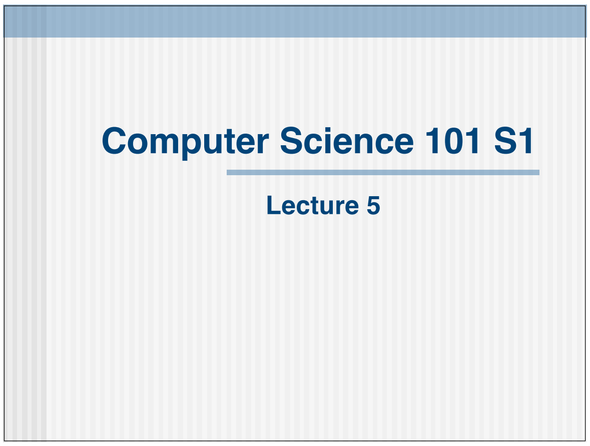
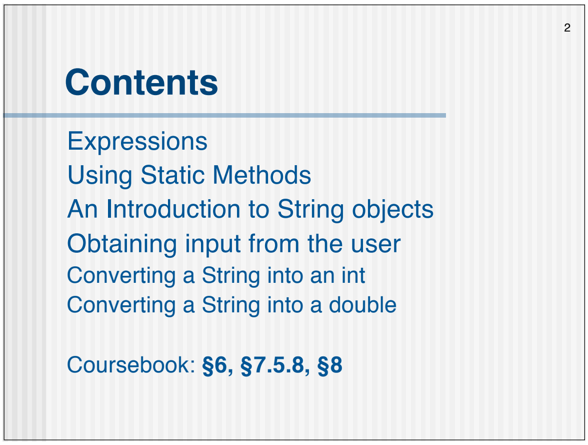


Computer Science 101 S1

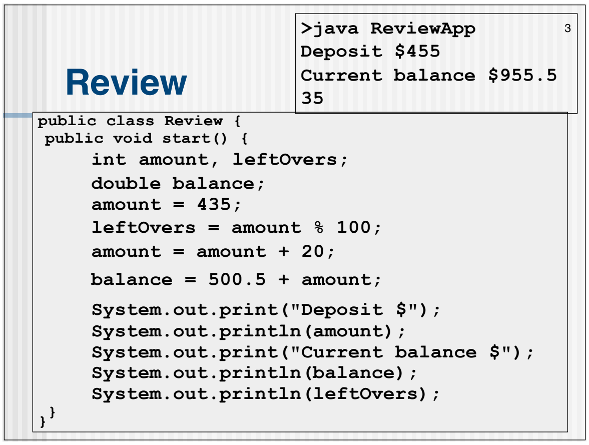
Lecture 5



Contents

Expressions
Using Static Methods
An Introduction to String objects
Obtaining input from the user
Converting a String into an int
Converting a String into a double

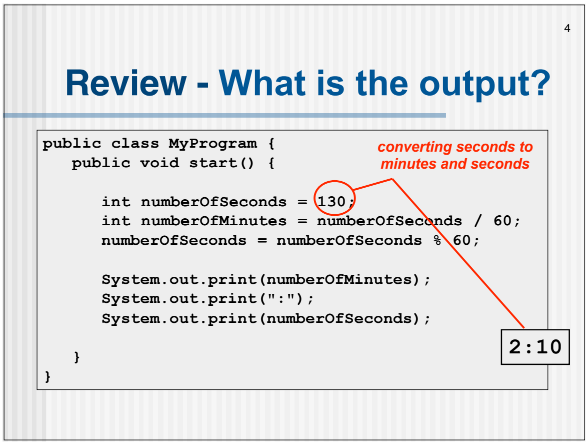
Coursebook: **§6, §7.5.8, §8**



Review

```
>java ReviewApp
Deposit $455
Current balance $955.5
35
```

```
public class Review {
 public void start() {
     int amount, leftOvers;
     double balance;
     amount = 435;
     leftOvers = amount % 100;
     amount = amount + 20;
     balance = 500.5 + amount;
     System.out.print("Deposit $");
     System.out.println(amount);
     System.out.print("Current balance $");
     System.out.println(balance);
     System.out.println(leftOvers);
 }
}
```

