Topics: Selection (conditional) statement, `while` and `for` loops, methods, keyboard input using `Scanner` class
Reading: Sec 3.1-3.5, 3.7, 3.11; Sec 4.1-4.3, 4.5-4.7

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);
  }
}
```

Conditional Statement

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

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

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

Use **{}** to enclose a *block statement. For example,*

```
if ( condition1 ) {
   statement1;
   statement2;
}
else
   statement3;
```

The `while` loop

```
while ( condition )

     statement-to-repeat ;
```

Pattern for doing something *n* times

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

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

Shortcut expressions

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

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

The `for` loop

```
for ( initialization; condition; update )

     statement-to-repeat ;
```

Initialization, *condition*, and *update* are not required, but the semi-colons (;) are required

How a **`for`** loop is executed:

- *Initialization* is done once, before loop begins
- *condition* is evaluated
- Loop body executes only if *condition* evaluates to **true**
- *update* is executed. Then **loop back to evaluate the** ***condition***

Pattern for doing something *n* times

```
for ( int i=0; i<n; i++ )

    // do something
```

Example: Factorial

Write two program fragments to calculate *k*! (the factorial of *k*), one with a **`while`** loop and the other with a **`for`** loop. Assume *k* is given and *k*>=0.

Methods

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`**
- *parameter-list* :
  - type-name pairs separated by commas. Example: **`int lo, int hi`**
  - A parameter is a variable that is declared *in* the method

Calling a `static` method

Calling a `static` method that is in a <u>different</u> class: *`classname.methodname`*`(…)`
Examples: **`Math.random()`**
`Math.pow(2.5,2)`
Calling a `static` method that is in the <u>same</u> class: *`methodname`*`(…)`
For example, our class **`MyRandom`** has a **`static`** method **`randInt`**, so an example method call <u>within</u> the class can be
`randInt(3,8)`

See the complete file with more methods and example method calls online!

```
import java.util.Scanner;

/* Methods for generating random numbers and letters */
public class MyRandom {

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

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

    Scanner keyboard= new Scanner(System.in);
    System.out.println("Enter lower bound: ");
    int L= keyboard.nextInt();
    System.out.println("Enter upper bound: ");
    int R= keyboard.nextInt();

    int r= randInt(L, R);
    System.out.println("Random int in [" + L + ".." + R + "]:  " + r);
  }

}//class MyRandom
```

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 2.13 (*Gaddis*) for more information on the **`Scanner`** class.

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