Counting

Ngày 16 tháng 11 năm 2012

Question

What do we count?

Question

What do we count?

Question

What do we count?

Question

Why do we count?

Question

What do we count?

Question

Why do we count?

Question

What do we count?

Question

Why do we count?

Question

How do we count?

Question

What do we count?

Question

Why do we count?

Question

How do we count?

Question

What do we count?

Question

Why do we count?

Question

How do we count?

To answer these questions we shall start practicing counting using common sense.

Question

What do we count?

Question

Why do we count?

Question

How do we count?

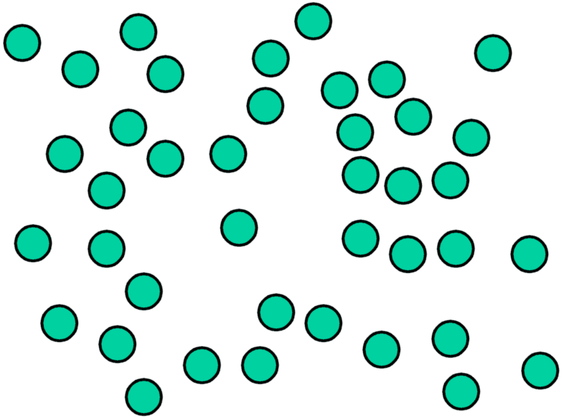
To answer these questions we shall start practicing counting using common sense.

A list of counting problems can be found in the file Let Us Count.pdf.

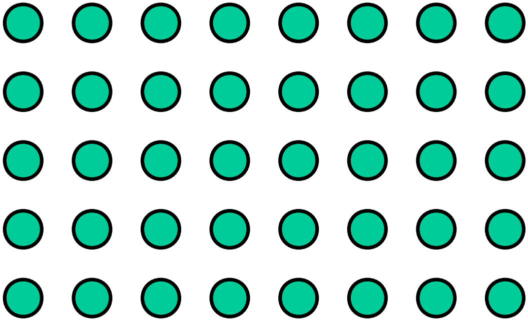
Làm thế nào nhiều trứng được vận chuyển trên các xe gắn máy trong ảnh?

How many students are attending this class?

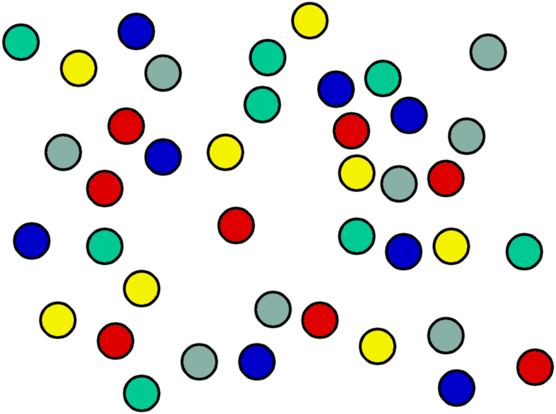
How many green disks are in this picture?



Can you count now?



And how about now?



1. Counting, especially of a large collection of objects, can be hard.

1. Counting, especially of a large collection of objects, can be hard.
2. If a collection can be “organized” (physically or conceptually, for example the “green” rectangular array) it can help us count the number of objects in the collection.

1. Counting, especially of a large collection of objects, can be hard.
2. If a collection can be “organized” (physically or conceptually, for example the “green” rectangular array) it can help us count the number of objects in the collection.
3. If the collection can be partitioned into “smaller” collections, in particular if every smaller collection has the same number of objects, it may again help us count.

Why count?

Why count?

You are a mathematician. Your friend, a programmer asks you:

Why count?

You are a mathematician. Your friend, a programmer asks you:

Question

I have an array with 10,000,000 integers. The weight of a segment $(a_i, a_{i+1} \ldots, a_j)$ *is* : $\sum_{i=0}^{j} a_i$.

I need to find the largest segment in the array.

Why count?

You are a mathematician. Your friend, a programmer asks you:

Question

I have an array with 10,000,000 integers. The weight of a segment $(a_i, a_{i+1} \ldots, a_j)$ *is :* $\sum_{i=0}^{j} a_i$.

I need to find the largest segment in the array.

Why count?

You are a mathematician. Your friend, a programmer asks you:

Question

I have an array with 10,000,000 integers. The weight of a segment $(a_i, a_{i+1} \ldots, a_j)$ *is* : $\sum_{i=0}^{j} a_i$.

I need to find the largest segment in the array.

This problem looks very simple to answer:

Why count?

You are a mathematician. Your friend, a programmer asks you:

Question

I have an array with 10,000,000 integers. The weight of a segment $(a_i, a_{i+1} \ldots, a_j)$ *is :* $\sum_{i=0}^{j} a_i$.

I need to find the largest segment in the array.

This problem looks very simple to answer:

1. Generate all segments.

Why count?