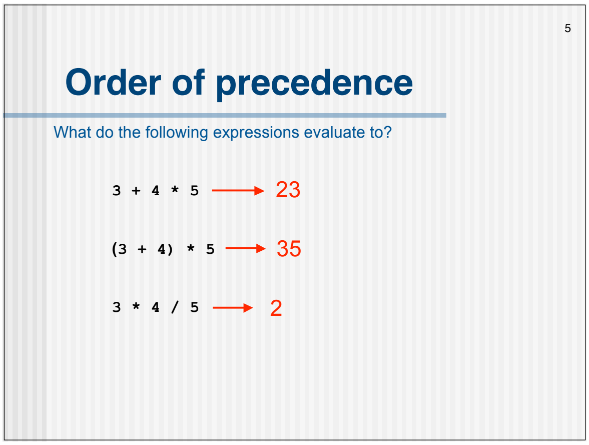


Review - What is the output?

```
public class MyProgram {
   public void start() {

      int numberOfSeconds = 130;
      int numberOfMinutes = numberOfSeconds / 60;
      numberOfSeconds = numberOfSeconds % 60;

      System.out.print(numberOfMinutes);
      System.out.print(":");
      System.out.print(numberOfSeconds);

   }
}
```

converting seconds to minutes and seconds

```
2:10
```



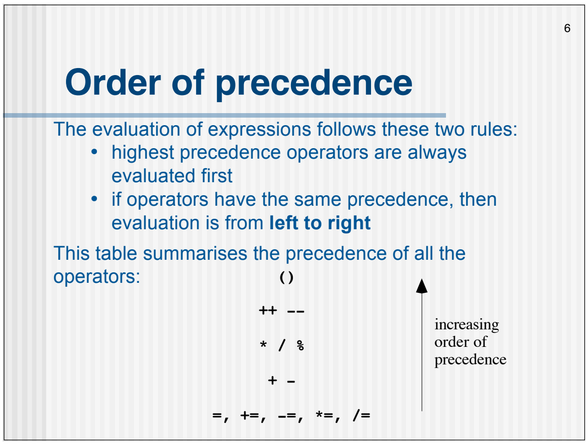
Order of precedence

What do the following expressions evaluate to?

`3 + 4 * 5` → 23

`(3 + 4) * 5` → 35

`3 * 4 / 5` → 2



Order of precedence

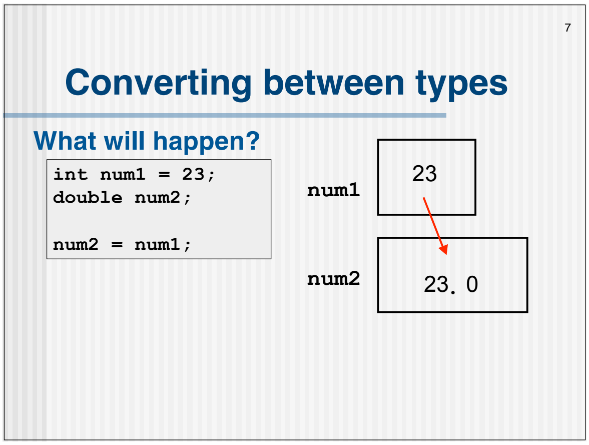
The evaluation of expressions follows these two rules:

- highest precedence operators are always evaluated first
- if operators have the same precedence, then evaluation is from **left to right**

This table summarises the precedence of all the operators:

```
()
++ --
* / %
+ -
=, +=, -=, *=, /=
```

