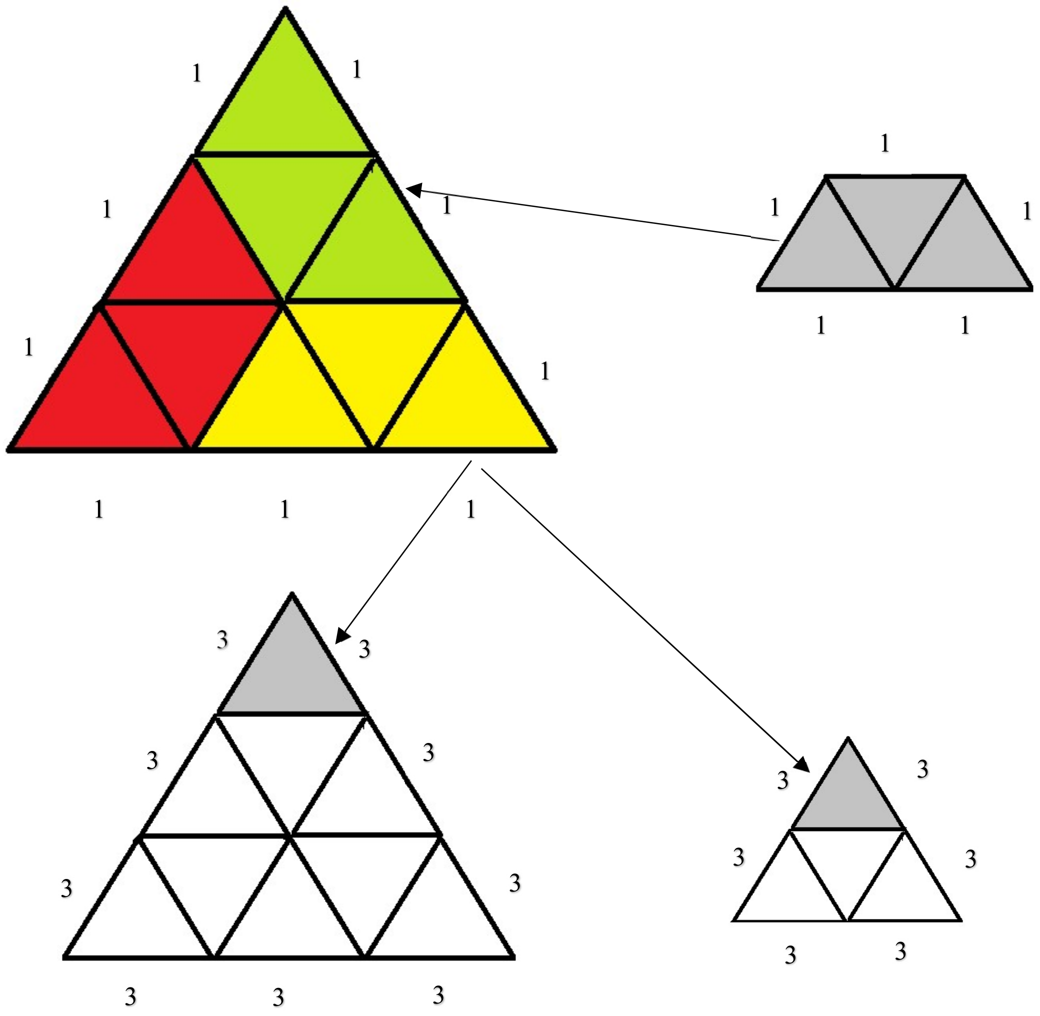
Part A

An equilateral triangle, with side of length 3n for some natural number n, is made of smaller equilateral triangles. See the figure below for the case n=2. A bucket-shaped trapezium shown in the right of the below figure is made from three equilateral triangles. Prove that it is possible to cover the remaining squares with non-overlapping trapeziums

Answer:

The problem needs to be solved through induction. If n=1, it is an equilateral triangle of length 3 on each side as shown in the figure below. The bucket shaped trapezium can then be used to cover the squares without overlapping as shown below. The three colors represent the three trapezoid used to cover the equilateral triangle. This is the base case triangle.



Assume now that n = 2, it would create an equilateral triangle of length 6 on each side as shown in the figure above (to the right). This larger triangle can then be broken down into 4 separate non overlapping base case triangles each of side 3. As we were able to fill the base case triangle with non-overlapping trapezoids, we will be able to fill the larger triangle as well.

