CSIT113
Problem Solving

UNIT 4
INDUCTION

UNIVERSITY OF WOLLONGONG AUSTRALIA

SIM GLOBAL EDUCATION

Overview

- Introducing Recursion
- Induction
- How to use Induction to solve Problems?
- Mathematical Induction

Methods for specifying Repetitive Processes

- There are two approaches to specifying repetitive process:
  - Iteration: Using for-loop or while-loop as discussed in Unit 1.
  - Recursion: The next few slides shall introduce recursion for specifying repetitive process.
- Though algorithms design from Induction can also be specified using iteration, however, recursion provides a direct and natural way (hence, easiest way) for specifying them.
- Next, we shall introduce Recursion.

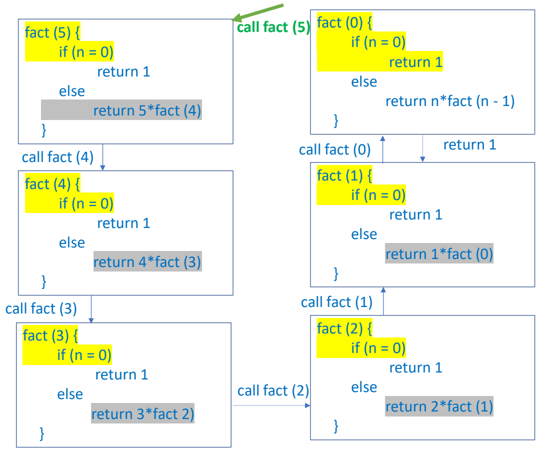
Introducing Recursion through Example - Factorial

- A recursive algorithm calls itself by supplying appropriate arguments to solve smaller sub-problem. For example,

```
fact (n) {
    if (n = 0)
        return 1
    else
        return n*fact (n - 1)
}
```

- Note that fact (n): $n! = n*(n-1)*......*2*1$, $0! = 1$. For example, $4! = 4*3*2*1 = 24$.

How does recursion work? – An illustration on execution of fact (5)



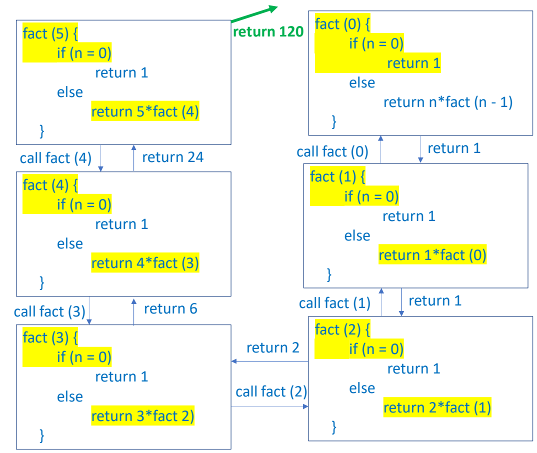
Upon execution of fact(5), altogether, 5 recursive call generated:

fact(4), fact(3), fact(2), fact(1) and fact(0)

statements executed

execution of a statement started but paused for execution of recursive call

How does recursion work? – An illustration on execution of fact (5)



fact(5) is calculated as follows:

- fact(0) = 1
- fact(1) = 1*fact(0) = 1*1 =1
- fact(2) = 2*fact(1) = 2*1 = 2
- fact(3) = 3*fact(2) = 3*2 = 6
- fact(4) = 4*fact(3) = 4*6 = 24
- fact(5) = 5*fact(4) = 5*24 = 120

statements executed

How to use Induction?

- Induction is an elegant problem solving strategy.
- It always solves a problem based on its **size** as follows:
  - ✓ **Smallest Size**: solve it directly.
  - ✓ **Otherwise:** Solve it by producing the solution of the whole problem from the solutions of suitable smaller subproblem(s).

The smaller subproblems will be solved in the same way too.

- We specify an **Induction Hypothesis** to state that these smaller subproblem(s) can be solved.
- Identifying the suitable smaller subproblems is **the key in using induction**.
- Induction can be used in designing algorithm and proving statements in mathematics (Mathematical Induction).

Problem Size

- Problem size is a quantitative property of a problem to measure of its size,
- Usually, it is not difficult to find such a property.
- For example, if we have a sequence of numbers to sort, the number of elements in the sequence is a good measure of size.
- Or, with the match games, the number of matches in the pile.

Using Induction for to Solve Problems

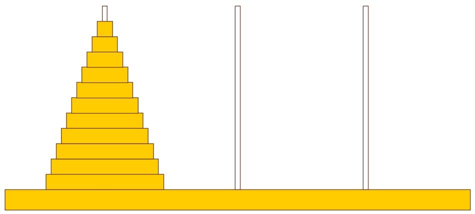
- We shall study the use of induction to solve problems through illustrative examples.
- We will discuss the first two examples in details and the remaining examples only briefly.