increasing order of precedence

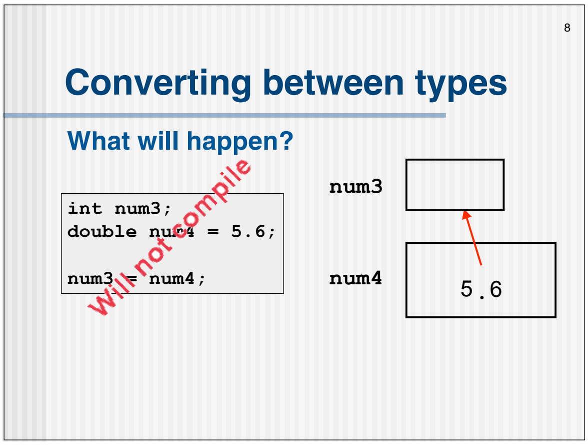


Converting between types

What will happen?

```
int num1 = 23;
double num2;

num2 = num1;
```

num1
23
num2
23. 0



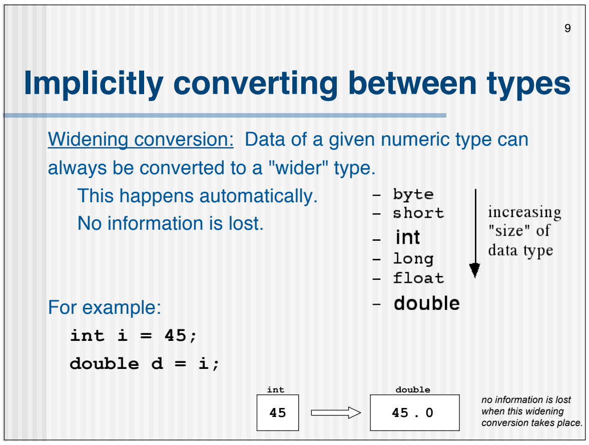
Converting between types

What will happen?

```
int num3;
double num4 = 5.6;

num3 = num4;
```

Will not compile

num3
num4
5.6



Implicitly converting between types

Widening conversion: Data of a given numeric type can always be converted to a "wider" type.

This happens automatically.
No information is lost.

- byte
- short
- int
- long
- float
- double

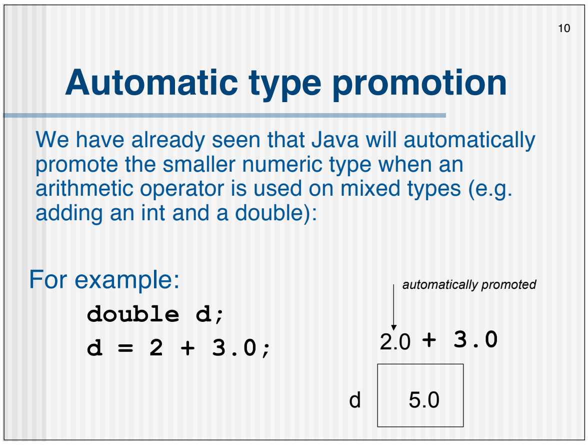
increasing "size" of data type

For example:

```
int i = 45;
double d = i;
```

int
45
double
45.0

no information is lost when this widening conversion takes place.



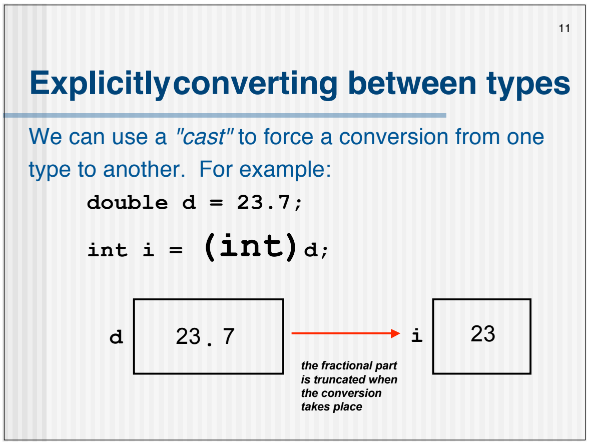
Automatic type promotion

We have already seen that Java will automatically promote the smaller numeric type when an arithmetic operator is used on mixed types (e.g. adding an int and a double):

For example:

```
double d;
d = 2 + 3.0;
```

automatically promoted

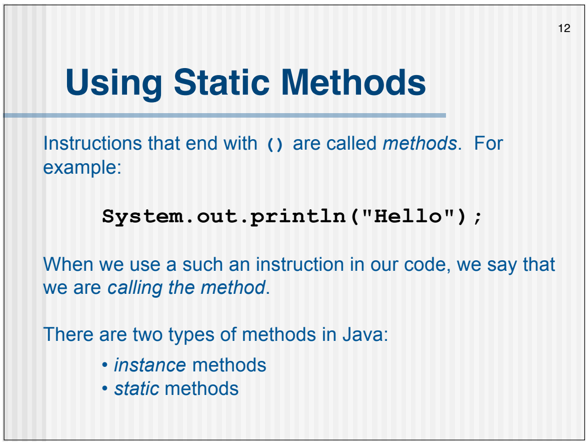


Explicitlyconverting between types

We can use a *"cast"* to force a conversion from one type to another. For example:

```
double d = 23.7;
int i = (int)d;
```

the fractional part is truncated when the conversion takes place



Using Static Methods

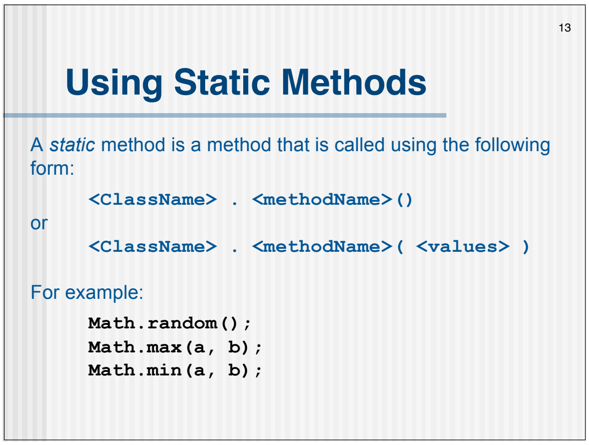
Instructions that end with `()` are called *methods*. For example:

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

When we use a such an instruction in our code, we say that we are *calling the method*.

