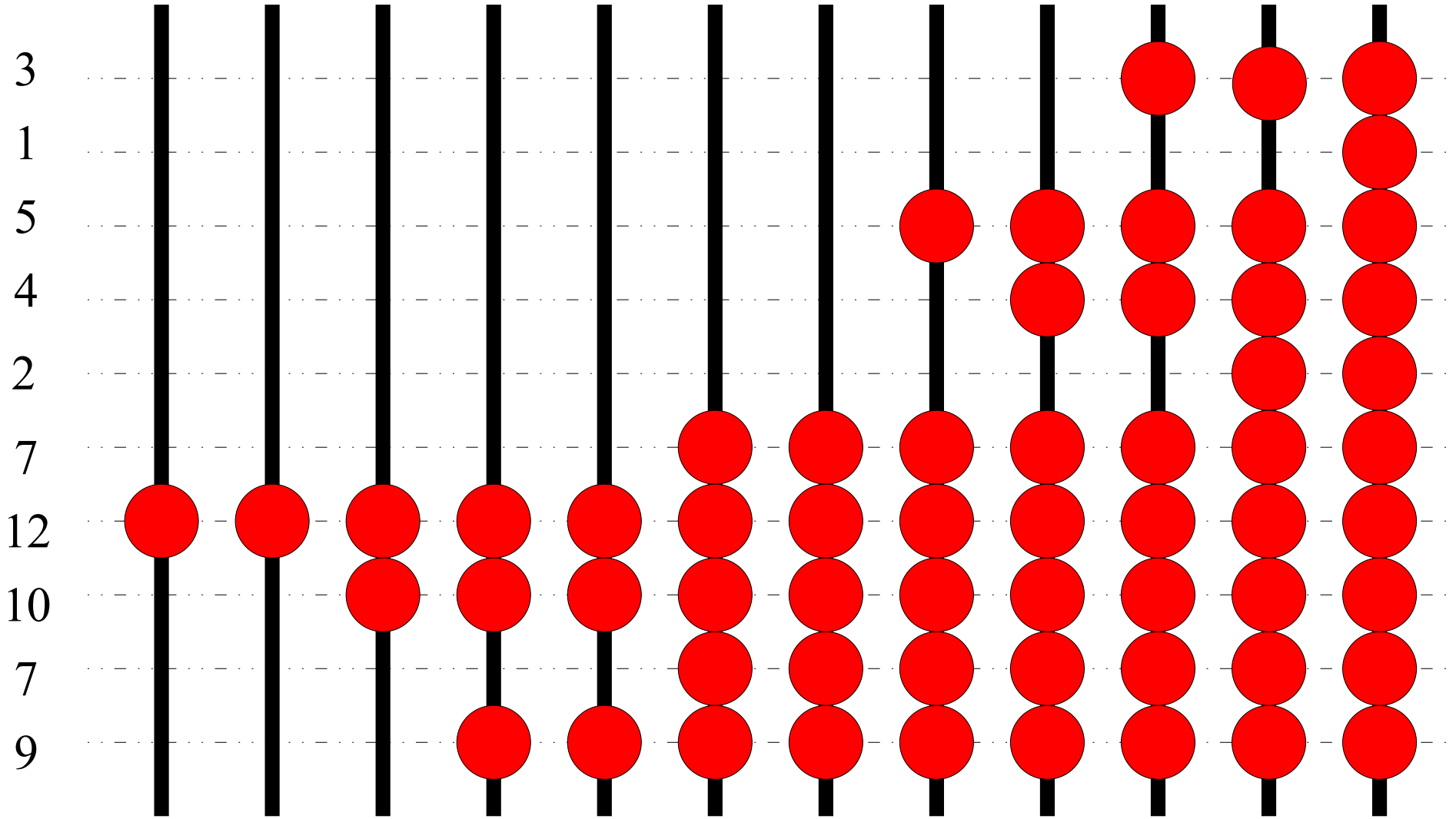
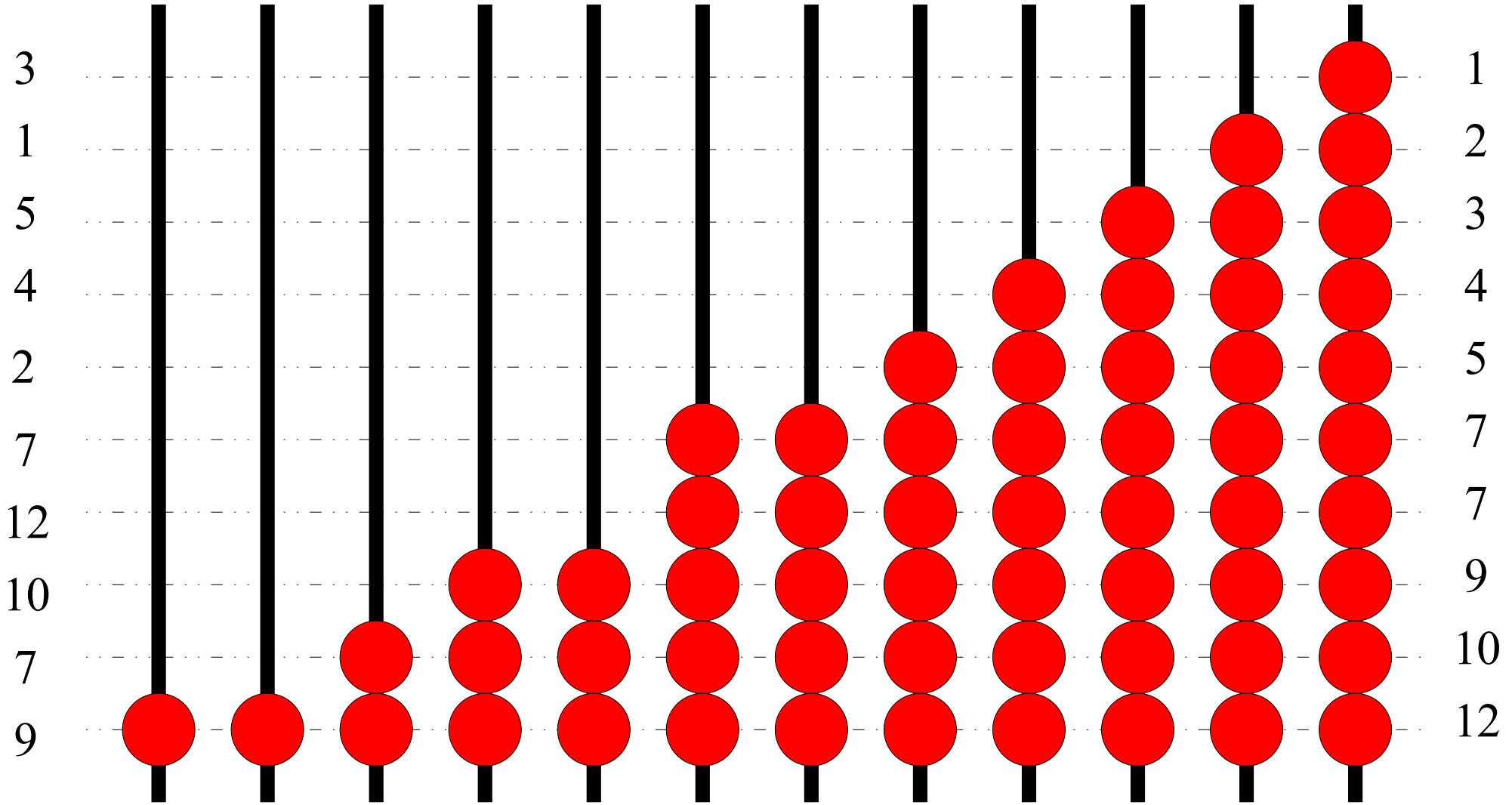
# Impractical Sorting Algorithms

