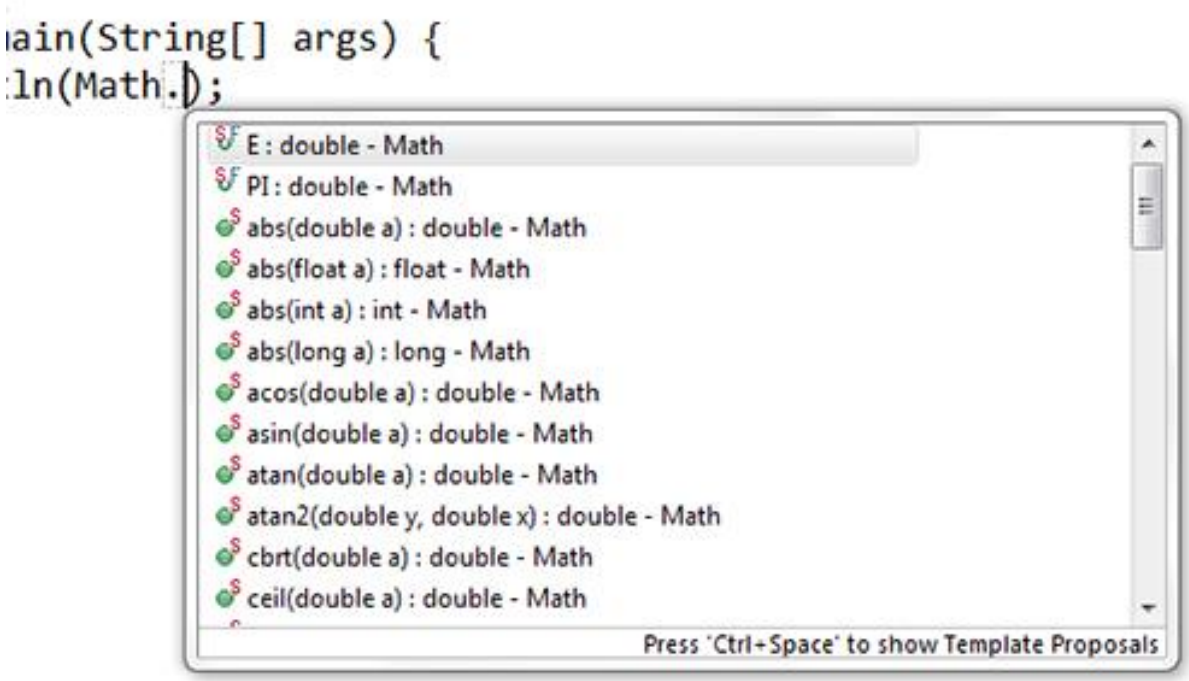
به طور خلاصه می توان گفت که کلیه کلاس های API زبان جاوا به دو صورت Implicit و Explicit به ترتیب به معانی **"نا آشکار"** و **"آشکار"** به برنامه ما Import می شوند. به طور مثال کلاسی همچون کلاس Scanner به صورت Explicit یا واضح و آشکار به برنامه Import می شود. به عبارت دیگر پس از به کار گیری از این کلاس در برنامه خود، در اولین خط از کد برنامه دستور import java.util.Scanner; را خواهیم دید. از سوی دیگر یکسری از کلاس ها مثل کلاس String و یا کلاس Math و بسیاری از دیگر کلاس های زبان جاوا هستند که به صورت Implicit یا نا آشکار به برنامه Import می شوند. به طور مثال زمانیکه ما در برنامه خود از کلاس String استفاده می کنیم، مشاهده می شود که هیچ دستور Import یی در برنامه مشاهده نمی شود و این مسئله به این خاطر است که این کلاس به صورت ضمنی یا نا آشکار در هر پروژه ای که در اکلیپس بسازیم Import خواهد شد. کلاس Math هم به همین صورت است. به عبارت دیگر برای استفاده از این کلاس نیازی به Import کردن آشکارا یا Explicit این کلاس نداریم.

اولین متدی که از کلاس Math می خواهیم مورد بررسی قرار دهیم، متد abs است که مخفف واژه absolute به معنی **"مطلق"** است. کاری که این متد انجام می دهد این است که هر عددی که داخل آن قرار گیرد خواه مثبت و خواه منفی، آن عدد را به یک عدد مثبت تبدیل می سازد.