Problem 1 - The Towers of Hanoi

- According to legend, at the time of creation, god created three diamond needles set in a slab of pure gold and on one of them he placed 64 discs of pure gold, each smaller than the one below it.
- He tasked a group of monks with moving the discs from their starting needle to another needle.
- But the monks had to obey certain rules:

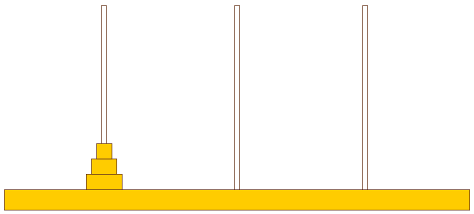
1. The discs may only be placed on the needles.
2. Only one disc may be moved at a time.
3. A disc must never be placed on a disc that is smaller than itself.

Problem 1 - The Towers of Hanoi



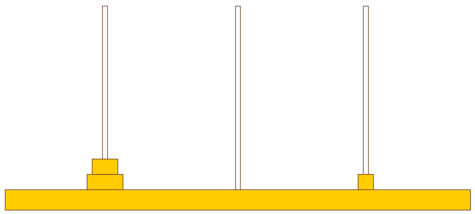
The Towers of Hanoi - A small example

- Let's try a small example.
- We organize the moves into the following two steps and perform alternatively until we have completed:
  - Step 1: move the smallest disc
  - Step 2: move the other disc that can be moved



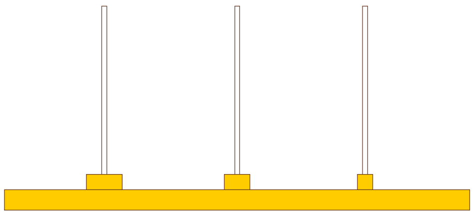
The Towers of Hanoi - A small example

- Step 1



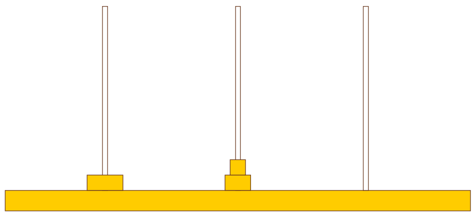
The Towers of Hanoi - A small example

- Step 2



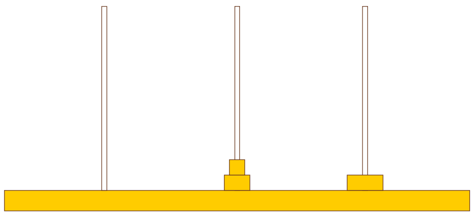
The Towers of Hanoi - A small example

- Step 1



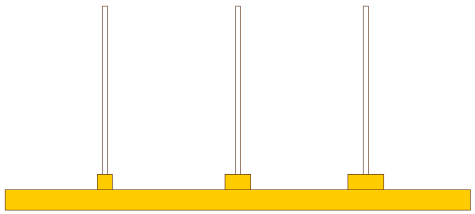
The Towers of Hanoi - A small example

- Step 2



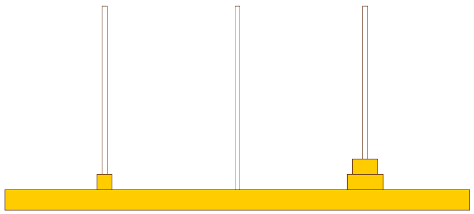
The Towers of Hanoi - A small example

- Step 1



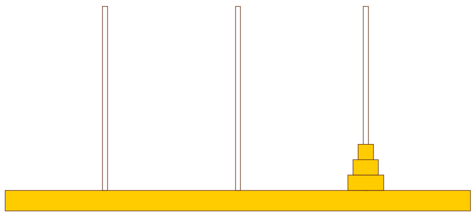
The Towers of Hanoi - A small example

- Step 2



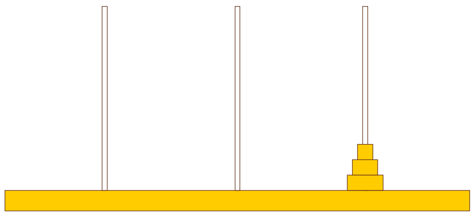
The Towers of Hanoi - A small example

- Step 1



The Towers of Hanoi - A small example

- All on the final needle – problem solved.



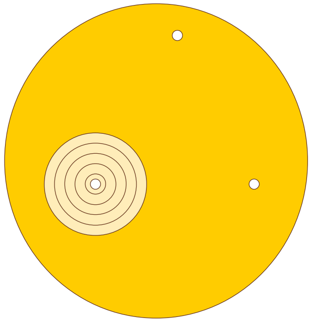
The Towers of Hanoi - A small example

- How did this solution arise?
- Why does it work?
- The problem with this solution is that it provides no insight into the answers to the above two question:
  - It's like pulling a rabbit out of a hat.
  - Magic not technology
- Let us approach the problem from a different perspective.
- Let's try induction to find solutions for moving any number of discs and in any direction such that the solutions can be automatically implemented.

The Towers of Hanoi

- Identifying suitable smaller subproblems such that we can use the solution(s) of these subproblem(s) to produce the solution of the whole problem is **the crucial task in using induction.**
- After that, we define an **induction hypothesis** to state that the above subproblems can be solved.
- For this purpose, let us make the problem more general.
- Imagine the towers are arranged in a circle.
- The discs are numbered as 1, 2, 3, ….n, according to the order of their sizes, such that disc 1 is the smallest disc and disc n is the largest disc.

