Lecture 5: class activities

prof. dr. Irma Ravkic

Start a new project in your workspace or add a package for Activity 5.

Activity 5.1 Recursion 1: sum

You: Write an **iterative** method that will find a sum of elements in an array.
Together: Write a **recursive** method doing the same.
(*You can find the demonstration of this solution in the textbook*)

Activity 5.2 Reference trouble

What would you expect to be printed in line 9? Type this code in a class in your project. Is the result what you expected? Why yes/not? How would you change this code to produce the output you expected?

```
1  public class ReferenceDemo{
   public static void incrementIntegerByOne(Integer c){
3      c = c + 1;
   }
5
   public static void main(String[] args){
7      Integer b = new Integer(3);
       incrementIntegerByOne(b);
9      System.out.println(b);
   }
11 }
```

Activity 5.3 Recursion 2: traversing a linked list

For this activity you need your LLNode⟨T⟩ code. Please re-type it or copy paste it into your project first. And then solve the exercise.

You: Write **iterative** methods that will a) traverse a linked list iteratively to print its values, b) calculate the number of elements in a linked list.

Together: Write **recursive** methods doing the same as above.
(*You can find the demonstration of this solution in the textbook*)

Activity 5.4 Recursion 3: transforming a linked list

For this activity you need your LLNode⟨T⟩ code. Please re-type it or copy paste it into your project first. And then solve the exercise.

You: Write an **iterative** method with the following 'pseudo' method signature (you need to fill out the ? with the right concept):

```
public static ? addToEnd(LLNode<?> node);
```

that will add a new node to the end of a linked list. Use a small example, print it to make sure it works correctly.

Together: Write a **recursive** method doing the same as above. Test it for an empty linked list!

Activity 5.5 Recursion 4: Towers of Hanoi Demo

The Tower of Hanoi puzzle was invented by the French mathematician Edouard Lucas in 1883. He was inspired by a legend that tells of a Hindu temple where the puzzle was presented to young priests. At the beginning of time, the priests were given three poles and a stack of 64 gold disks, each disk a little smaller than the one beneath it. Their assignment was to transfer all 64 disks from one of the three poles to another, with two important constraints. They could only move one disk at a time, and they could never place a larger disk on top of a smaller one. The priests worked very efficiently, day and night, moving one disk every second. When they finished their work, the legend said, the temple would crumble into dust and the world would vanish. Although the legend is interesting, you need not worry about the world ending any time soon. The number of moves required to correctly move a tower of 64 disks is $2^{64-1} = 18,446,744,073,709,551,615$. At a rate of one move per second, that is $584,942,417,355$ years! Clearly there is more to this puzzle than meets the eye.

We will see how this problem can be solved recursively with 3 disks (so without the world vanishing).

You can see the problem setup in the book or in the slides I shared with you. In your class activity package you have a **Tower.java** file. Add it to your existing activity project under a different package called **demos**.