برای روشن شدن موضوع، کد فوق را به شکل زیر تکمیل می کنیم:

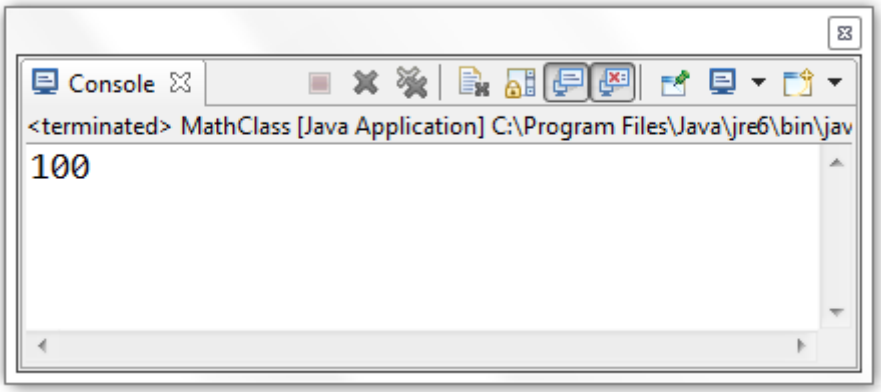
```
public class MathClass {
	public static void main(String[] args) {
		System.out.println(Math.abs(-100));
	}
}
```

همانطور که مشاهده می شود، به منظور استفاده از کلاس Math نام این کلاس را نوشته سپس یک نقطه قرار داده و پس از آن نام متدی که می خواهیم مورد استفاده قرار دهیم را می نویسیم. در واقع پس از نوشتن نام کلاس Math و قرار دادن یک نقطه، اکلیپس به صورت خودکار کلیه متدهای مرتبط با این کلاس را برای ما خواهد آورد:



چنانچه اکلیپس این کار را به صورت خودکار انجام نداد، به سادگی با فشردن هم زمان کلید های Ctrl و Space می توان به پنجره فوق دست پیدا کرد. اکنون از میان کلیه متدها، متد abs را می خواهیم مورد استفاده قرار دهیم. برای این منظور یا می توان حروف abs را به صورت دستی پس از نقطه نوشت و یا همانطور که در پنجره فوق مشاهده می شود می توان روی یکی از گزینه های مرتبط با abs کلیک کرد(از آنجا که می خواهیم در این مثال یک عدد صحیح را مورد بررسی قرار دهیم، از این رو روی گزینه abs(int a) : int – Math کلیک می نماییم).

سپس در پرانتزهای داخل این متد یک عدد مثل منفی صد وارد می کنیم. اکنون پس از اجرای برنامه، خروجی زیر مشاهده خواهد شد:



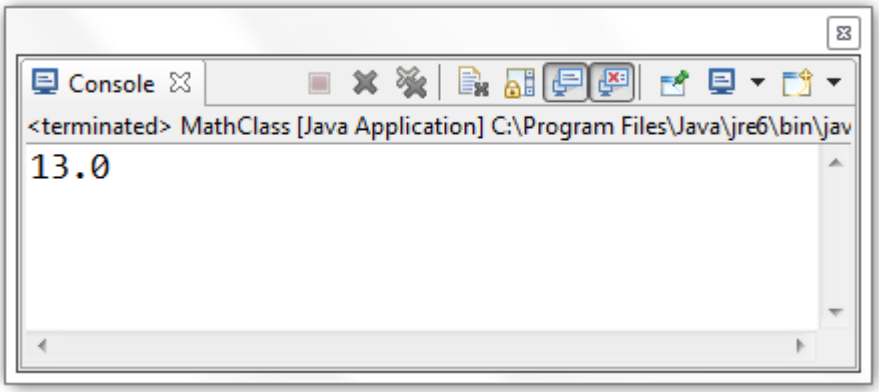
همانطور که ملاحظه می شود با استفاده از این متد، عدد منفی صد به عدد 100 تبدیل شده است.

متد دیگر که از کلاس Math می خواهیم مورد بررسی قرار دهیم، متد ceil() است. در واقع کلید واژه ceil مخفف واژه ceiling به معنی **"سقف اتاق"** است. داده ای که این متد می گیرد از نوع عدد اعشاری است و کاری که این متد انجام می دهد این است اعداد اعشاری را رو به بالا رند می سازد. مثلا اگر عدد 12.3 را داخل این متد قرار دهیم، پس از اجرای برنامه این عدد به عدد 13 تغییر پیدا خواهد کرد:

برای روشن شدن مطلب کد فوق را به شکل زیر تکمیل می کنیم:

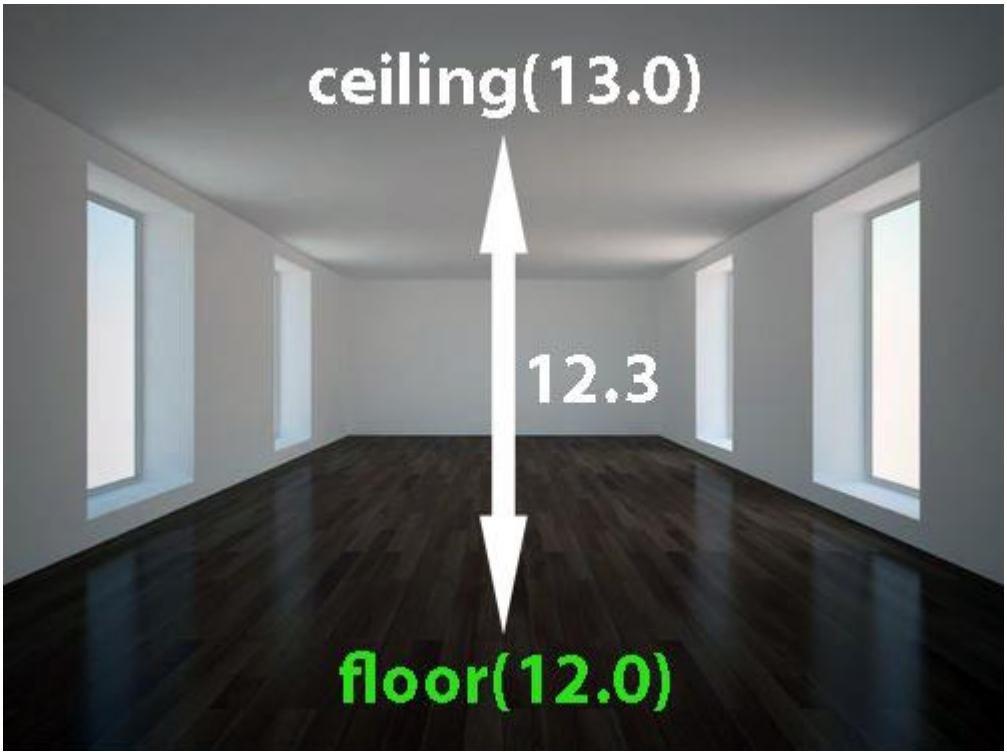
```
public class MathClass {
	public static void main(String[] args) {
		System.out.println(Math.ceil(12.3));
	}
}
```

همانطور که در کد فوق ملاحظه می شود، پس از کلاس Math متد ceil قرار گرفته و داخل پرانتزهای این متد عدد 12.3 را نوشته ایم. حال برنامه را اجرا می کنیم:



همانطور که در اجرای فوق ملاحظه می شود، عدد 12.3 به عدد 13 تغییر یافته است.

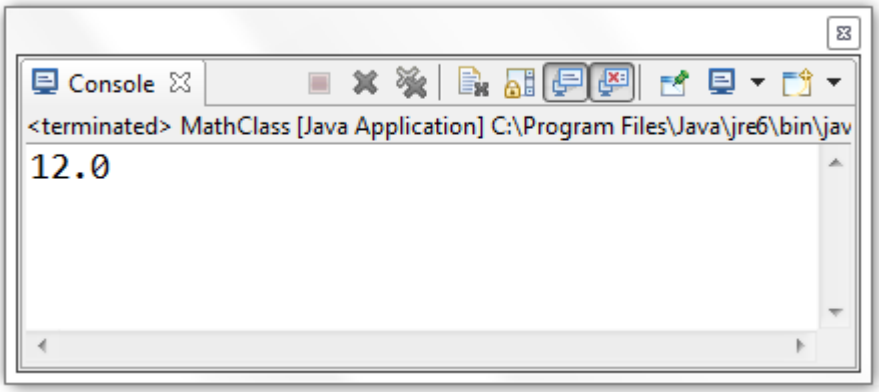
متد متضاد ceil() متدی است تحت عنوان floor() به معنی **"کف اتاق"** است که کار این متد این است که اعداد اعشاری را رو به پایین رند می کند:



برای روشن شدن مطلب کد پیشین را به صورت زیر بازنویسی می کنیم:

```
public class MathClass {
	public static void main(String[] args) {
		System.out.println(Math.floor(12.3));
	}
}
```

حال کد فوق را اجرا می کنیم:



همانطور که مشاهده می شود متد floor عدد 12.3 را رو با پایین رند کرده و آن را به عدد 12 کاهش داده است.

پس از مطالعه این آموزش انتظار می رود بتوانیم به سؤالات زیر پاسخ بدهیم:

۱. انواع روش های Import در زبان جاوا کدامند؟
۲. متدهای مرتبط با کلاس Math کدامند؟
۳. تفاوت استفاده از متد ceil و floor در چیست؟

در قسمت آینده با دیگر متدهای پرکاربرد کلاس Math که عبارتند از max و min و pow و sqrt و round آشنا خواهیم شد.