For n= 3, it would create an equilateral triangle of length 9 on each side as shown in the figure above (to the left). This larger triangle can then be broken down into 9 separate non overlapping base case triangles each of side 3. As we were able to fill the base case triangle with trapezoids, we will be able to fill the larger triangle as well.

For n = 4, it would create an equilateral triangle of length 12 on each side. This larger triangle can be broken down into 4 smaller triangles of size 6 on each side. As we have shown earlier that a triangle of size 6 can be filled with non-overlapping trapezoid, so can this larger triangle.

This pattern continues, where for any natural number value of n, an equilateral triangle of length 3(n+1) can be broken down into separate non overlapping triangles. These triangles can consequently be derived from the base case triangle. As we have shown earlier that the base case has been filled with non-overlapping trapeziums, all subsequent triangles that are derived can also be filled with non-overlapping trapeziums as well.

A table showing how the value of n, the number of base case triangles, number of trapeziums and the number of triangles (of side length 1) are related to each other.

value of n	number of base case triangles	number of trapeziums	number of triangles
1	1	3	9
2	4	12	36
3	9	27	81
4	16	48	144
5	25	75	225
. . .	. . .	. . .	. . .

From the table able we are able to derive a formula for the number of trapezoids required for any value of n, shown below. We were also able to derive the formula for the number of base case triangles required for any value of n, shown below.

<u>Formula for the number of trapeziums:</u> $3n^2$

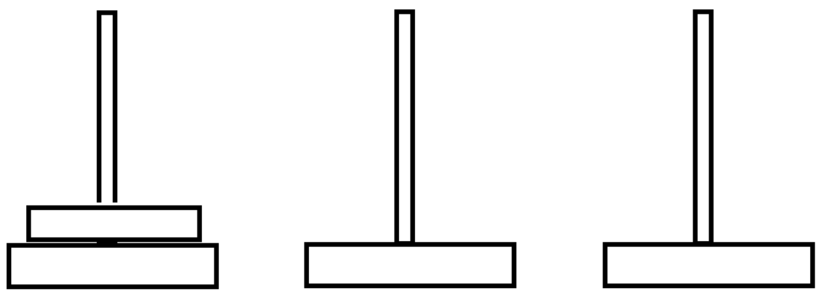
<u>Formula for the number of base case triangles:</u> n^2

Part B

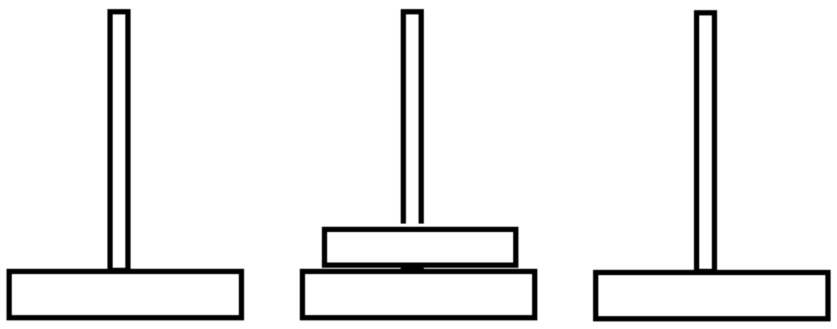
There are n disks of different sizes and three pegs. Initially, all the disks are on the first peg in order of size, the largest on the bottom and the smallest on top. The object is to transfer all the disks to the third peg. Only one disk can be moved at a time, and it is forbidden to place a larger disk on top of a smaller one. In addition, any move should either place a disk on the middle peg or move a disk from that peg. Design an algorithm that solves the problem in the minimum number of moves.

Answer

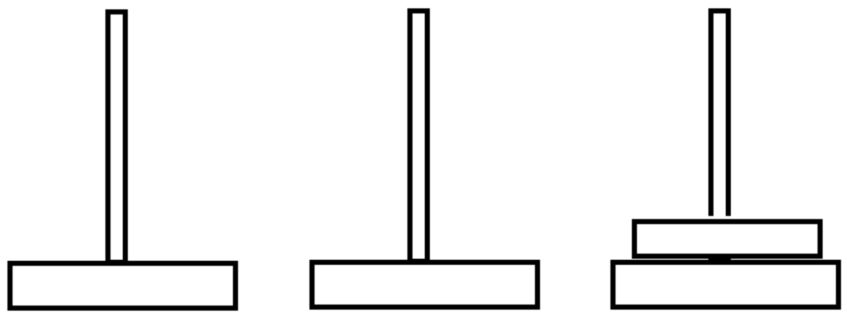
The problem needs to be solved through recursion. Let us consider the case when n=1 shown below, we move the only disk from the source peg to the auxiliary peg (middle peg) and then to the destination peg. Total number of moves required is 2.



