۳۳

۱۴۴۱
چهارشنبه
۲۵ محرم

2019
مهر ۹۸
Sep 25

معرفی متدها (توابع): یک بلاک از کد است که به هنگام فراخوانی، اجرا می‌شود.

متدها به برنامه‌نویس کمک می‌کنند تا با استفاده از آنها:
① برنامه‌نویسی ساده‌تر را دنبال کند،
② خواناتر،
③ و هم تکرار همه کدها را با قابلیت استفاده مجدد،
تولید نماید.

در ادامه سعی داریم، متدها را براساس داشتن یا نداشتن پارامترهای ورودی و خروجی در سه حالت رده‌بندی کرده و آنها را معرفی کنیم.

حالت ۱: متدهای فاقد پارامترهای ورودی و خروجی: ساده‌ترین حالت است و در قالب زیر تعریف می‌شود:

```
Static Void MethodName(){

// code block

}
```

و برای فراخوانی متد فوق از دستور MethodName(); در داخل برنامه استفاده می‌کنیم.

مهر ۹۸
Sep 26

پنجشنبه
۲۶ محرم

مثال از متدهای تعریف شده

نکته ۱: متدها در داخل کلاس تعریف می‌شوند و ضرورتی ندارد که حتماً قبل از متد main تعریف شوند بلکه می‌توانند قبل یا بعد از آن باشند.

مثال:

```
static void myMethod () {
    System.out.println("Hello World!");
}
```

فراخوانی: myMethod();

خروجی: Hello World!

حالت ۲: متدهای دارای پارامترهای ورودی و فاقد خروجی:

در این حالت پارامترهای ورودی به صورت متغیرهای قابل استفاده در داخل متد شناخته می‌شوند. قالب کلی برای تعریف این حالت به صورت زیر است:

```
static void MethodName ( Datatype1 Var Name1, ---- ,
                         Datatype n Var Name n) {
    //Code block
}
```

2019
مهر ۹۸
Sep 27

۱۴۴۱
جمعه
۲۷ محرم

۱۴۴۱ شنبه ۲۸ محرم

2019 مهر ۹۸ Sep 28

نحوه فراخوانی متد به صورت زیر است:

MethodName(Val$_1$, ..., Val$_n$);

نکته: Val$_i$ باید حاوی مقداری از نوع Datatype$_i$ باشد، در غیر اینصورت با خطای زمان کامپایل مواجه می‌شویم.

مثال:

```
Static Void myMethod (string VarName,
                      string VarFamily)
{
    System.out.Println ("Name is :" + VarName + ", Family is :" +
                        VarFamily);
}
```

فراخوانی:

```
myMethod ("Ali", "Sadeghi");
```

خروجی:

```
Name is: Ali, Family is: Sadeghi
```

حالت ۳: متدهای دارای پارامتر ورودی و خروجی:

در این حالت می‌بایست به جای کلمه کلیدی Void، نوع داده‌ای را که می‌خواهیم متد برگرداند را بگذاریم و از کلمه return در بدنه متد نیز برای برگرداندن خروجی متد استفاده کنیم.

کلیدی

قالب کلی برای تعریف حالت سوم به صورت زیر می باشد:

```
static datatype MethodName (Datatype1 VarName1, ...
                            Datatypen VarNamen)
{
    //code block
    return (Val);
}
```

نحوه فراخوانی: `Datatype Val = MethodName (Val1, ..., Valn);`

مثال:

```
static int myMethod (int x, int y)
{
    int sum = x+y;
    return sum;
}
```

فراخوانی:

```
int Val= myMethod(5,10);
System.out.println("Sum: "+Val);
```

خروجی:

```
Sum: 15
```

۸

۱۴۴۱ دوشنبه ۱ صفر

2019 مهر ۹۸ Sep 30

معرفی متدها (یا توابع) بازگشتی (Recursion):

تکنیک (رویکردی) برای حل مساله به کمک متدهاست که در آن یک متد خودش را فراخوانی می‌کند.

در واقع با استفاده از این تکنیک، مساله به مساله‌های کوچکتر شکسته می‌شود و مساله‌های کوچکتر که معمولاً حل آنها ساده‌تر است حل شده و نهایتاً با ترکیب حل‌ها، حل کل مساله حاصل می‌شود.

هم چنین رویکردی ممکن است در آغاز کمی سخت به نظر برسد. با این حال در ادامه سعی داریم بازگشت را به کمک چند مثال تشریح کنیم.

مثال ۱: محاسبه فاکتوریل:

```
static int fact(int k)
{
    if (k>1) {
        return k* fact(k-1)
    }
    else
    {
        return 1;
    }
}
```

```
system.out.Println ( fact(5));
```

فراخوانی :

خروجی : 120

مثال ۲. محاسبه جمع توالی اعداد 1 و 2 و ... و n :

```
static  int  sum ( int n)
{
    if ( n>0)
    {
      return  n+sum(n-1);
    }
    else
    {
       return 0;
    }
}
```

```
System.out.Println (sum(5));
```

فراخوانی :

خروجی : 15