You are a mathematician. Your friend, a programmer asks you:

Question

I have an array with 10,000,000 integers. The weight of a segment $(a_i, a_{i+1} \ldots, a_j)$ *is :* $\sum_{i=0}^{j} a_i$.

I need to find the largest segment in the array.

This problem looks very simple to answer:

1. Generate all segments.
2. Calculate the weight of each segment.

Why count?

You are a mathematician. Your friend, a programmer asks you:

Question

I have an array with 10,000,000 integers. The weight of a segment $(a_i, a_{i+1} \ldots, a_j)$ *is* : $\sum_{i=0}^{j} a_i$.

I need to find the largest segment in the array.

This problem looks very simple to answer:

1. Generate all segments.
2. Calculate the weight of each segment.
3. Compare with the current largest weight, replace if the current weight is bigger.

Why count?

You are a mathematician. Your friend, a programmer asks you:

Question

I have an array with 10,000,000 integers. The weight of a segment $(a_i, a_{i+1}\ldots, a_j)$ *is :* $\sum_{i=0}^{j} a_i$.

I need to find the largest segment in the array.

This problem looks very simple to answer:

1. Generate all segments.
2. Calculate the weight of each segment.
3. Compare with the current largest weight, replace if the current weight is bigger.
4. Return the largest weight.

You get a phone call in the middle of the night. Your friend is on the line. He is panicked.

You get a phone call in the middle of the night. Your friend is on the line. He is panicked.

I implemented your solution and ran it. It started 30 hours ago and the computer is still running! What is wrong?

You get a phone call in the middle of the night. Your friend is on the line. He is panicked.

I implemented your solution and ran it. It started 30 hours ago and the computer is still running! What is wrong?

Hmmm, you decide to analyze your solution: count how many additions the computer is executing.

You get a phone call in the middle of the night. Your friend is on the line. He is panicked.

I implemented your solution and ran it. It started 30 hours ago and the computer is still running! What is wrong?

Hmmm, you decide to analyze your solution: count how many additions the computer is executing.

1. You have $n-1$ subarrays of length 2 $(a_1, a_2), (a_2, a_3), \ldots, (a_{n-1}, a_n)$ each requires one addition.

You get a phone call in the middle of the night. Your friend is on the line. He is panicked.

I implemented your solution and ran it. It started 30 hours ago and the computer is still running! What is wrong?

Hmmm, you decide to analyze your solution: count how many additions the computer is executing.

1. You have $n-1$ subarrays of length 2 $(a_1, a_2), (a_2, a_3), \ldots, (a_{n-1}, a_n)$ each requires one addition.
2. You have $n-2$ subarrays of length 3 $(a_1, a_2, a_3), (a_2, a_3, a_4), \ldots, (a_{n-2}, a_{n-1}, a_n)$ each requires 2 additions.

You get a phone call in the middle of the night. Your friend is on the line. He is panicked.

I implemented your solution and ran it. It started 30 hours ago and the computer is still running! What is wrong?

Hmmm, you decide to analyze your solution: count how many additions the computer is executing.

1. You have $n-1$ subarrays of length 2 $(a_1, a_2), (a_2, a_3), \ldots, (a_{n-1}, a_n)$ each requires one addition.
2. You have $n-2$ subarrays of length 3 $(a_1, a_2, a_3), (a_2, a_3, a_4), \ldots, (a_{n-2}, a_{n-1}, a_n)$ each requires 2 additions.
3. In general, you have $n-j$ subarrays of length $j+1$ each requires j additions.

You get a phone call in the middle of the night. Your friend is on the line. He is panicked.

I implemented your solution and ran it. It started 30 hours ago and the computer is still running! What is wrong?

Hmmm, you decide to analyze your solution: count how many additions the computer is executing.

1. You have $n-1$ subarrays of length 2 $(a_1, a_2), (a_2, a_3), \ldots, (a_{n-1}, a_n)$ each requires one addition.
2. You have $n-2$ subarrays of length 3 $(a_1, a_2, a_3), (a_2, a_3, a_4), \ldots, (a_{n-2}, a_{n-1}, a_n)$ each requires 2 additions.
3. In general, you have $n-j$ subarrays of length $j+1$ each requires j additions.
4. So the total number of additions required by this solution is:

You get a phone call in the middle of the night. Your friend is on the line. He is panicked.

I implemented your solution and ran it. It started 30 hours ago and the computer is still running! What is wrong?

Hmmm, you decide to analyze your solution: count how many additions the computer is executing.

1. You have $n-1$ subarrays of length 2 $(a_1, a_2), (a_2, a_3), \ldots, (a_{n-1}, a_n)$ each requires one addition.
2. You have $n-2$ subarrays of length 3 $(a_1, a_2, a_3), (a_2, a_3, a_4), \ldots, (a_{n-2}, a_{n-1}, a_n)$ each requires 2 additions.
3. In general, you have $n-j$ subarrays of length $j+1$ each requires j additions.
4. So the total number of additions required by this solution is:
5. $\sum_{i=1}^{n-1} j \times (n-j)$

You get a phone call in the middle of the night. Your friend is on the line. He is panicked.