The Towers of Hanoi



- Now we can move discs in two directions:
  - clockwise;
  - anticlockwise.

Now the problem becomes:
Move *n* discs to the needle in direction *d*, where *d* is either clockwise or anticlockwise.

The Towers of Hanoi

- We have to be more careful in how we state it.
- As we saw earlier, if we have small discs on the towers they block the movement of the larger discs.
- Our induction hypothesis must be that it is possible to move the *n* ***smaller*** discs one step in an arbitrary direction *d* starting from ***any valid position***.
- By ***valid*** we mean any position in which no disc is on top of n smaller discs and the *n* smaller discs are in a single pile.

The Towers of Hanoi

- We identify the number of discs, **n**, as the size of this problem.
- For n = 0 the solution is obvious and trivial – no move needed.
- If we can move k discs in direction d, we can also move k discs in direction $\sim d$, simply by "reversing" the direction of the pattern.

The Towers of Hanoi – Recursive Solution

- Induction hypothesis: we can move k discs in any direction.
- Based on the hypothesis, we can move k+1 discs as follows:

1. Move k smaller discs in direction $\sim d$.
2. Move disc $k + 1$ in direction d.
3. Move k smaller discs in direction $\sim d$.

- For Step 1 and 3, in order to move k discs, we need to follow the above steps again that include the moving of k-1 discs, and so on ….. This is a recursive formulation of the solution.
- The subsequent slides will discuss the transformation of this recursive solution to an iterative solution.

The Towers of Hanoi – The Recursive Algorithm

Input An input tower of hanoi towerH with m disc to move to the tower at direction d
Output The output tower of hanoi towerH

```
towerHanoiR(towerH, m, d) {
        if (m = 0)
           do nothing
        else   {
           towerHanoiR(towerH, m-1, ~d)
           Move m - th disc in direction d.
           towerHanoiR(towerH, m-1, ~d)
        }
}
```


The Towers of Hanoi - transformation recursive to iterative solution

Let introduce some more notations:

- Let $H_{n,d}$
  (Hanoi solution for n discs in direction d)
  be the sequence of moves required to move the
  n smaller discs in direction d.
- Let $\langle k,d \rangle$ represent a single move of disc k in direction d.

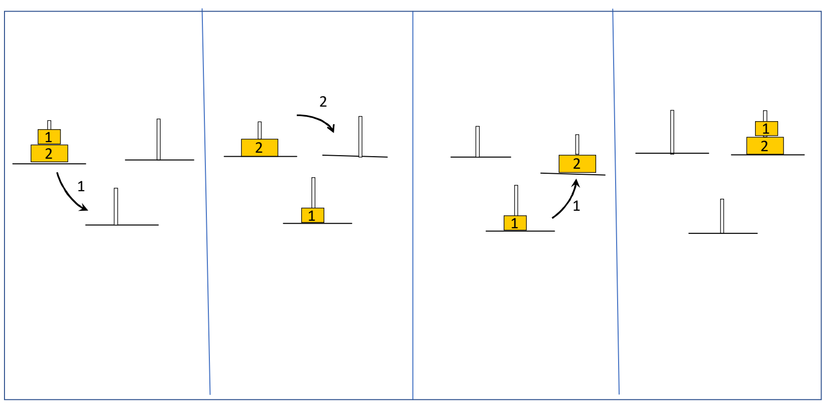
The Towers of Hanoi - transformation recursive to iterative solution

- $H_{0.d} = [\ ]$
- $H_{n+1.d} = H_{n.\neg d}\ ;\ \langle n+1, d\rangle\ ;\ H_{n.\neg d}$
- Note that H_{n+1} involves H_n.
  - This is what is known as a ***recursive*** formulation.
- For the direction we can use
  - c for clockwise and
  - a for anticlockwise.

The Solution $H_{2.c}$

- Now we can use the definitions to solve a sample problem.
- What is $H_{2.c}$?
  - $H_{2.c} = H_{1.a}\ ;\ \langle 2,c\rangle\ ;\ H_{1.a}$
  - $H_{1.a} = H_{0.c}\ ;\ \langle 1,a\rangle\ ;\ H_{0.c}$
  - $H_{0.c} = [\ ]$
- We can substitute upwards to give the following result:
  - $H_{0.c} = [\ ]$
  - $H_{1.a} = H_{0.c}\ ;\ \langle 1,a\rangle\ ;\ H_{0.c}$
  - $H_{1.a} = [\ ]\ ;\ \langle 1,a\rangle\ ;\ [\ ] = \langle 1,a\rangle$
  - $H_{2.c} = H_{1.a}\ ;\ \langle 2,c\rangle\ ;\ H_{1.a}$
  - $H_{2.c} = \langle 1,a\rangle\ ;\ \langle 2,c\rangle\ ;\ \langle 1,a\rangle$
- We can use this process to determine how to solve a problem of any size.