# Outer Space Sort (OSS)

# Outer Space Sort (OSS)



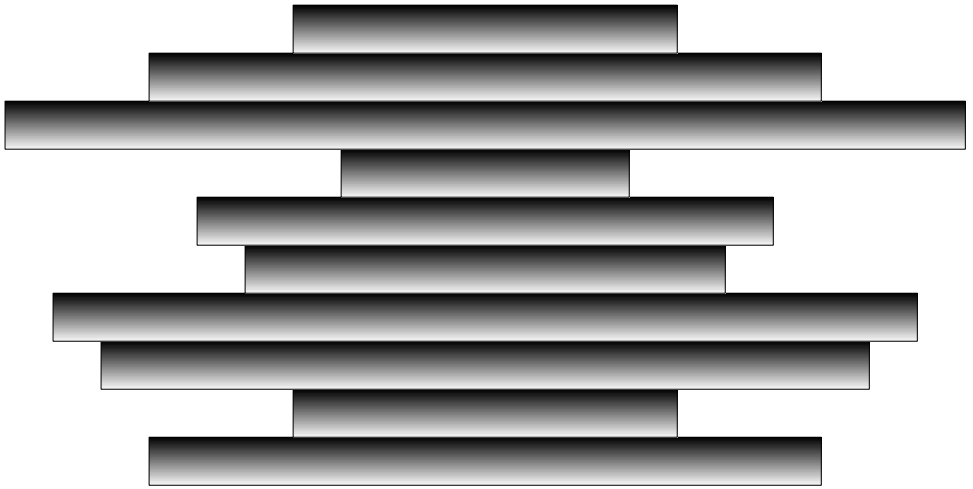
# Outer Space Sort (OSS)