There are two types of methods in Java:

- *instance* methods
- *static* methods



Using Static Methods

A *static* method is a method that is called using the following form:

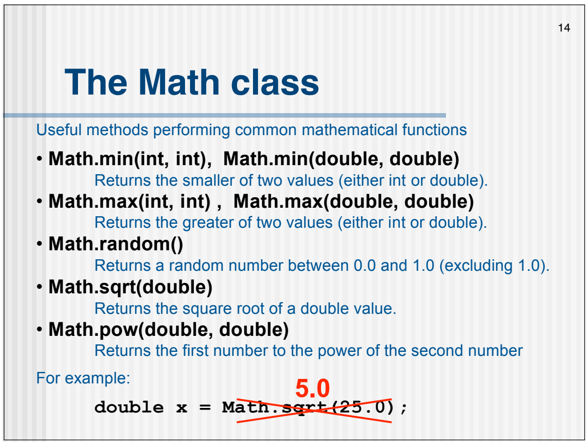
```
    <ClassName> . <methodName>()
```

or

```
    <ClassName> . <methodName>( <values> )
```

For example:

```
    Math.random();
    Math.max(a, b);
    Math.min(a, b);
```



The Math class

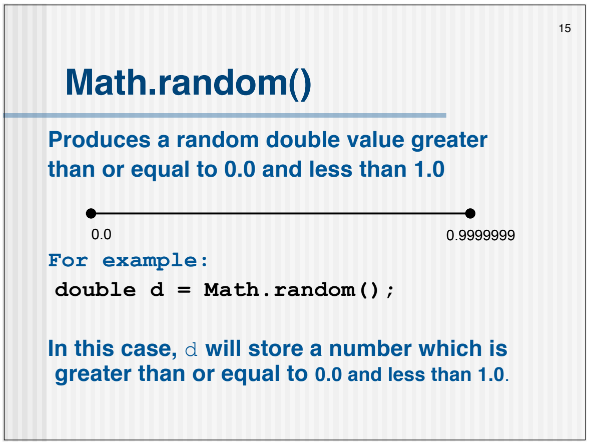
Useful methods performing common mathematical functions

- **Math.min(int, int), Math.min(double, double)**
  Returns the smaller of two values (either int or double).
- **Math.max(int, int) , Math.max(double, double)**
  Returns the greater of two values (either int or double).
- **Math.random()**
  Returns a random number between 0.0 and 1.0 (excluding 1.0).
- **Math.sqrt(double)**
  Returns the square root of a double value.
- **Math.pow(double, double)**
  Returns the first number to the power of the second number

For example:

5.0

double x = ~~Math.sqrt(25.0)~~;



Math.random()

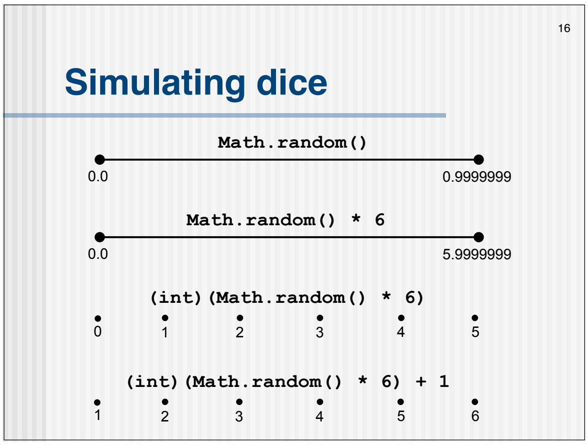
Produces a random double value greater than or equal to 0.0 and less than 1.0

0.0
0.9999999

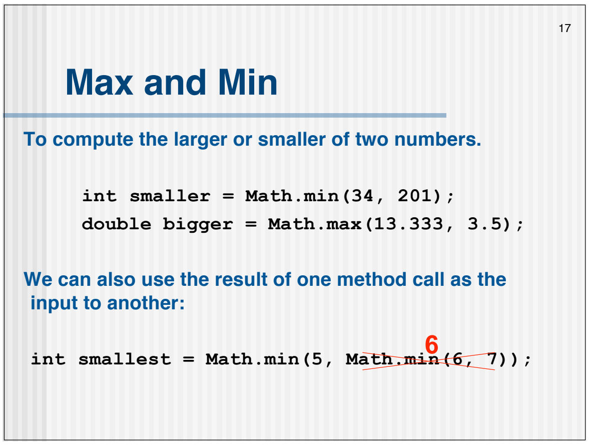
For example:

```
double d = Math.random();
```

In this case, `d` will store a number which is greater than or equal to 0.0 and less than 1.0.