Diagrammatic View of $H_{2.c}$



The Solution $H_{3.c}$

What is $H_{3.c}$?

$H_{3.c} = H_{2.a}\ ;\ \langle 3,c\rangle\ ;\ H_{2.a}$

From previous slide:

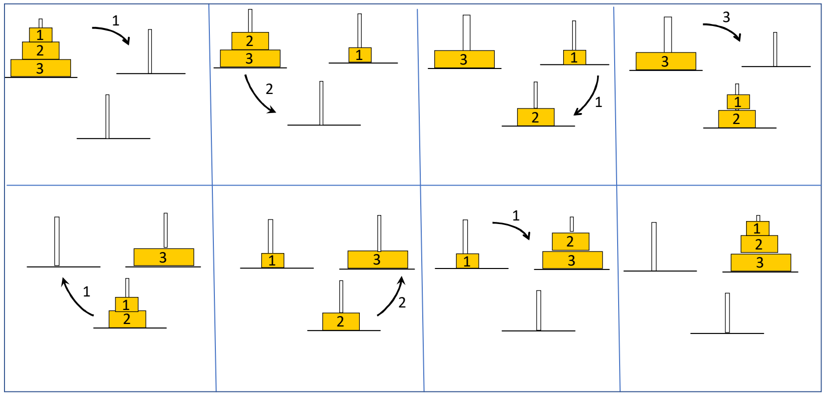
$H_{2.a} = \langle 1,c\rangle\ ;\ \langle 2,a\rangle\ ;\ \langle 1,c\rangle$

Hence:

$H_{3.c} = \langle 1,c\rangle\ ;\ \langle 2,a\rangle\ ;\ \langle 1,c\rangle\ ;\ \langle 3,c\rangle;\ \langle 1,c\rangle\ ;\ \langle 2,a\rangle\ ;\ \langle 1,c\rangle$

The next slide gives a diagrammatic of $H_{3.c}$

Study the Recursive Solution $H_{3.c}$ -- Diagram



The Towers of Hanoi - transformation recursive to iterative solution

- Please follow the same way to work out $H_{2.\neg c}$, $H_{3.\neg c}$, $H_{4.c}$, $H_{4.\neg c}$, $H_{5.c}$, $H_{5.\neg c}$, etc,. to identify the pattern of moves.
- If we examine the solution $H_{n.d}$ carefully, we will see that:
  - First, move the smallest disc (disc 1) in direction d if there are odd number of discs and in direction ¬d if otherwise.
  - The smallest disc (disc 1) always moves in alternate steps in the same direction (d for an odd number of discs and ¬d for an even number of discs).
- All that remains is to examine how to move the other disc alternatively with disc 1.
- Further examination is shown in the next few slides.

The Towers of Hanoi - transformation recursive to iterative solution

- After we move disc 1 we have three possible situations (**cases**):
  1. There are discs on each of the other two towers.
  2. There are discs on only one of the other two towers.
  3. There are no discs on either of the other two towers.
- Let us examine each of these in turn.

The Towers of Hanoi - transformation recursive to iterative solution

- **Case 1:** There are top discs on each of the other two towers.
  - Clearly one of the two towers must have the smaller of these two discs.
  - The only move that makes any sense is to move this disc on top of the larger disc.
  - Any other legal move requires us to move disc 1 again.

The Towers of Hanoi - transformation recursive to iterative solution

- **Case 2**: There are discs on only one of the other two towers.
  - Now, the only possible move that does not involve disc one is to move the other disc (smallest) to the empty tower.

The Towers of Hanoi - transformation recursive to iterative solution

- **Case 3**: There are no discs on either of the other two towers.
  - Now, no move is possible that does not involve moving disc 1.
  - But this is OK because …
  - …we have finished!

The Towers of Hanoi – Iterative Solution

- Perform the following two steps until all the discs have moved to the destination:

  1. Move the smallest disc (disc 1) in direction f (f = d for an odd number of discs and f = ¬d for an even number of discs).
  2. If it is possible to move another disc, move it.

- Note that the previous slides have shown that the move for the second step is unique.

The Towers of Hanoi – The Iterative Algorithm

- From the analysis, we design the following iterative algorithm:

```
towerHanoiI(inputTH, n, d) {
  while (not all the discs have moved to the destination) {
      Move the smallest disc (disc 1) in direction d if n is odd and ¬d if otherwise
      If (there are discs on each of the other two towers)
            move the smaller top disc on top of the bigger top disc
      If (there are discs on only one of the other two towers)
            move the top disc on this tower to the empty tower
      }
}
```

The Towers of Hanoi – A Summary

- There is a simple, elegant and uninformative solution designed from induction directly (recursive solution).
- There is a complicated, ugly but informative solution derived based on induction design (iterative solution).
- We can transform the first solution into the second solution by careful examination.

