نحو جاوا(Java Syntax)

مقدمه

در فصل قبل، یک فایل جاوا به نام Main.java ساختیم و با استفاده از کد زیر، اولین برنامه‌ی خود را اجرا کردیم:

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

این برنامه جمله‌ی «Hello World» را روی صفحه چاپ می‌کند. ممکن است این کد برای کسی که تازه شروع کرده کمی پیچیده به نظر برسد. پس بیایید قدم‌به‌قدم آن را بررسی کنیم.

---

۱) کلاس (Class) در جاوا چیست؟

در جاوا، هر برنامه باید حداقل شامل یک **کلاس (class)** باشد. کلاس، چیزی شبیه یک جعبه است که همه‌ی کدهای شما در آن قرار می‌گیرند.

```
public class Main {
  // all code goes here
}
```

- کلمه‌ی class برای تعریف یک کلاس استفاده می‌شود.
- نام کلاس در اینجا Main است.
- طبق استاندارد جاوا، نام کلاس باید با **حرف بزرگ** شروع شود (مثل Main, Student, MyProgram)
- جاوا به **بزرگی و کوچکی حروف حساس است** .(case-sensitive) یعنی Main با main یا MAIN کاملاً متفاوت است.

---

۲) نام فایل جاوا

- نام فایل جاوا باید **دقیقاً** با نام کلاس عمومی (public class) یکسان باشد.

- مثلاً اگر کلاس شما Main نام دارد، فایل باید Main.java نام‌گذاری شود.
- اگر نام کلاس و نام فایل یکی نباشند، برنامه اجرا نخواهد شد.

---

۳) متد main()

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

این خط از کد نقطه‌ی شروع اجرای برنامه است. هر برنامه جاوا از اینجا شروع به اجرا می‌کند.

اجزای این خط را فعلاً لازم نیست به‌طور کامل بفهمید. فقط بدانید:

- main نام یک **متد** است (متد = تابعی که کاری انجام می‌دهد).
- هر کدی که داخل این متد بنویسید اجرا خواهد شد.
- علامت [] نشان‌دهنده‌ی آرایه است، و args یعنی ورودی‌هایی که می‌توان به برنامه داد (در سطح پیشرفته‌تر به آن خواهیم پرداخت).

چاپ خروجی با println

برای چاپ متن روی صفحه از System.out.println() استفاده می‌کنیم:

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

- System یک کلاس داخلی جاوا است.
- out عضوی از این کلاس است که به خروجی (مثل صفحه‌نمایش) اشاره می‌کند.
- println() مخفف **print line** است، یعنی چاپ کردن و رفتن به خط بعد.
- برای چاپ بدون رفتن به خط بعد می‌توان از System.out.print() استفاده کرد.

مثال:

```
System.out.print("Hello ");
System.out.print("World");
```

خروجی:

Hello World

نقطه‌ویرگول(Semicolon)

- هر دستور در جاوا باید با ; تمام شود.
- این نشانه، پایان یک دستور را به جاوا اعلام می‌کند.

System.out.println("Hello"); // ← نقطه‌ویرگول در پایان خط

بدون نقطه‌ویرگول، برنامه با خطا اجرا نمی‌شود.

کامنت‌ها در جاوا

کامنت‌ها خطوطی هستند که در اجرای برنامه نادیده گرفته می‌شوند و برای نوشتن توضیحات استفاده می‌شوند.

کامنت تک‌خطی:

```
// This is a single-line comment
System.out.println("Hello");
```

کامنت چندخطی:

```
/* This is a
   multi-line comment */
System.out.println("World");
```

قالب‌بندی و فاصله‌گذاری

جاوا نسبت به فاصله‌گذاری حساس نیست، اما برنامه‌نویسان حرفه‌ای کدهای خود را به‌صورت منظم می‌نویسند تا راحت‌تر خوانده شوند.

مثال نامنظم:

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

مثال منظم:

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

8.کاراکترهای ویژه(Escape Characters)

برای چاپ بعضی کاراکترهای خاص، باید از کاراکترهای فرار استفاده کنیم:

علامت	معنی
\n	رفتن به خط جدید
\t	فاصله‌ی افقی(tab)
\"	چاپ علامت دابل کوتیشن
\\	چاپ بک‌اسلش

مثال:

```
System.out.println("Line one\nLine two");
System.out.println("She said: \"Hello\"");
System.out.println("Path: C:\\Program Files\\Java");
System.out.println("Name\tAge");
```

استفاده از printf برای چاپ فرمت‌بندی‌شده

در جاوا، علاوه بر print() و println() می‌توانیم از printf() نیز استفاده کنیم.

کلمه‌ی printf مخفف **print formatted**(چاپ با قالب‌بندی) است و زمانی استفاده می‌شود که بخواهید متنی دقیقاً با یک فرمت خاص چاپ شود؛ مثلاً تعداد ارقام اعشار، نحوه‌ی نمایش اعداد، یا ترکیب متن و عدد در یک خط.

ساختار کلی:

```
System.out.printf("format string", values);
```

مثال ساده:

```
String name = "Sara";

int age = 20;


System.out.printf("Name: %s, Age: %d", name, age);
```

خروجی:

```
Name: Sara, Age: 20
```

علامت‌های جایگزین(Format Specifiers)

در printf() از علامت‌هایی به نام *specifier* استفاده می‌کنیم تا مشخص کنیم چه نوع داده‌ای را می‌خواهیم چاپ کنیم.

Specifier	توضیح
%d	عدد صحیح(int)
%f	عدد اعشاری(float, double)
%s	رشته(String)
%c	یک کاراکتر(char)
%%	چاپ علامت درصد %

کنترل تعداد ارقام اعشار

برای نمایش دقیق‌تر اعداد اعشاری می‌توانید تعداد ارقام بعد از نقطه را مشخص کنید:

```
double pi = 3.14159265;

System.out.printf("Value of pi: %.2f", pi);
```

خروجی:

```
Value of pi: 3.14
```

- در این مثال، %2f. یعنی فقط دو رقم بعد از نقطه‌ی اعشار چاپ شود.

اضافه کردن خط جدید درprintf

در printf، برای رفتن به خط بعد باید از \n استفاده کنید:

```
System.out.printf("Hello %s! \nWelcome to Java.", "Ali");
```

ترکیب متن و چند مقدار

```
String product = "Book";
int quantity = 5;
double price = 12.99;
System.out.printf("Product: %s | Quantity: %d | Price: $%.2f\n", product, quantity, price);
```

خروجی:

```
Product: Book | Quantity: 5 | Price: $12.99
```

نکات تکمیلی:

- هر فایل جاوا می‌تواند فقط **یک کلاس public** داشته باشد.
- متد main() همیشه نقطه شروع برنامه است.
- استفاده از print() و println() بسته به نیاز شما برای کنترل چاپ است.