Simulating dice



Max and Min

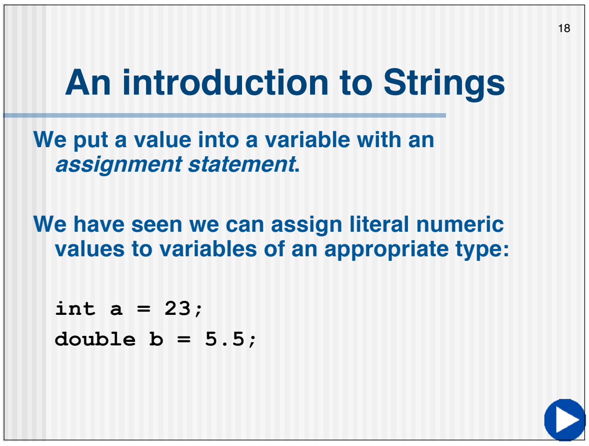
To compute the larger or smaller of two numbers.

```
int smaller = Math.min(34, 201);
double bigger = Math.max(13.333, 3.5);
```

We can also use the result of one method call as the input to another:

6

int smallest = Math.min(5, ~~Math.min(6, 7)~~);

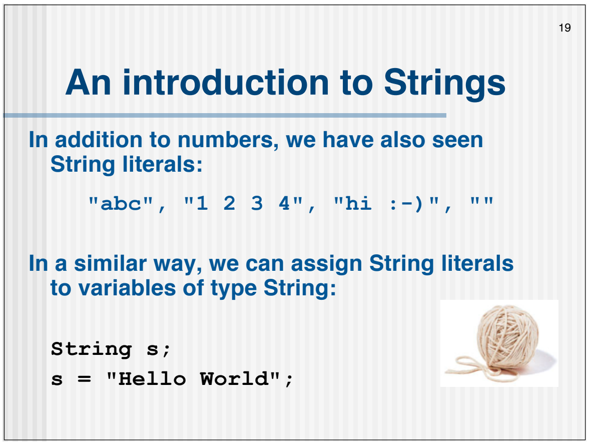


An introduction to Strings

We put a value into a variable with an *assignment statement.*

We have seen we can assign literal numeric values to variables of an appropriate type:

```
int a = 23;
double b = 5.5;
```



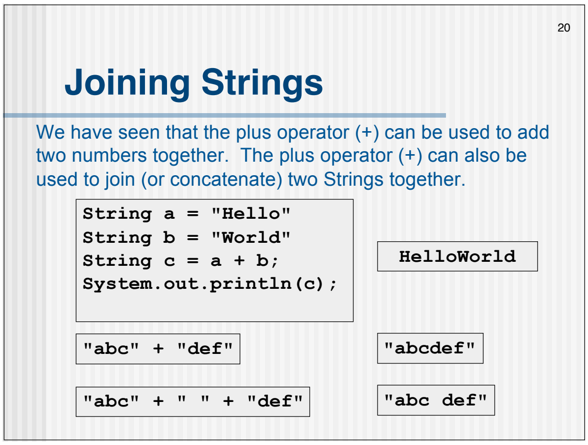
An introduction to Strings

In addition to numbers, we have also seen String literals:

```
"abc", "1 2 3 4", "hi :-)", ""
```

In a similar way, we can assign String literals to variables of type String:

```
String s;
s = "Hello World";
```



Joining Strings

We have seen that the plus operator (+) can be used to add two numbers together. The plus operator (+) can also be used to join (or concatenate) two Strings together.

```
String a = "Hello"
String b = "World"
String c = a + b;
System.out.println(c);
```