# Pancake Sort

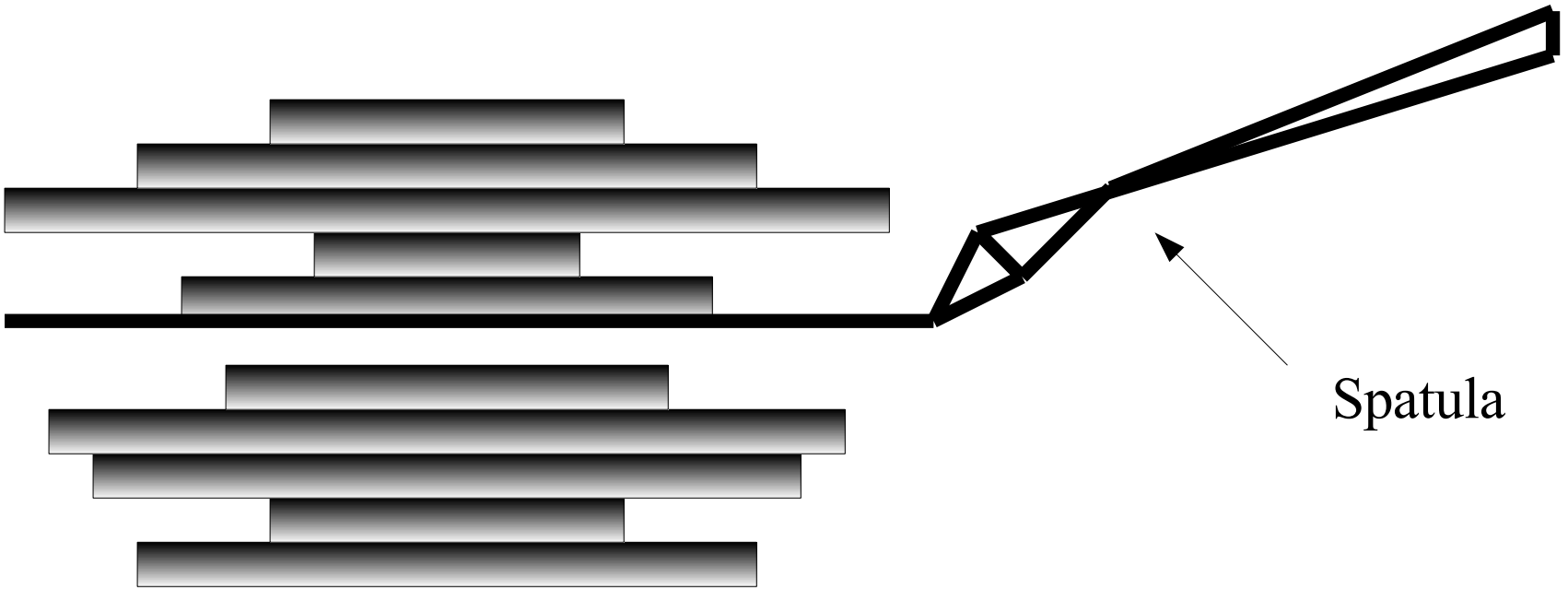
# Pancake Sort

Numbers are represented as pancakes – list of numbers as a stack of pancakes



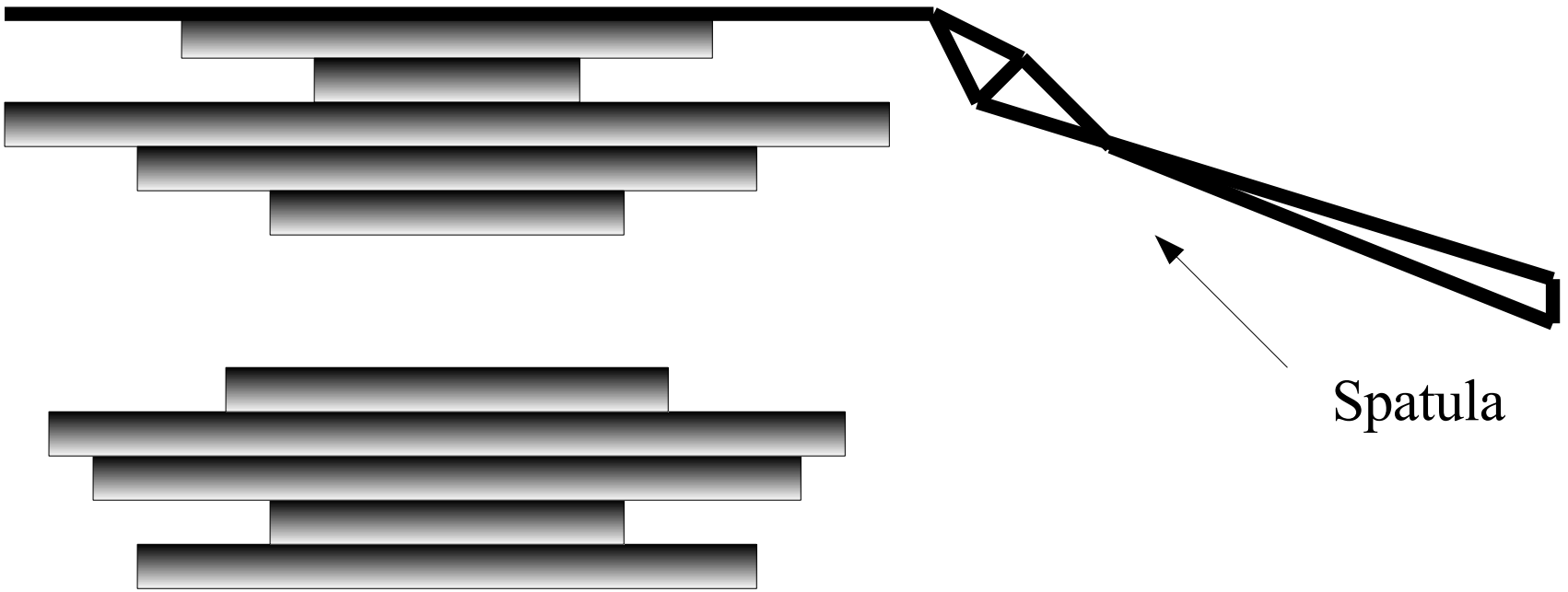
# Pancake Sort

Numbers are represented as pancakes – list of numbers as a stack of pancakes



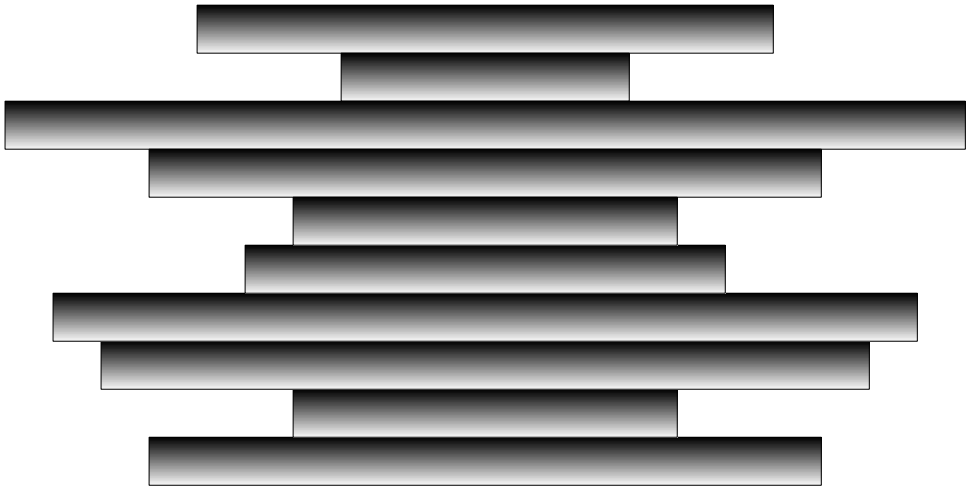
# Pancake Sort

Numbers are represented as pancakes – list of numbers as a stack of pancakes



