

The next topic in the course is *recursion*. Recursion is closely related to a proof technique in mathematics called *mathematical induction*. Many of you have seen mathematical induction already, so this is a good place to start.

## Mathematical induction

Suppose we would like to prove some statement  $P(n)$  – often called a proposition – that involves all positive integers  $n$ . For example we would like to prove: for any  $n \geq 1$ ,

$$1 + 2 + 3 + \cdots + (n - 1) + n = \frac{n(n + 1)}{2}.$$

In this case,  $P(n)$  is the equation i.e. left side = right side. This proof can be done with mathematical induction.

The technique of mathematical induction requires us to prove two things:

1. *base case*: the statement  $P(n)$  is true for  $n = 1$
2. *induction step*: if the statement  $P(n)$  is true for any  $n = k$  where  $k \geq 1$  (called the “induction hypothesis”), then the statement is true for  $n = k + 1$ .

The logic is that you show  $P(n)$  for  $n = 1$  (base case), and then the induction step implies that the statement must be true for  $n = 2$  (since the induction hypothesis holds for  $n = 1$  which is the base case), which in turn implies that it must be true for  $n = 3$ , and so on.

Intuitively, think of a infinite sequence of rocks, numbered 1,2,3, etc. which goes out into a shallow but infinitely wide lake. Imagine you want to be able to stand on *any* of the rocks. To do so, it is sufficient that you know how to do two things: One thing is that you can stand on rock 1 (and not fall off). The second thing is that you know how to step from any rock to the next rock (and not fall off). If you have these two skills, then you have what it takes to stand on any rock  $n$ .

### Example 1

Prove the formula for the following well-known *arithmetic series*:

$$1 + 2 + 3 + \cdots + n = \sum_{i=1}^n i = \frac{n(n + 1)}{2}$$

The base case is easy to verify since

$$\sum_{i=1}^1 i = \frac{1 \cdot (1 + 1)}{2} = 1.$$

We prove the induction step as follows:

$$\begin{aligned} \sum_{i=1}^{k+1} i &= (k + 1) + \sum_{i=1}^k i &= (k + 1) + \frac{k(k + 1)}{2} &\text{by induction hypothesis} \\ &= \left(1 + \frac{k}{2}\right)(k + 1) \\ &= \frac{1}{2}(k + 2)(k + 1) &\text{which is what we wanted to show.} \end{aligned}$$

**Example 2**

Show that: for all  $n \geq 1$ ,

$$1 + 3 + 5 + \dots + (2n - 1) = n^2$$

The base case of  $n = 1$  is obvious, since there is only a single term on the left hand side, i.e.  $1 = 1^2$ . So, assume its true for  $n = k$  and prove for  $n = k + 1$ .

$$\begin{aligned} \sum_{i=1}^{k+1} (2i - 1) &= 2(k + 1) - 1 + \sum_{i=1}^k (2i - 1) \\ &= 2(k + 1) - 1 + k^2 \text{ by induction hypothesis} \\ &= 2k + 1 + k^2 \\ &= (k + 1)^2 \end{aligned}$$

which is what we were trying to prove.

**Example 3: strange one, failure of base case only**

A correct proof by mathematical induction requires that both the base case and induction step are proved. If one can be proved but not the other, then the proof is not valid.<sup>1</sup>

For example, suppose you wished to prove the (obviously incorrect) statement: for all  $n \geq 1$ ,  $n < n - 1$ .

The base case fails, since the proposition  $1 < 0$  is false. However, the induction step still holds. Why? The induction *hypothesis* is that  $k < k - 1$ . If this hypothesis were true, then adding 1 to both sides of the inequality gives that  $k + 1 < (k + 1) - 1$ , which is exactly  $P(k + 1)$  i.e. what we want to prove for the induction step!

If this example bothers you, it is likely because it is so obvious that the induction hypothesis  $k < k - 1$  is false. But for more challenging problems, *you will not be able to judge in advance whether the induction hypothesis is true or false*. It might be true, but it might not be. In trying to prove the induction step, your task is not to evaluate whether the hypothesis is true, but rather to evaluate whether the hypothesis  $P(k)$  implies that  $P(k + 1)$ , for all  $k \geq 1$ .

**Example 4: more general base case**

Sometimes the base case involves a range of values of  $n$ , say  $n_0, \dots, n_1$ . Again you need to show two things. First, you need show that your proposition  $P(n)$  holds for the base case values. Second, you need to show that *if* the proposition holds for the base case values  $n = n_0, \dots, n_1$ , then it must also hold for the value  $n = n_1 + 1$ . By the same logic as before, the statement must therefore hold for all  $n \geq n_0$ .