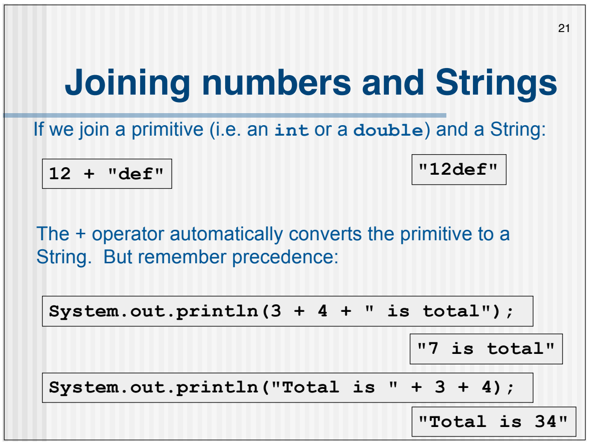
```
HelloWorld
```

```
"abc" + "def"
```

```
"abcdef"
```

```
"abc" + " " + "def"
```

```
"abc def"
```



Joining numbers and Strings

If we join a primitive (i.e. an `int` or a `double`) and a String:

```
12 + "def"
```

```
"12def"
```

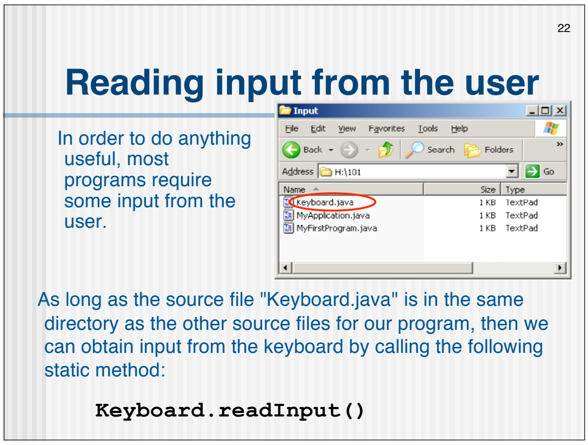
The + operator automatically converts the primitive to a String. But remember precedence:

```
System.out.println(3 + 4 + " is total");
```

```
"7 is total"
```

```
System.out.println("Total is " + 3 + 4);
```

```
"Total is 34"
```



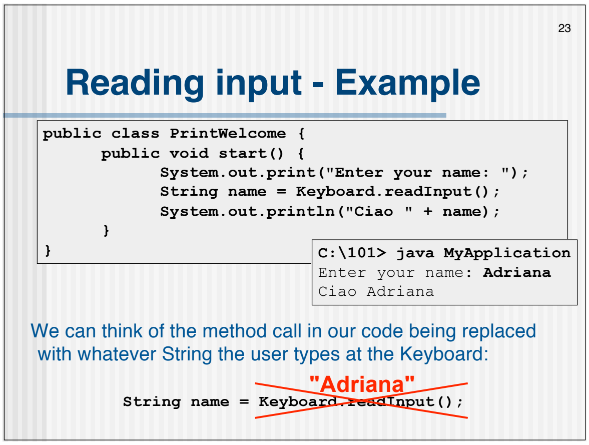
Reading input from the user

In order to do anything useful, most programs require some input from the user.

Name	Size	Type
Keyboard.java	1 KB	TextPad
MyApplication.java	1 KB	TextPad
MyFirstProgram.java	1 KB	TextPad

As long as the source file "Keyboard.java" is in the same directory as the other source files for our program, then we can obtain input from the keyboard by calling the following static method:

```
Keyboard.readInput()
```



Reading input - Example

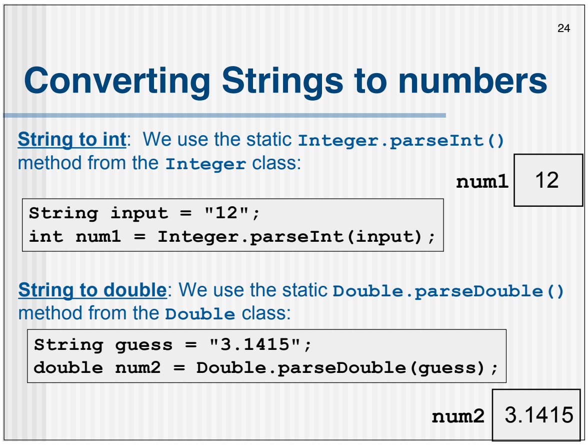
```
public class PrintWelcome {
      public void start() {
            System.out.print("Enter your name: ");
            String name = Keyboard.readInput();
            System.out.println("Ciao " + name);
      }
}
```

```
C:\101> java MyApplication
Enter your name: Adriana
Ciao Adriana
```

We can think of the method call in our code being replaced with whatever String the user types at the Keyboard:

"Adriana"

String name = ~~Keyboard.readInput()~~;