# Pancake Sort

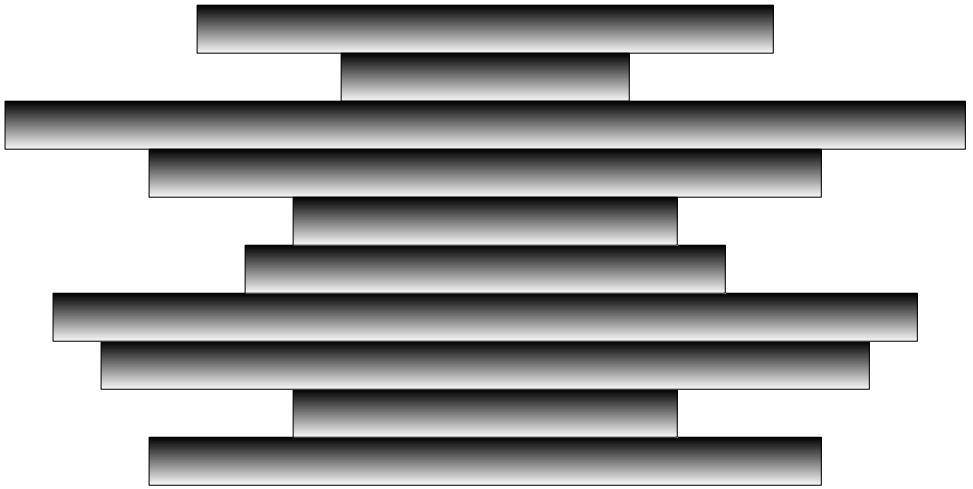
Numbers are represented as pancakes – list of numbers as a stack of pancakes



Only operation: flip the pancakes from inside the stack somewhere

# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom



# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom

# Pancake Sort

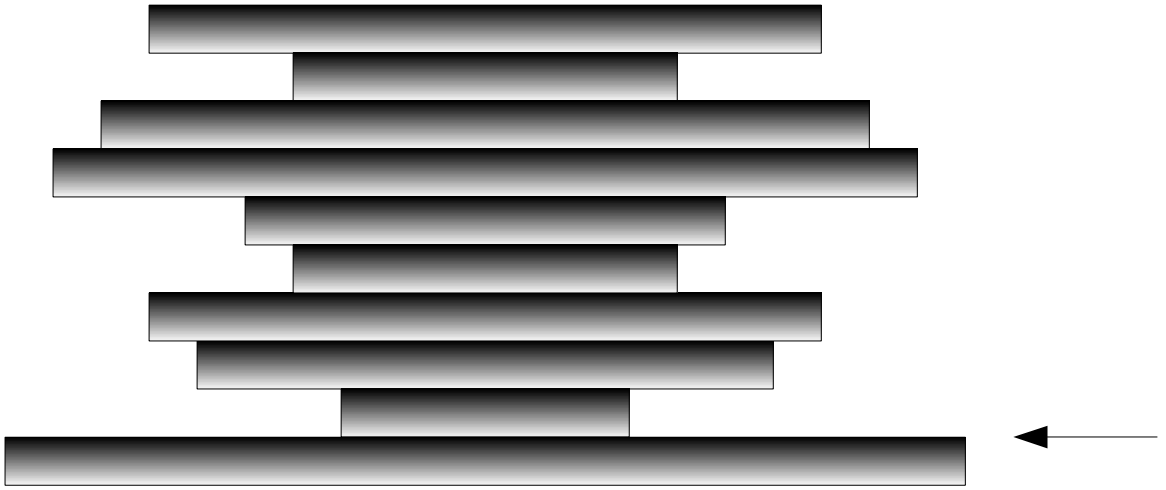
Algorithm: find largest that is not yet sorted, flip to top, flip to bottom

# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom

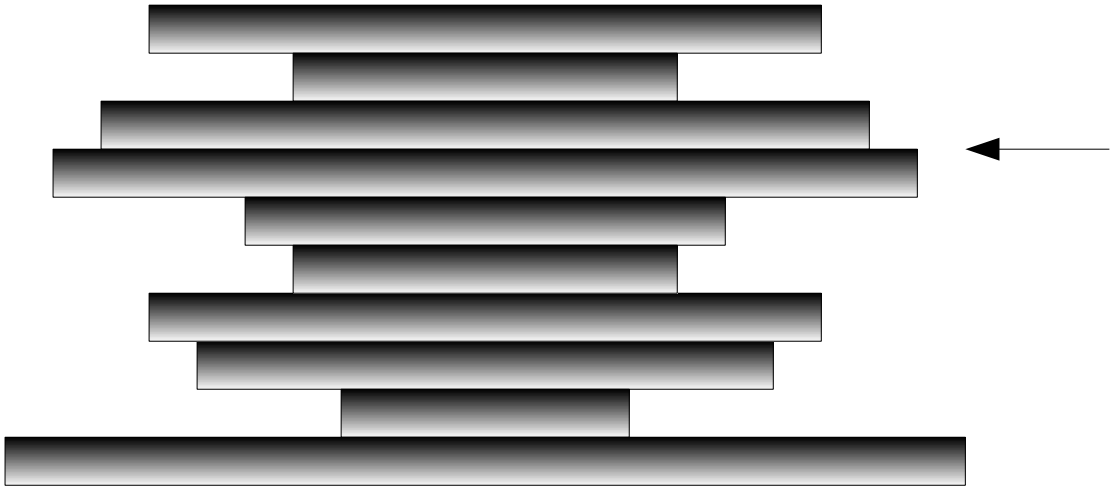
# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom



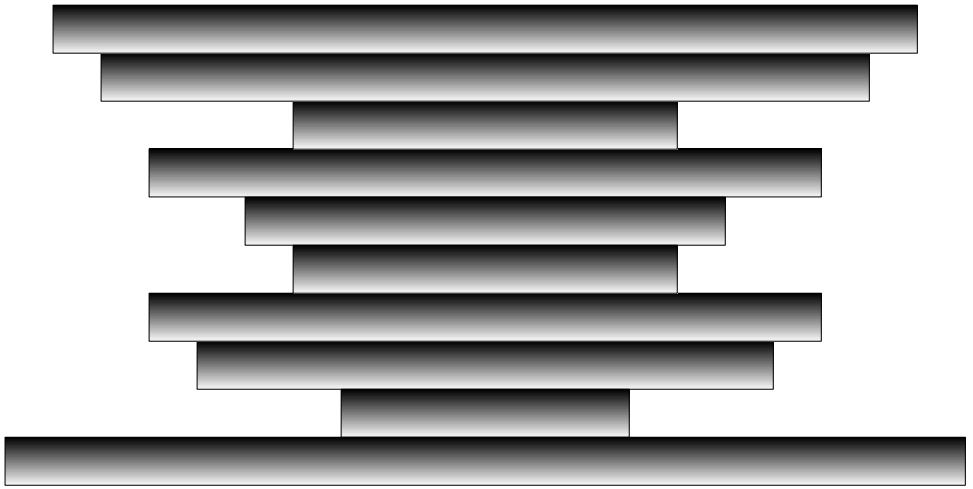
# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom



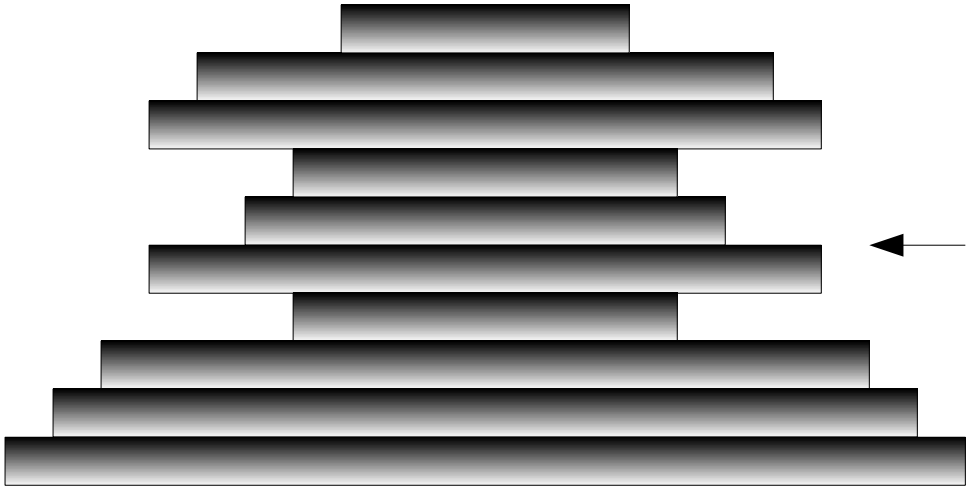
# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom



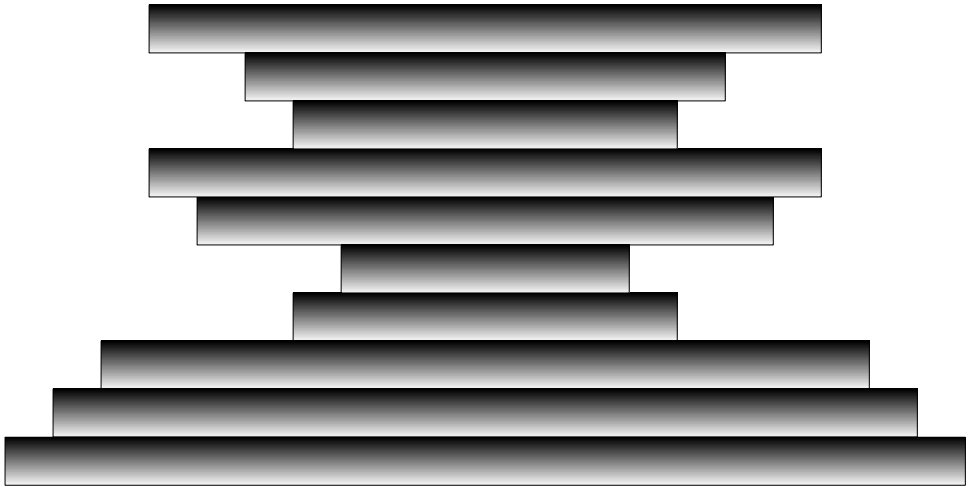
# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom



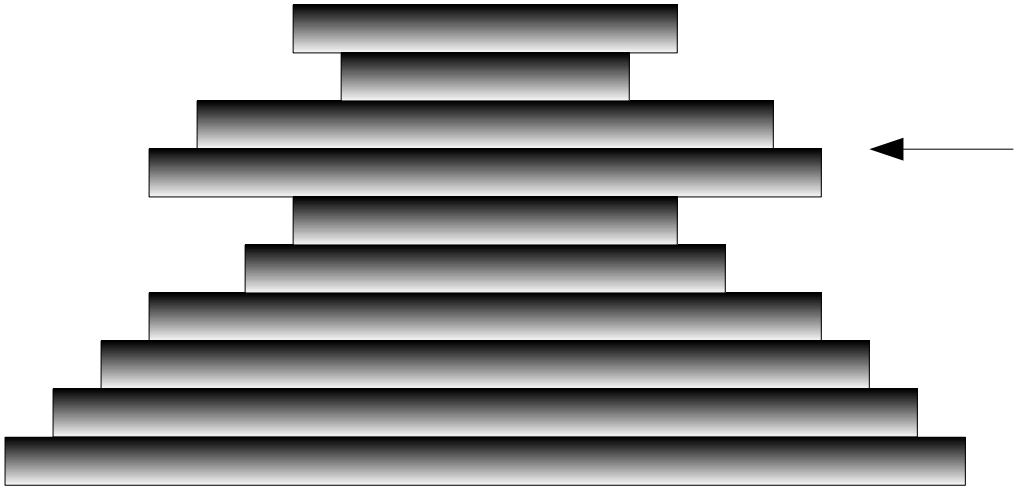
# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom



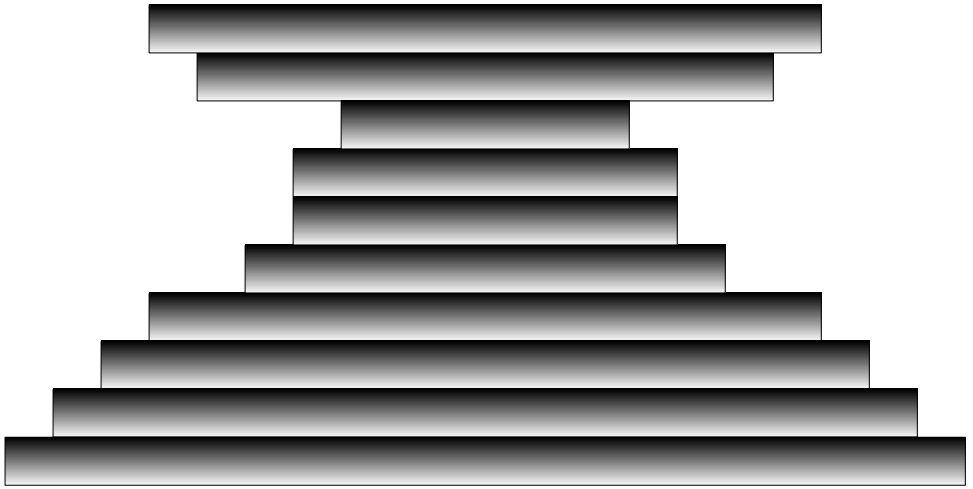
# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom



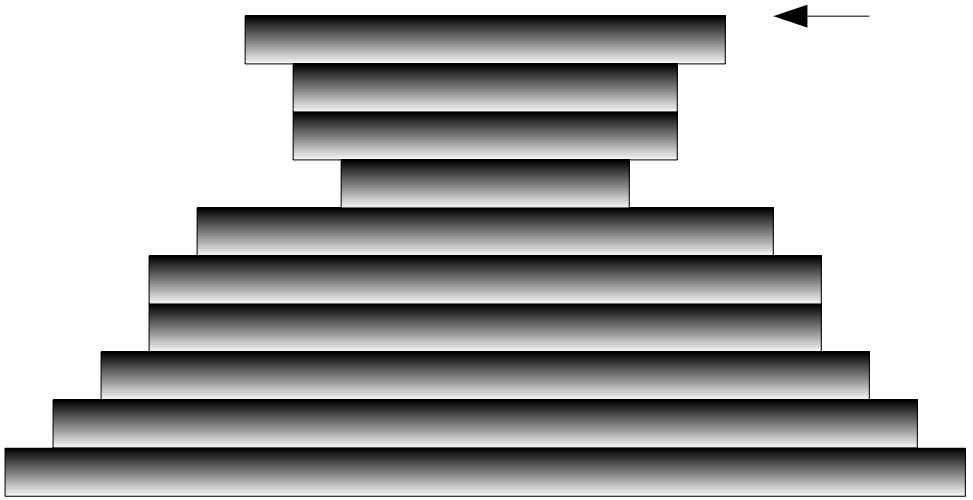
# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom



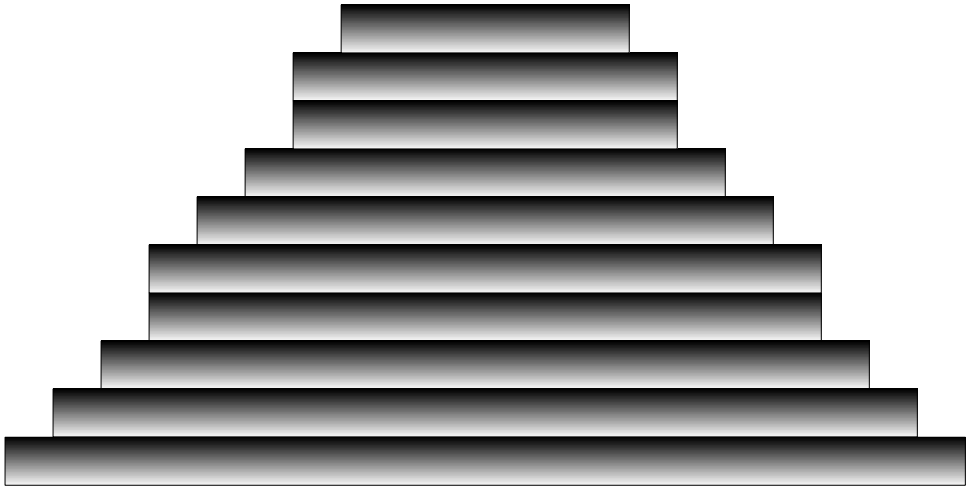
# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom



# Pancake Sort

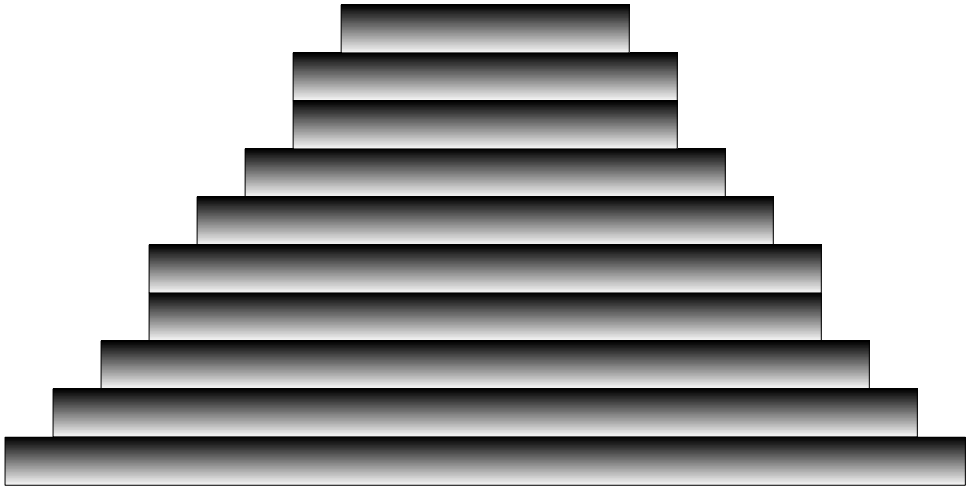
Algorithm: find largest that is not yet sorted, flip to top, flip to bottom



# Pancake Sort

Algorithm: find largest that is not yet sorted, flip to top, flip to bottom

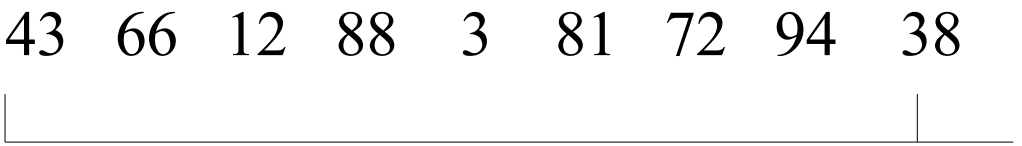
The *one and only* well-known published paper of Bill Gates describes an efficient pancake sort algorithm!