Problem 2 – Reversing the elements in an sequence

- Problem: Reverse the elements in a sequence A of n elements without using any temporary sequence.
- Example:

  input sequence:

1	5	77	888

  After reversing the elements:

888	77	5	1

- Example:

  input sequence:

1	5	7	888	9999

  After reversing the elements:

9999	888	7	5	1

Reversing the elements in a sequence – Recursive Algorithm

```
reverseR(A, f, l) {
// this is to reverse the numbers in A starting from index f and ending at index l
    if f < l  {
        swap A[f] with A[l]
        reverseR(A, f+1 , l-1)
    }
    else
        return A
    }
}
```


Reversing the elements in a sequence - transformation of recursive to iterative solution

Let introduce some more notations:

- Let $R_{i,j}$ be the sequence of swaps required to reverse the elements in a sequence A starting from index *i* and ending at index *j*.
- Let $\langle i, j\rangle$ represent the swap of A[*i*] with A[*j*].

Reversing the elements in a sequence - transformation recursive to iterative solution

- $R_{0.0} = [\ ]$
- $R_{0.n-1} = \langle 0, n\text{-}1 \rangle$; $R_{1.n-2}$
- Note that $R_{0.n-1}$ involves $R_{1.n-2}$.
  - This is what is known as a ***recursive*** formulation.

Reversing the elements in a sequence - transformation recursive to iterative solution

- Now we can use the definitions to solve some sample problems.
- We can substitute upwards to give the following results for some cases:
  - $R_{i.j} = [\ ]$ if $i \geq j$
  - $R_{0.1} = \langle 0, 1 \rangle$; $R_{1.0} = \langle 0, 1 \rangle$; [] = $\langle 0, 1 \rangle$
  - $R_{0.2} = \langle 0, 2 \rangle$; $R_{1.1} = \langle 0, 2 \rangle$; [] = $\langle 0, 2 \rangle$
  - $R_{0.3} = \langle 0, 3 \rangle$; $R_{1.2} = \langle 0, 3 \rangle$; $\langle 1, 2 \rangle$; $R_{2.1} = \langle 0, 3 \rangle$; $\langle 1, 2 \rangle$; [] = $\langle 0, 3 \rangle$; $\langle 1, 2 \rangle$
  - $R_{0.4} = \langle 0, 4 \rangle$; $R_{1.3} = \langle 0, 4 \rangle$; $\langle 1, 3 \rangle$; $R_{2.2} = \langle 0, 4 \rangle$; $\langle 1, 3 \rangle$; [] = $\langle 0, 4 \rangle$; $\langle 1, 3 \rangle$
  - $R_{0.5} = \langle 0, 5 \rangle$; $R_{1.4} = \langle 0, 5 \rangle$; $\langle 1, 4 \rangle$; $R_{2.3} = \langle 0, 5 \rangle$; $\langle 1, 4 \rangle$; $\langle 2, 3 \rangle$; $R_{3.2} = \langle 0, 5 \rangle$; $\langle 1, 4 \rangle$; $\langle 2, 3 \rangle$; [] = $\langle 0, 5 \rangle$; $\langle 1, 4 \rangle$; $\langle 2, 3 \rangle$
- From the results, we can see that we always do it as follows: first, we swap the first element with last element, second, swap the second element with the second element from the end, third, swap the third element with the third element from the end, and, so on.
- Furthermore, if the size of the sequence is odd, the central element will remain unchanged.

Reversing the elements in a sequence - The Iterative Algorithm

- From the analysis, we design the following iterative algorithm:

```
reverseI(A, f, l) {
// this is to reverse the elements in  a sequence A starting from index f and ending at index l

  while (f < l)     {
      swap A[f] with A[l]
      f = f + 1
      l = l – 1
  }
  return A
}
```

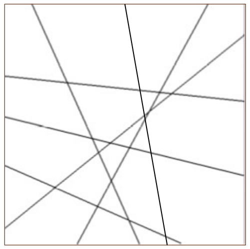

Q

Problem 3

- A number of straight lines are drawn across a sheet of paper, each line extending from one edge to another.
- In this way the paper is broken up into a number of regions.
- Show that we can colour the regions using just black and white in such a way that no two adjacent regions have the same colour.

Problem 3

- Here the size of the problem, n, is the number of lines that have been drawn.

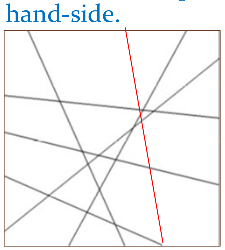


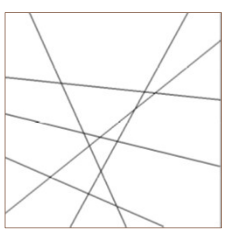
Problem 3

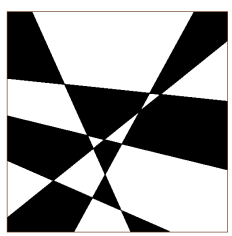
- The solution for $n = 0$ is trivial:
  - Either colour the paper all white;
  - Or colour the paper all black.
- For the inductive step we assume that we have a satisfactory colouring for a pattern of k (n=k) crossing lines.

Problem 3

- With the assumption in the inductive step, now we find a solution for size n = k+1 .
- We exclude one of the lines (the red colour line) in the problem of size n=k+1 (figure on the left), then the problem is of size n= k (figure in the middle).
- From the assumption, we have the solution shown in the figure on the right hand-side.