Converting Strings to numbers

String to int: We use the static `Integer.parseInt()` method from the `Integer` class:

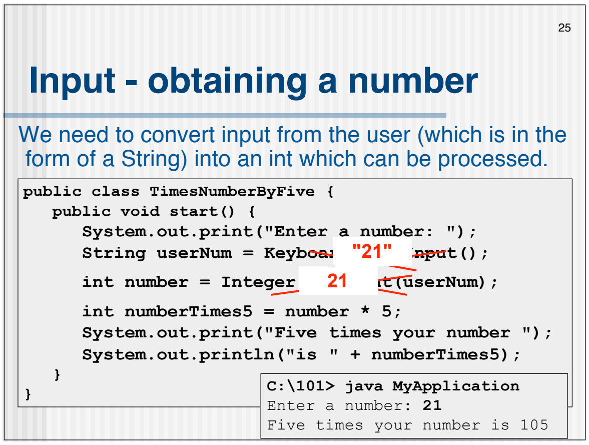
```
String input = "12";
int num1 = Integer.parseInt(input);
```

num1	12

String to double: We use the static `Double.parseDouble()` method from the `Double` class:

```
String guess = "3.1415";
double num2 = Double.parseDouble(guess);
```

num2	3.1415

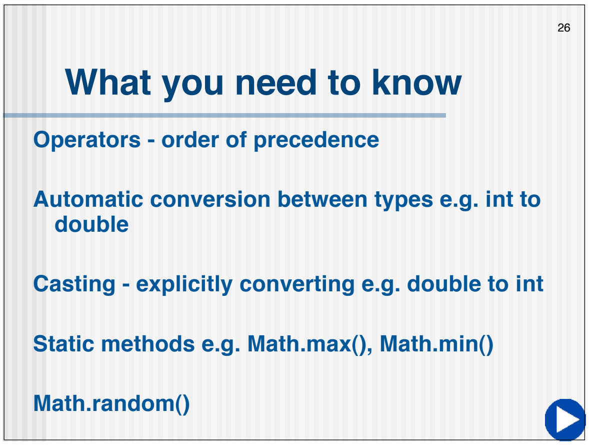


Input - obtaining a number

We need to convert input from the user (which is in the form of a String) into an int which can be processed.

```
public class TimesNumberByFive {
   public void start() {
      System.out.print("Enter a number: ");
      String userNum = Keyboar "21" nput();
      int number = Integer  21  t(userNum);
      int numberTimes5 = number * 5;
      System.out.print("Five times your number ");
      System.out.println("is " + numberTimes5);
   }
}
```

```
C:\101> java MyApplication
Enter a number: 21
Five times your number is 105
```



What you need to know

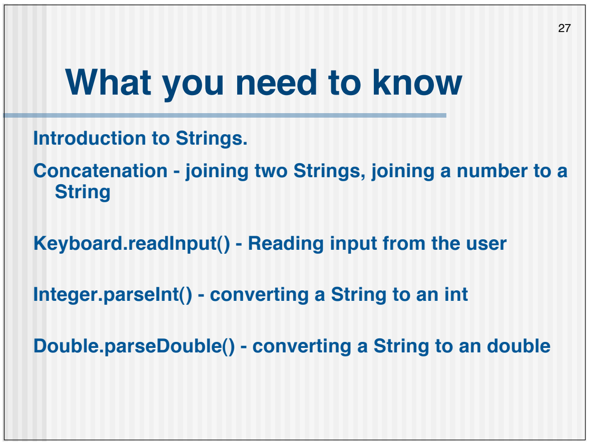
Operators - order of precedence

Automatic conversion between types e.g. int to double

Casting - explicitly converting e.g. double to int

Static methods e.g. Math.max(), Math.min()

Math.random()



What you need to know

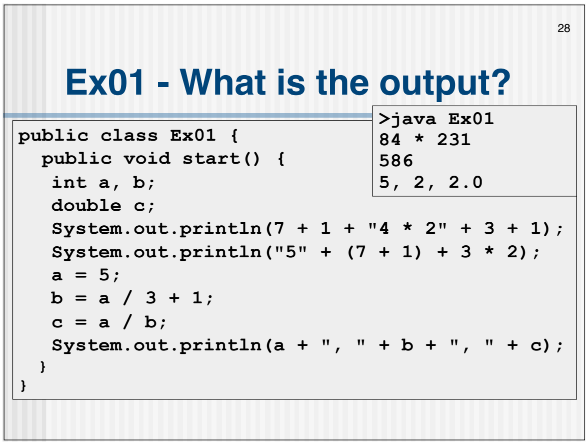
Introduction to Strings.