MOVE = 0



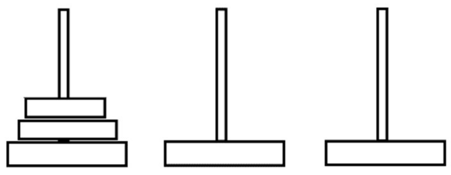
MOVE = 1



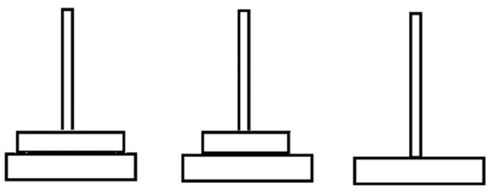
MOVE = 2

Let us consider the case when n=2 shown below, we move the top disk from the source peg to the auxiliary peg, and then from there to the destination peg. Then we move the bottom disk from the source peg to the auxiliary peg. We then move the top disk from the destination peg to the auxiliary peg (on top of the bottom disk) and then to the source peg. We then move the bottom disk from the auxiliary peg to the destination peg. Finally, we move the top disk from the source peg to the auxiliary peg and then to the destination peg (on top of the bottom disk). Total number of moves required is 8.

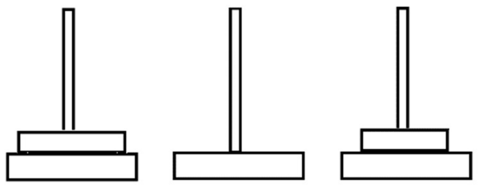
MOVE = 0



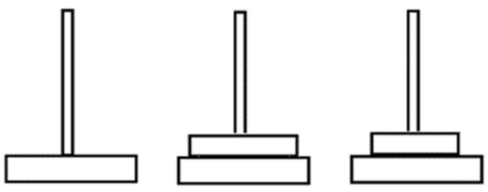
MOVE = 1



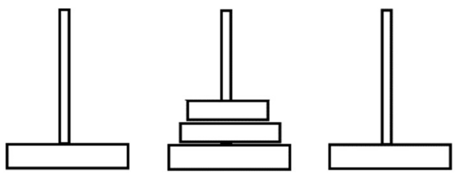
MOVE = 2



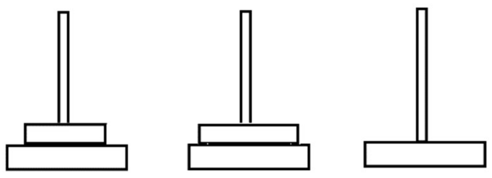
MOVE = 3



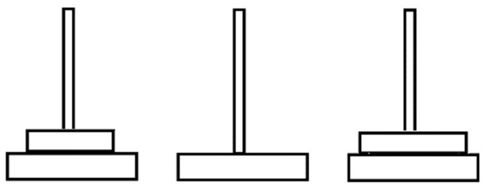
MOVE = 4



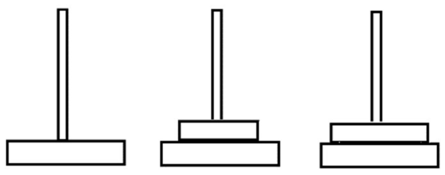
MOVE = 5



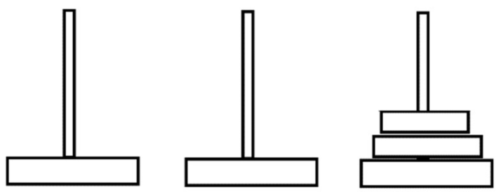
MOVE = 6



MOVE = 7



MOVE = 8



-

Let us now consider n=3, like n=2, we need to move the top number of disks which in this case is 2, from source peg to the destination peg via the auxiliary peg. However as there are more than 1 top disk the process needs to be done recursively following the same patten as in n=2. Then we need to move the bottom disk from the source peg to the auxiliary peg. Then we move the top number of disks recursively from the destination peg to the source peg via the auxiliary peg. We move the bottom disk from the auxiliary peg to the destination peg. Finally, we move the top number of disks recursively from the source peg to the destination peg. Total number of moves required is 26.