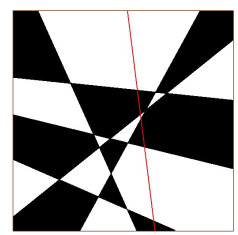






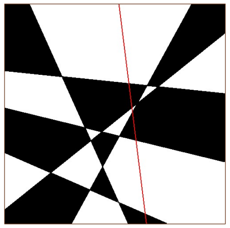
Problem 3

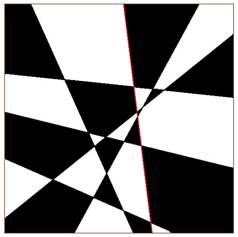
- We see that the regions on each side of the line excluded are correctly coloured for the problem of size n=k+1.
- But, the regions across the line excluded are the **same** colour.



Problem 3

- So we invert the colour of these regions.
- And, the figure on the right-hand-side is the solution for the original problem of size n = k+1.





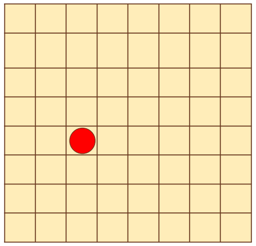
Problem 3

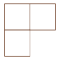
- We now have a solution for any number of lines:
  - Start with no lines.
  - Include lines one at a time until all the lines given in the problem are included.
  - Each time when we include a line flip the colours of all the regions on one side of the line included.
- Note that there are two possible solutions for any specific problem as we can colour the paper white or black when n = 0.

Problem 4

- A square board is divided into a 2^n x 2^n grid.
- One grid square is covered by a coin.
- A triomino is a shape consisting of three squares arranged in an "L" shape.
- Show that the remaining squares can be covered with triominoes without any overlap.

Problem 4





Problem 4

- The size here is related to the number of squares on the board.
- As each board has sides of length 2^n, n is the obvious choice for measuring the problem size.
- For $n = 0$ we have a 1x1 square which is covered by the coin – problem solved without the need of doing anything!

Problem 4

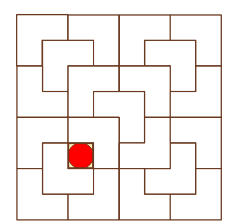
- Let us now consider a problem of size n = k + 1
- We can divide a 2^{k+1} square into 4 2^k sub-boards.
- One of these 4 sub-boards will contain the coin, so we can assume that this sub-board can be solved under inductive step.
- We can always place a single triomino so that one of its squares lies in each of the 3 empty squares. Hence, we can also assume that the other 3 2^k sub-boards can also be solved under the inductive step.
- That is, these three sub-boards now each has one grid cell covered and are now soluble in the same way as the sub-board with the coin under the assumption from the inductive step.

Problem 4

- This gives us a general strategy:
  - Divide the board into 4 equal sized squares;
  - place a single triomino so that it covers one cell in each empty board;
  - Repeat with each of the 4 sub-boards until the whole board is covered.

Problem 4

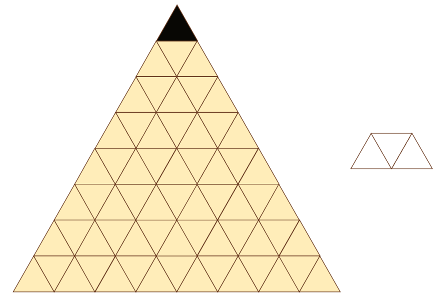
- Here is an example
  - Divide the board
  - Place a triomino in the centre
  - Divide each sub-board
  - Place triominoes
  - Divide again
  - Place triominoes
  - Done!



Problem 5

- An equilateral triangle with sides of length 2^n is made up of smaller triangles.
- The topmost triangle is covered by black colour.
- Show that it is possible to tile the remainder of the triangle with non-overlapping 3-triangle trapezoids.

Problem 5



Problem 5

- Once again, we can use the fact that the triangle has sides of length 2^n to define the problem size as n.
- For $n = 0$ the solution is trivial as we have a single triangle.
- So, it is covered by black colour - problem solved without the need of doing anything!

Problem 5

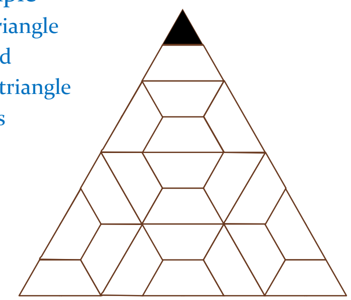
- Let us now consider a problem of size $n = k+1$
- We can divide a 2^{k+1} triangle into 4 2^k sub-triangles.
- One of these 4 triangles will have a black vertex and we can assume that this can be solved under the inductive step.
- We can always place a single trapezoid so that one of its triangles is at a vertex of each of the 3 empty triangles.

Problem 5

- These three triangles now each have one grid cell covered and are now soluble in the same way as the former triangle with the vertex covered under the assumption in the inductive step.
- This gives us a general strategy:
  - Divide the triangle into 4 equal sized triangle s;
  - place a single trapezoid so that it covers one vertex of each empty triangle;
  - Repeat with each of the 4 triangles until the whole board is covered.