Concatenation - joining two Strings, joining a number to a String

Keyboard.readInput() - Reading input from the user

Integer.parseInt() - converting a String to an int

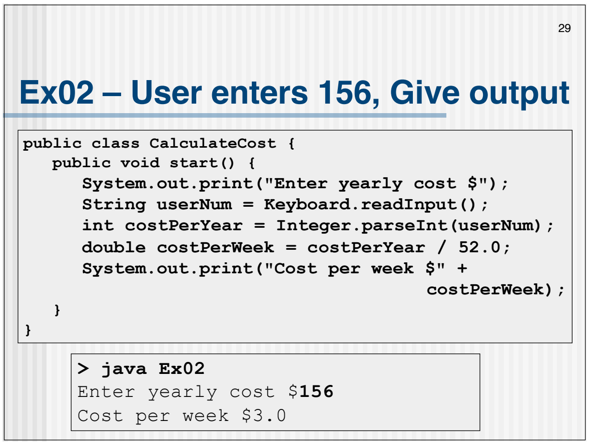
Double.parseDouble() - converting a String to an double



Ex01 - What is the output?

```
public class Ex01 {
  public void start() {
   int a, b;
   double c;
   System.out.println(7 + 1 + "4 * 2" + 3 + 1);
   System.out.println("5" + (7 + 1) + 3 * 2);
   a = 5;
   b = a / 3 + 1;
   c = a / b;
   System.out.println(a + ", " + b + ", " + c);
  }
}
```

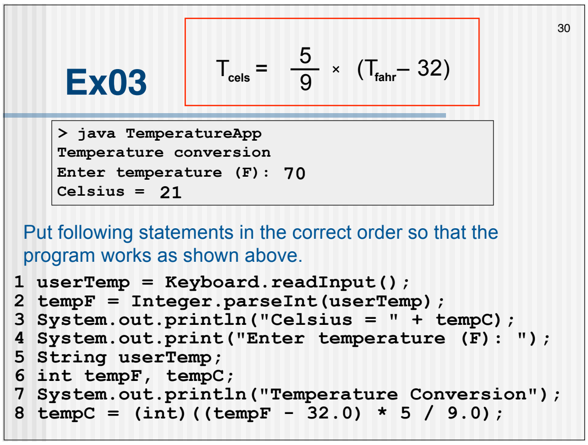
```
>java Ex01
84 * 231
586
5, 2, 2.0
```



Ex02 – User enters 156, Give output

```
public class CalculateCost {
   public void start() {
      System.out.print("Enter yearly cost $");
      String userNum = Keyboard.readInput();
      int costPerYear = Integer.parseInt(userNum);
      double costPerWeek = costPerYear / 52.0;
      System.out.print("Cost per week $" +
                                  costPerWeek);
   }
}
```

```
> java Ex02
Enter yearly cost $156
Cost per week $3.0
```



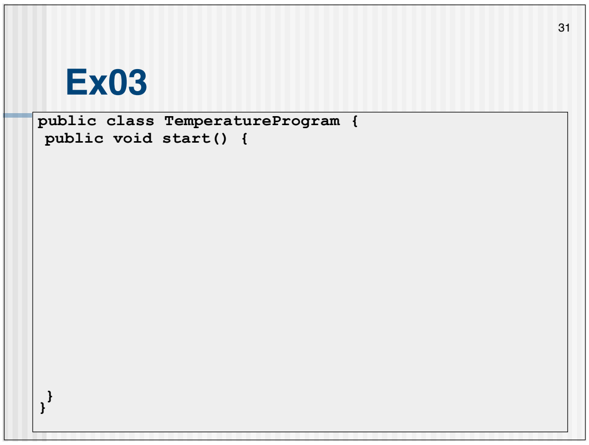
Ex03

$$T_{cels} = \frac{5}{9} \times (T_{fahr} - 32)$$

```
> java TemperatureApp
Temperature conversion
Enter temperature (F): 70
Celsius = 21
```

Put following statements in the correct order so that the program works as shown above.

```
1 userTemp = Keyboard.readInput();
2 tempF = Integer.parseInt(userTemp);
3 System.out.println("Celsius = " + tempC);
4 System.out.print("Enter temperature (F): ");
5 String userTemp;
6 int tempF, tempC;
7 System.out.println("Temperature Conversion");
8 tempC = (int)((tempF - 32.0) * 5 / 9.0);
```



Ex03

```
public class TemperatureProgram {
 public void start() {









 }
}
```