Topics: Selection (conditional) statement, keyboard input using `Scanner` class, `while` loop, methods
Reading: Sec 2.2, Sec 3.1 (exclude pp104-109), Sec 3.2 (exclude pp112-114), pp 130-135 of Sec 3.3 (exclude the `do-while` loop)

Example: Quadratic function, re-visited

Write a program to find the minimum value of the quadratic function $q(x)=x^2+bx+c$ on the interval [L, R].

```
/* Min value of q(x) = x^2 + bx + c on interval [L,R]
 */
public class MinQuadratic {
  public static void main(String[] args) {

    final double b=2, c=-1.5;
    double L=-3, R=5;
    double qMin, qL, qR;  // Min value of q, q(L), q(R)

    double xc= -b/2;
    if (L<=xc && xc<=R)
      // qMin is q(xc)
      qMin= xc*xc + b*xc + c;
    else {
      // qMin is q(L) or q(R)
      qL= L*L + b*L + c;
      qR= R*R + b*R + c;
      if (qL < qR)
        qMin= qL;
      else
        qMin= qR;
    }

    System.out.println("Min value is " + qMin);
  }
}
```

User Input

We'll use the class **`Scanner`** to read in user input from the keyboard. First, you need to *import* the class using the **`import`** statement *outside of the class body*:

```
import java.util.Scanner;
```

Inside a method (e.g., **`main`** method), you create an object of the **`Scanner`** class. Below, we create such an object and refer to it with the variable **`keyboard`**:

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

Now we can use **`keyboard`** to read user input. Below are some example method calls. Read Sec 2.2 (*Savitch*) for more information on the **`Scanner`** class.

Examples:

```
int var1= keyboard.nextInt();
double var2= keyboard.nextDouble();
char var3= keyboard.nextChar();
boolean var4= keyboard.nextBoolean();
```

The `Math` class

A collection of basic mathematical functions.

```
double tmp = Math.exp(1);
tmp = 3*Math.sin(2);
tmp = Math.floor(3*Math.sin(2));
tmp = Math.random();  // in [0,1)
```

Shortcut expressions

Increment: **`i++;`**
Decrement: **`i--;`**

Assignment operators:
```
s += val;
s -= val;
s *= val;
s /= val;
```

Conditional Statement

```
if ( condition1 )
   statement1;
```

```
if ( condition1 )
   statement1;
else
   statement2;
```

```
if ( condition1 )
   statement1;
else if ( condition2 )
   statement2;
else
   statement3;
```

The `while` loop

```
while ( condition )

    statement ;
```

Pattern for doing something *n* times

```
int i= 1;
while ( i<=n ) {
    // do something

    // increment counter
       i= i + 1;
}
```

Pattern for doing something an indefinite number of times

```
% initialization

while ( not stopping signal ) {
    // do something

    // update status (variables)
}
```

Example: Factorial

Write a program fragment to calculate *k*! (the factorial of *k*). Assume *k* is given and *k*>=0. Use a **`while`** loop.

Method

```
/* = a random integer in [lo..hi]
 */
public static int randInt(int lo, int hi) {
  return (int) Math.floor(
                Math.random()*(hi-lo+1) + lo);
}
```


Method

A method is a named, parameterized group of statements

modifiers *return-type* *method-name* **(** *parameter-list* **) {**
    statement-list
}

return-type **`void`** means nothing is returned from the method

There must be a **`return`** statement, unless return-type is **`void`**

Method

A method is a named, parameterized group of statements

modifiers *return-type* *method-name* **(** *parameter-list* **) {**
    statement-list
}

parameter-list : type-name pairs separated by commas

```
int randInt(int lo, int hi)
```

A parameter is a variable that is declared in the method

Calling a **`static`** method ...

... that is in a different class:

classname.*methodname*(...)

Examples: **`Math.random()`**
`Math.pow(2.5,2)`

... that is in the same class:

***`methodname`*`(…)`**

Our class **`MyRandom`** has a **`static`** method **`randInt`**, so an example method call within the class can be

`randInt(3,8)`