This process mentioned for n=3 can be repeated for any number of disks n, where we move top number of disks n-1 recursively along the pegs and bottom disk n in a standard way. Therefor the algorithm to solve for n disks is as follows:

Step 1: Solve recursively the problem of moving n-1 disks from source peg to destination peg via auxiliary peg.
Step 2: Move the nth disk (largest disk) from source peg to auxiliary peg.
Step 3: Solve recursively the problem of moving n-1 disks recursively from the destination peg to source peg via auxiliary peg.
Step 4: Move the nth disk from the auxiliary peg to the destination peg.
Step 5: Solve recursively the problem of moving n-1 disks recursively from the source peg to the destination peg via the auxiliary peg.

As we can see from the algorithm that the number of moves is larger by 2 moves (standard move) and 3 times the preceding number of moves (recursive moves).

Therefore, we can derive a recurrence relation for the number of moves G(n) as follows:

G(n) = 3G(n-1) + 2 where G(0) = 0 and all n is a natural number.

We use the recurrence relation derived earlier to find the first 5 value for G(n)

n	G(n)
1	2
2	8
3	26
4	80
5	242

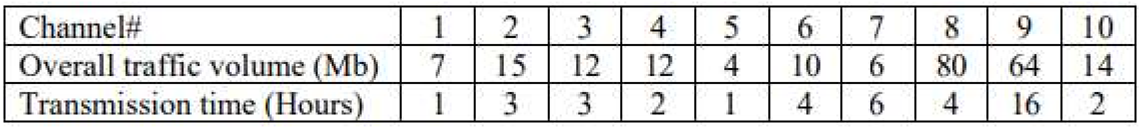
A formula can be derived from the table for the number of moves G(n) based on the value of n, the number of disks in the system.

$G(n) = 3^n - 1$,

this formula can further be verified by substituting into the recurrence formula mentioned earlier.

Part C

Eve, an eavesdropper, wants to capture as much communication traffic as possible. She can eavesdrop on a specific channel for at most two hours, contiguously or otherwise, or she will be detected. There are ten channels for her to choose from and she can eavesdrop for a total of 10 hours. Each channel cycles its transmission, but Eve will learn nothing by capturing the same traffic twice. The following table shows overall how much traffic is sent across a channel in the specified transmission time.



Channel#	1	2	3	4	5	6	7	8	9	10
Overall traffic volume (Mb)	7	15	12	12	4	10	6	80	64	14
Transmission time (Hours)	1	3	3	2	1	4	6	4	16	2

Answer

a) Describe how you can determine the greatest volume of traffic Eve can capture, given the constraints of the system.

- We would use the greedy method to capture the greatest volume of traffic. The greedy method involves selecting the best item first, then a worse one in succession. In the case of the table shown above, this would involve choosing the channel that performs the best first, followed by a worse one and then a worse one and so on.

To qualify the performance of a channel we will need to determine the volume per unit time for each one of the channels Therefore, the channel with the greatest volume per unit time would be captured first, followed by the second greatest and so on, up to the maximum timeframe of two hours. We also have to take into account that Eve will learn nothing new by capturing the same channel twice, so she has to move to the next best channel when the two hours are complete or till the maximum time of transmission has expired. She cannot go back to a channel she has eavesdrop on before, or else she will get caught. Collectively she can eavesdrop for only 10 hours.

b) Use your method of part a) to determine which channels Eve should eavesdrop on to maximise the traffic captured. Show the details of your working.

- Here is the table that that shows overall how much traffic is sent across a channel in the specified transmission line. We have added a third row below, showing how much volume is transmitted per unit time. We did this by dividing the volume by the time it took.

#	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Mb	7	15	12	12	4	10	6	80	64	14
Hrs	1	3	3	2	1	4	6	4	16	2
Mb/hr	7	5	4	6	4	2.5	1	20	4	7

By considering that Eve can only eavesdrop for two hours or till the transmission time has expired we have identified 6 distinct channels that would capture the greatest volume of traffic in the 10hr time frame.

1. Channel 8: 2hrs, collects 20 mb/hr * 2hr = 40mb

2. Channel 10: 2hrs, collects 7 mb/hr * 2hr = 14mb

3. Channel 1: 1hr, collects 7 mb/hr * 1hr = 7mb

4. Channel 4: 2hrs, collects 6 mb/hr * 2hr = 12mb

5. Channel 2: 2hrs, collects 5 mb/hr * 2hr = 10mb

6. Channel 5: 1hr, collects 4 mb/hr*1hr = 4mb

c) Specify the overall volume of traffic captured by Eve.

- Eve will eavesdrop for 10hrs and collect a total of 87mb of traffic