# Stooge Sort

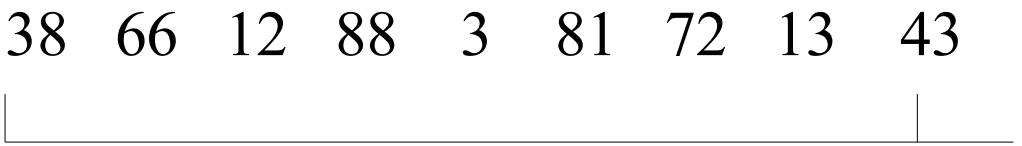
43 66 12 88 3 81 72 94 38

# Stooge Sort



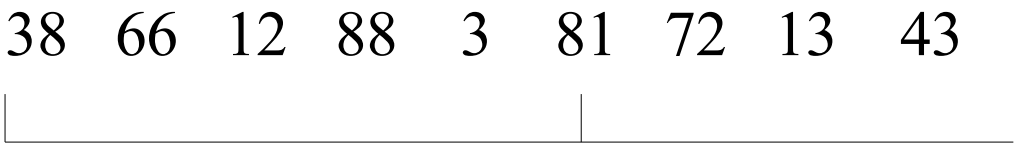

Compare last and first numbers and swap if out of order

# Stooge Sort



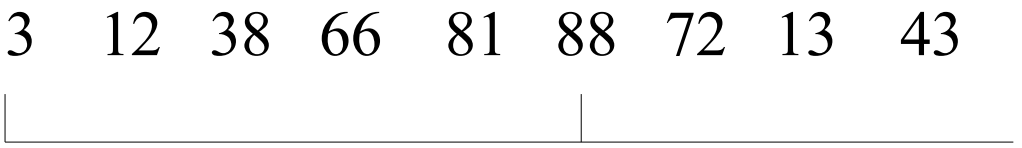

Compare last and first numbers and swap if out of order

# Stooge Sort



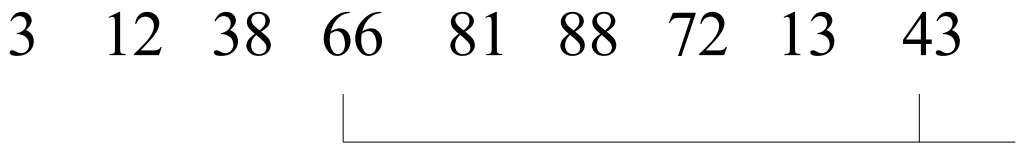

Sort the 1st two thirds of the list

# Stooge Sort



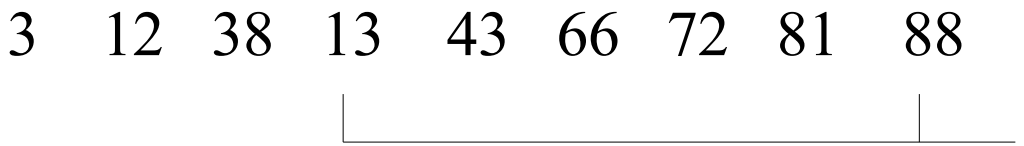

Sort the 1st two thirds of the list

# Stooge Sort



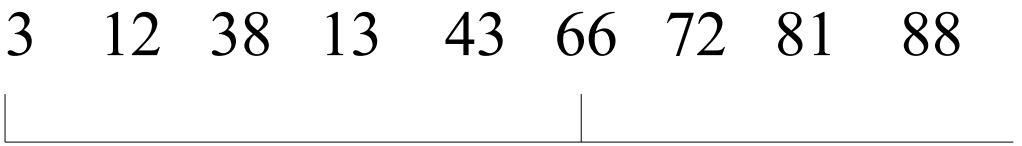

Sort the last two thirds of the list

# Stooge Sort



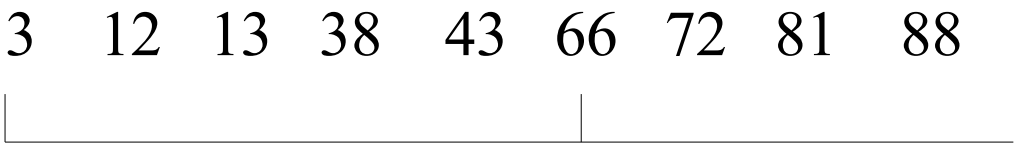

Sort the last two thirds of the list

# Stooge Sort




Sort the 1st two thirds of the list

# Stooge Sort




Sort the 1st two thirds of the list

# Bogo Sort

# Bogo Sort

**Given**: 9 23 8 92 3 45 7 16 88 19

# Bogo Sort

**Given**: 9 23 8 92 3 45 7 16 88 19

Roll dice

# Bogo Sort

**Given**: 9 23 8 92 3 45 7 16 88 19

Roll dice

7 8 88 19 16 45 92 9 23 3

# Bogo Sort

**Given**: 9 23 8 92 3 45 7 16 88 19

Roll dice

7 8 88 19 16 45 92 9 23 3

Average complexity:

Assume $n$ items to sort

Let $X$ be r.v. with value $i$ in the event it takes $i$ rolls to sort

$Pr(X=i) = (1-1/n!)^{i-1}(1/n!)$

$E\{X\} = Pr(X>0) + Pr(X>1) + Pr(X>2) + ...$

$E\{X\} = 1 + (1-1/n!) + (1-1/n!)^2 + (1-1/n!)^3 + ...$

$= 1/(1-(1-1/n!)) = n!$

# Digression

$$\mathrm{E}\{X\} = Pr(X=1) + 2Pr(X=2) + 3Pr(X=3) + \ldots$$

$$\begin{aligned}
\mathrm{E}\{X\} = Pr(X=1) + Pr(X=2) + Pr(X=3) + \ldots &= Pr(X>0) \\
+ Pr(X=2) + Pr(X=3) + \ldots &= Pr(X>1) \\
+ Pr(X=3) + \ldots &= Pr(X>2)
\end{aligned}$$