Induction is a very useful proof technique for proving correctness and bound the running time of an algorithm. Most of the statements about the running times of algorithms that we'll see during this course will hold *for all* $n > 0$, where n is the size of the input. However, this is a claim about an infinite set of numbers and hence we can't prove it by proving it for each element separately.

This is where induction comes in. Induction proves the statement for a *base case*, the smallest element for which the claim must hold ($n = 1$). In some cases, it's helpful to prove an extended base case by explicitly showing that the claim holds for the first few elements, say $n = 1$, $n = 2$, and $n = 3$.

Next, we proceed to the *induction step*, which assumes that the claim holds for all values of n smaller than the current one. This assumption is called the *induction hypothesis*. Using the induction hypothesis, we then show that the claim holds for the next value of n. Typical applications of this are using the induction hypothesis for $n - 1$ to prove the claim for n or using the induction hypothesis for $n/2$ to prove the claim for n when n is even.

As a refresher, work out the problems in this tutorial sheet. If need, refer to Section 1.2.3 in the class textbook ("Algorithm Design and Applications" by Goodrich and Tamassia) for more detail background on induction and more examples.

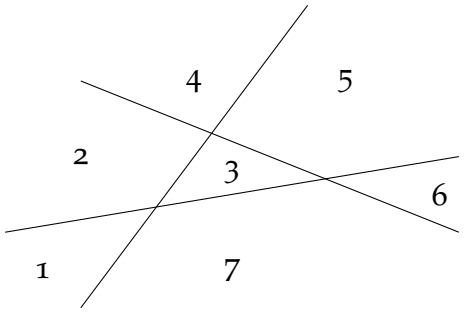
Warm-up

Problem 1. Use induction to show that $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$.

Problem 2. Recall that one way of defining the Fibonacci sequence is as follows: $F(1) = 1$, $F(2) = 2$, and $F(n) = F(n-2) + F(n-1)$ for $n > 2$.
Use induction to prove that $F(n) < 2^n$.

Problem solving

Problem 3. One line divides the plane into two halves or regions, while two lines divide the plane into four regions. In this problem we study the number of regions in which n lines divide the plane. For simplicity, assume that no three lines intersect at a given point and that no two lines are parallel to one another. Here is a picture of the 7 regions defined by 3 lines.



Consider the following algorithm that builds the set of regions by starting from a single region encompassing the whole plane, and iteratively refines the set of regions by processing one line at a time.

```
1   def build_arrangement(lines)
2     regions = [new region spanning whole plane]
3     for line in lines:
4       old = []
5       new = []
6       for region in regions:
7         if line intersects region:
8           append region to old
9           left, right = split region through line
10          append left to new
11          append right to new
12      for x in old: remove x from regions
13      for x in new: append x to regions
14    return regions
```

a) Prove that at the beginning of the ith iteration of the for loop in Line 3, the line cuts through exactly i regions from the previous iteration.

b) Prove that at the end of the ith iteration of the for loop in Line 3, we have $\frac{i^2+i+2}{2}$ regions.

Remember to use the fact that no three lines intersect in the same point!