1. *base case*: the statement is true for some range of consecutive integers  $n = n_0, \dots, n_1$
2. *induction step*: *if* the statement is true for the range  $k = n_0, \dots, n_1$  *then* the statement is also true for all  $k = n_1 + 1$ .

---

<sup>1</sup>Saying a proof is not valid is different from saying that the statement you are trying to prove is false. You can write an incorrect proof of a claim that is in fact true.

**Example 5**

For all  $n \geq 5, n^2 < 2^n$ . The interesting part of this proof is that the induction step requires us to prove that for all  $n \geq 5, 2n + 1 < 2^n$ . So the proof goes like this (*See the slides for the details.*) :

```

Prove that: for all  $n \geq 5, n^2 < 2^n$  {
  Base case:  $5^2 < 2^5$ 
  Induction step: for all  $k \geq 5, k^2 < 2^k$  implies  $(k+1)^2 < 2^{k+1}$ 
  ...
  Prove that for all  $n \geq 5, 2n + 1 < 2^n$  {
    Base case:  $2*5 + 1 < 2^5$ 
    Induction step: for all  $k \geq 5,$ 
                       $2k+1 < 2^k$  implies  $2*(k+1)+1 < 2^{k+1}$ 
  }
}

```

This example should bring to mind that idea of a call stack discussed in lecture 7.

**Example 6: upper bound on Fibonacci numbers**

Consider the Fibonacci<sup>2</sup> sequence: 0, 1, 1, 2, 3, 5, 8, 13, 21, ... where

$$F(0) = 0, F(1) = 1,$$

and, for all  $n > 1$ ,

$$F(n) = F(n-1) + F(n-2).$$

Let's use induction to prove: for all  $n > 0$ ,

$$F(n) < 2^n. \quad (*)$$

Because the Fibonacci numbers sequence uses the previous two values, the base case requires two values, namely  $n = 0, 1$ . Note  $F(0) = 0, F(1) = 1$  and so (\*) holds for the base case since  $F(0) = 0 < 2^0$  and  $F(1) = 1 < 2^1$ .

For the induction step, we assume that  $F(n) < 2^n$  for all  $0 \leq n \leq k$  and show that (\*) holds for  $n = k + 1$ :

$$\begin{aligned}
 F(k+1) &= F(k) + F(k-1) \\
 &< 2^k + 2^{k-1} \text{ by induction hypothesis} \\
 &< 2^k + 2^k \\
 &= 2^{k+1}
 \end{aligned}$$

and so the induction step is also proven and we are done.

<sup>2</sup>[http://en.wikipedia.org/wiki/Fibonacci\\_number](http://en.wikipedia.org/wiki/Fibonacci_number)

## Induction and Recursion

Induction is used to prove many such statements in mathematics. It is also used to prove the correctness of certain computer programs. For example, “n factorial” is defined

$$n! = 1 \cdot 2 \cdot 3 \dots (n - 1) \cdot n$$

Here is an algorithm (written in Java) for computing it. The algorithm just applies the definition so there is nothing to prove about correctness.

```
int factorial(int n){ // assume n >= 1
    int result = 1;
    for (int i = 1; i <= n; i++)
        result *= i;
    return result;
}
```

But here is another way to define  $n!$  which is more subtle, namely if  $n > 1$ , then

$$n! = n \cdot (n - 1)!$$

The corresponding algorithm (written in Java) is:

```
int factorial(int n){ // algorithm assumes argument: n >= 1
    if (n == 1)
        return 1;
    else
        return n * factorial(n - 1);
}
```

Notice that the definition of the algorithm `factorial` involves a call to itself. Such an algorithm is said to be *recursive*.

Here is an induction argument to prove: “The algorithm `factorial(n)` computes  $n!$  for any input value  $n \geq 1$ .”

Base case: if the input argument is 1, then the algorithm returns 1, so the base case is proved.

Induction step: Suppose that the algorithm returns  $n!$  for any argument  $n$  from 1 to  $k$  (the induction hypothesis). We need to show that the algorithm returns  $(k + 1)!$  when input argument is  $n = k + 1$ . But this is easy, since when the argument is  $k$  the algorithm returns  $(k + 1) * k!$ , which is just  $(k + 1)!$ .