Problem 5

- Here is an example
  - Subdivide the triangle
  - Place a trapezoid
  - Subdivide each triangle
  - Place trapezoids
  - Divide again
  - Place
  - Done!

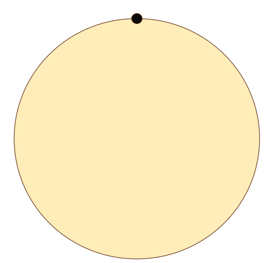


The Need for Proof

- To solve a problem, we may need to use formulas or patterns.
- Just because things look like they form a pattern doesn't mean that they really do.
- Consider the following question:
  - I mark n evenly spaced points around a circle and connect up all the points.
  - How many regions do I produce?
  - If we try this experiment we get a surprising result.

The Need for Proof

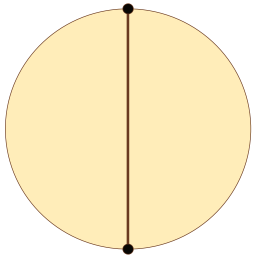
- $n = 1$
- $r = 1$



n	r
1	1

The Need for Proof

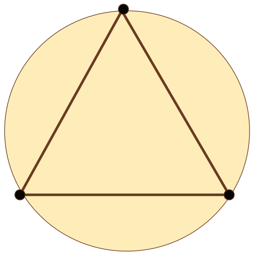
- $n = 2$
- $r = 2$



n	r
1	1
2	2

The Need for Proof

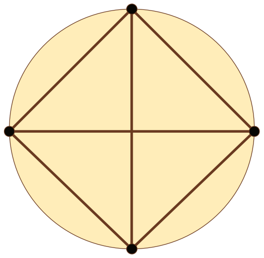
- $n = 3$
- $r = 4$



n	r
1	1
2	2
3	4

The Need for Proof

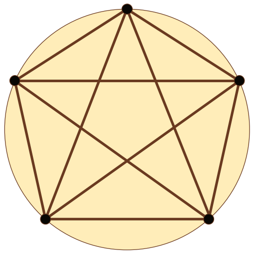
- $n = 4$
- $r = 8$



n	r
1	1
2	2
3	4
4	8

The Need for Proof

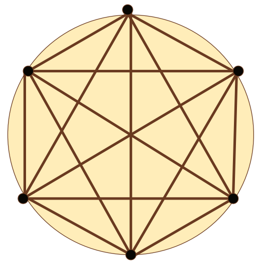
- $n = 5$
- $r = 16$



n	r
1	1
2	2
3	4
4	8
5	16

The Need for Proof

- $n = 6$
- $r = 30$



n	r
1	1
2	2
3	4
4	8
5	16
6	30

Initially, it gives an impression that $r = 2^{n-1}$. But, it is actually not. There is a need for proof for making conclusion.

Mathematical Induction

- **Mathematical induction** is a mathematical proof technique that uses Induction to solve the problem of forming mathematical proof.
- The method is also the same with using induction to solve problem except that in step 4, we form a proof instead of algorithm.
- It is commonly used to prove that a property $P(n)$ holds for infinite numbers of natural numbers, e.g., for each $n \geq 0$.
- It is commonly used in Computer Science.
- Next, we shall introduce some mathematical notations before giving some common examples on the property P(n).

Some Summation Notations

$$\sum_{i=1}^{n} i = 1+2+3+\cdots+n \qquad \sum_{i=1}^{4} i = 1+2+3+4$$

$$\sum_{i=1}^{n} i^2 = 1^2+2^2+3^2+\cdots+n^2 \qquad \sum_{i=1}^{3} i^2 = 1^2+2^2+3^2$$

a

$$\sum_{i=1}^{n} a = \underbrace{a+a+a+\cdots\ a}_{n\ times} \qquad \sum_{i=1}^{5} a = a+a+a+a+a$$

$$\sum_{i=p}^{q} a = \underbrace{a+a+a+\cdots\ a}_{q-p+1\ times} = (q-p+1)a \qquad \sum_{i=3}^{6} a = a+a+a+a$$

Example on Property P(n)

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} \text{ for all } n \geq 1$$

- when n=1, P(1) is $\sum_{i=1}^{1} i = \frac{1(1+1)}{2}$
- when n = 2, P(2) is $\sum_{i=1}^{2} i = \frac{2(2+1)}{2}$
- when n=3, P(3) is $\sum_{i=1}^{3} i = \frac{3(3+1)}{2}$

  ......

  ......
- when n=k, P(k) is $\sum_{i=1}^{k} i = \frac{k(k+1)}{2}$

  ......

  ......

Example on Property P(n)

$1 + 3 + 5 + 7 + \cdots + (2n-1) = n^2$

- when n=1, P(1) is $(2\times1 -1) = 1^2$, that is, $1 = 1^2$
- when n=2, P(2) is $(2\times1 -1) + (2\times2 -1) = 2^2$, that is, $1 + 3 = 2^2$
- when n=3, P(3) is $(2\times1 -1) + (2\times2 -1) + (2 \times3 -1) = 3^2$, that is, $1 + 3 +5 = 3^2$

  ......