I implemented your solution and ran it. It started 30 hours ago and the computer is still running! What is wrong?

Hmmm, you decide to analyze your solution: count how many additions the computer is executing.

1. You have $n-1$ subarrays of length 2 $(a_1, a_2), (a_2, a_3), \ldots, (a_{n-1}, a_n)$ each requires one addition.
2. You have $n-2$ subarrays of length 3 $(a_1, a_2, a_3), (a_2, a_3, a_4), \ldots, (a_{n-2}, a_{n-1}, a_n)$ each requires 2 additions.
3. In general, you have $n-j$ subarrays of length $j+1$ each requires j additions.
4. So the total number of additions required by this solution is:
5. $\sum_{i=1}^{n-1} j \times (n-j)$
6. Calculate: $\sum_{i=1}^{n-1} j \times (n-j) = \frac{1}{6}(n^3 - n)$

Your friend tells you that his computer can execute 10,000,000,000 (10^{10}) additions per second.

Your friend tells you that his computer can execute 10,000,000,000 (10^{10}) additions per second.

The array has 10,000,000 (10^7) integers.

Your friend tells you that his computer can execute 10,000,000,000 (10^{10}) additions per second.

The array has 10,000,000 (10^7) integers.

So we need to execute $\frac{1}{6}(10^{21} - 10^7) > 10^{20}$ additions.

Your friend tells you that his computer can execute 10,000,000,000 (10^{10}) additions per second.

The array has 10,000,000 (10^7) integers.

So we need to execute $\frac{1}{6}(10^{21} - 10^7) > 10^{20}$ additions.

So the computer will need more than 10^{10} seconds.

Your friend tells you that his computer can execute 10,000,000,000 (10^{10}) additions per second.

The array has 10,000,000 (10^7) integers.

So we need to execute $\frac{1}{6}(10^{21} - 10^7) > 10^{20}$ additions.

So the computer will need more than 10^{10} seconds.

Question

How long is 10^{10} seconds?

Your friend tells you that his computer can execute 10,000,000,000 (10^{10}) additions per second.

The array has 10,000,000 (10^7) integers.

So we need to execute $\frac{1}{6}(10^{21} - 10^7) > 10^{20}$ additions.

So the computer will need more than 10^{10} seconds.

Question

How long is 10^{10} seconds?

Your friend tells you that his computer can execute 10,000,000,000 (10^{10}) additions per second.

The array has 10,000,000 (10^7) integers.

So we need to execute $\frac{1}{6}(10^{21} - 10^7) > 10^{20}$ additions.

So the computer will need more than 10^{10} seconds.

Question

How long is 10^{10} seconds?

There are 86400 seconds per day. So your friend's computer will run about: $\frac{10^{10}}{864000}$ seconds. Which is:

Your friend tells you that his computer can execute 10,000,000,000 (10^{10}) additions per second.

The array has 10,000,000 (10^7) integers.

So we need to execute $\frac{1}{6}(10^{21} - 10^7) > 10^{20}$ additions.

So the computer will need more than 10^{10} seconds.

Question

How long is 10^{10} seconds?

There are 86400 seconds per day. So your friend's computer will run about: $\frac{10^{10}}{864000}$ seconds. Which is:

MORE THAN 27 YEARS!

So now you know why you need to count!

So what are you going to do next?

So now you know why you need to count!

So what are you going to do next?

Tell your friend to buy a faster computer?

So now you know why you need to count!

So what are you going to do next?

Tell your friend to buy a faster computer?

Or design a faster algorithm (an algorithm that executes a lot less additions).

So now you know why you need to count!

So what are you going to do next?

Tell your friend to buy a faster computer?

Or design a faster algorithm (an algorithm that executes a lot less additions).

Can it be done?

So now you know why you need to count!

So what are you going to do next?

Tell your friend to buy a faster computer?

Or design a faster algorithm (an algorithm that executes a lot less additions).

Can it be done?

ABSOLUTELY. ACTUALLY AN ALGORITHM THAT WILL SOLVE THE SAME PROBLEM IN A FEW SECONDS CAN BE DESIGNED

So now you know why you need to count!

So what are you going to do next?

Tell your friend to buy a faster computer?

Or design a faster algorithm (an algorithm that executes a lot less additions).

Can it be done?

ABSOLUTELY. ACTUALLY AN ALGORITHM THAT WILL SOLVE THE SAME PROBLEM IN A FEW SECONDS CAN BE DESIGNED

SO NOW YOU KNOW WHY WE NEED TO LEARN HOW TO COUNT!