  ......
- when n=k, P(k) is $1 + 3 + 5 + 7 + \cdots + (2k-1) = k^2$

  ......

  ......

Mathematical Induction

- Let P(n) be a statement defined for all natural numbers $n \geq n_o$.
- To prove P(n) holds using Mathematical Induction, there are two steps:
  - Basis Step: Prove that $P(n_o)$ holds.
  - Inductive Step: Assume that P(k) holds for $k \geq n_o$, and prove that P(k+1) holds
- If we can successfully complete these steps, then from the principle of mathematical induction, P(n) holds for all natural numbers $n \geq n_o$.

Example 1

Use mathematical induction to prove the following formula for all $n \geq 1$:

$$1 + 3 + 5 + 7 + \cdots + (2n-1) = n^2$$

Proof:

Basis Step: for n = 1

LHS = 1

RHS = 1^2= 1

Thus, LHS = RHS

Hence, the formula holds

Example 1

Inductive Step: Assume that the formula holds for n = k. That is, the following formula holds

$$1 + 3 + 5 + 7 + \cdots + (2k-1) = k^2$$

Then, for n = k+1:

$$\text{LHS} = 1 + 3 + 5 + 7 + \cdots + (2k - 1) + [2(k + 1) - 1]$$

$$= [1 + 3 + 5 + 7 + \cdots + (2k - 1)] + (2k + 2 - 1)$$

$$= k^2 + 2k + 1$$

$$= (k + 1)^2$$

$$\text{RHS} = (k + 1)^2$$

Thus, LHS = RHS

Hence, the formula holds

Therefore, from the principle of mathematical induction, the formula holds

Example 2

Use mathematical induction to prove the following formula for all n ≥ 1:

$$1^2 + 2^2 + 3^2 + 4^2 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6}$$

Proof:

Basis Step: for n = 1

LHS = 1^2 = 1

RHS = $\frac{1(2)(3)}{6} = 1$

Thus, LHS = RHS

Hence, the formula holds

Inductive Step: Assume that the formula holds for n = k. That is, the following formula holds

$$1^2 + 2^2 + 3^2 + 4^2 + \cdots + k^2 = \frac{k(k+1)(2k+1)}{6}$$

Example 2

Then, for n = k+1:

$$\textbf{LHS} = (1^2 + 2^2 + 3^2 + 4^2 + \cdots + k^2) + (k+1)^2$$

$$= \frac{k(k+1)(2k+1)}{6} + (k+1)^2$$

$$= \frac{k(k+1)(2k+1) + 6(k+1)^2}{6}$$ **Factor out a (k + 1)**

$$= \frac{(k+1)[k(2k+1) + 6(k+1)]}{6}$$

$$= \frac{(k+1)[2k^2 + 7k + 6]}{6} = \frac{(k+1)(k+2)(2k+3)}{6}$$

Example 2

$$\text{RHS} = \frac{(k+1)((k+1)+1)(2(k+1)+1)}{6}$$

$$= \frac{(k+1)(k+2)(2k+3)}{6}$$

Thus, LHS = RHS

Hence, the formula holds

Therefore, from the principle of mathematical induction, the formula holds

Another Proof for Example 2

Use mathematical induction to prove the following formula for all n ≥ 1:

$$1^2 + 2^2 + 3^2 + 4^2 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6}$$

Proof (in the second poof we use expansion instead of factorization):

Basis Step: for n = 1

LHS = 1^2 = 1

RHS = $\frac{1(2)(3)}{6}$ **= 1**

Thus, LHS = RHS

Hence, the formula holds

Inductive Step: Assume that the formula holds for n = k. That is, the following formula holds

$$1^2 + 2^2 + 3^2 + 4^2 + \cdots + k^2 = \frac{k(k+1)(2k+1)}{6}$$

Another Proof for Example 2

Then, for n = k+1:

LHS $= (\mathbf{1^2 + 2^2 + 3^2 + 4^2 + \cdots + k^2}) + (\mathbf{k+1})^2$

$$= \frac{k(k+1)(2k+1)}{6} + (k+1)^2$$

$$= \frac{k(k+1)(2k+1) + 6(k+1)^2}{6}$$ **Expand it**

$$= \frac{k(2k^2+3k+1) + 6(k^2+2k+1)}{6}$$

$$= \frac{(2k^3+3k^2+k) + (6k^2+12k+6)}{6}$$

$$= \frac{(2k^3+9k^2+13k+6)}{6}$$

Another Proof for Example 2

$$\text{RHS} = \frac{(k+1)((k+1)+1)(2(k+1)+1)}{6}$$ **Expand it**

$$= \frac{(k+1)(k+2)(2k+3)}{6}$$ **Expand it**

$$= \frac{(k^2+3k+2)(2k+3)}{6}$$

$$= \frac{(2k^3+6k^2+4k+3k^2+9k+6)}{6} = \frac{(2k^3+9k^2+13k+6)}{6}$$

Thus, LHS = RHS
Hence, the formula holds

Therefore, from the principle of mathematical induction, the formula holds