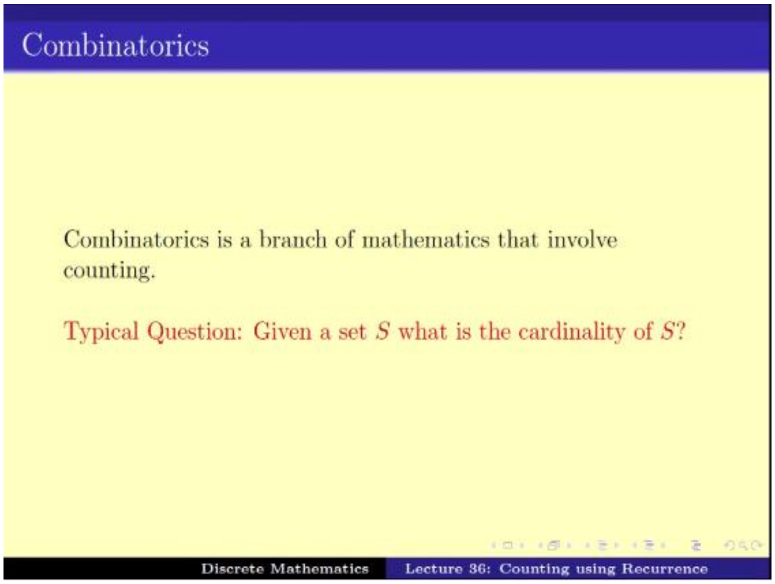
Discrete Mathematics
Prof. Sourav Chakraborty
Department of Mathematics
Indian Institute of Technology – Madras

Lecture - 36
Counting using Recurrence Relations (Part 1)

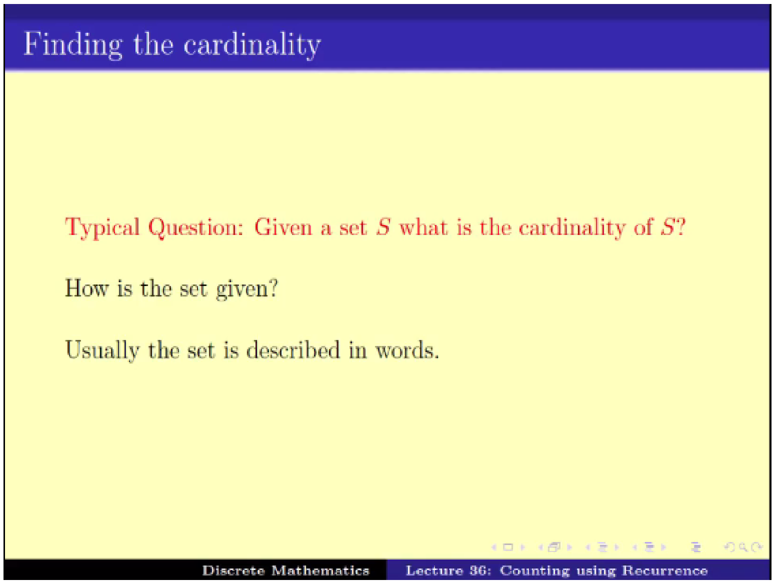
Welcome Back. So we have been looking at counting problems.

(Refer Slide Time: 00:07)



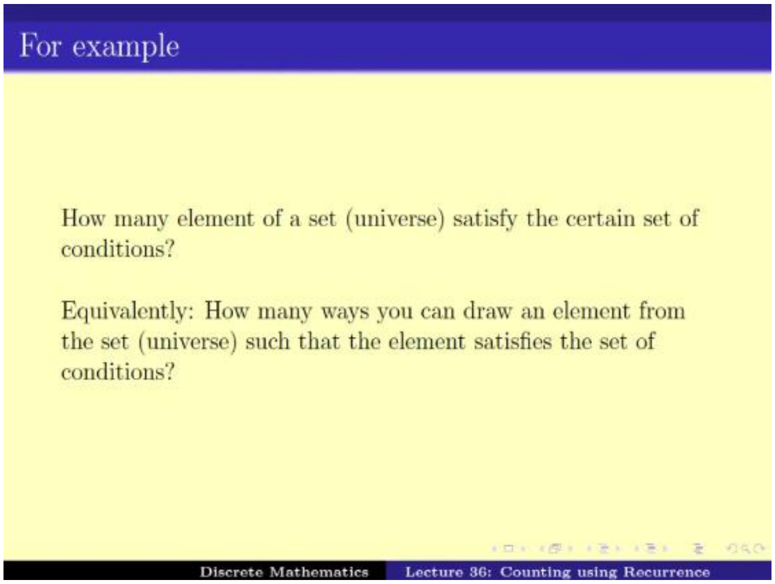
In particular, we have been looking at problems in Combinatorics which is a branch of mathematics that involve counting. So the typical question that one ask is, given a set what is the cardinality of the set S.

(Refer Slide Time: 00:28)



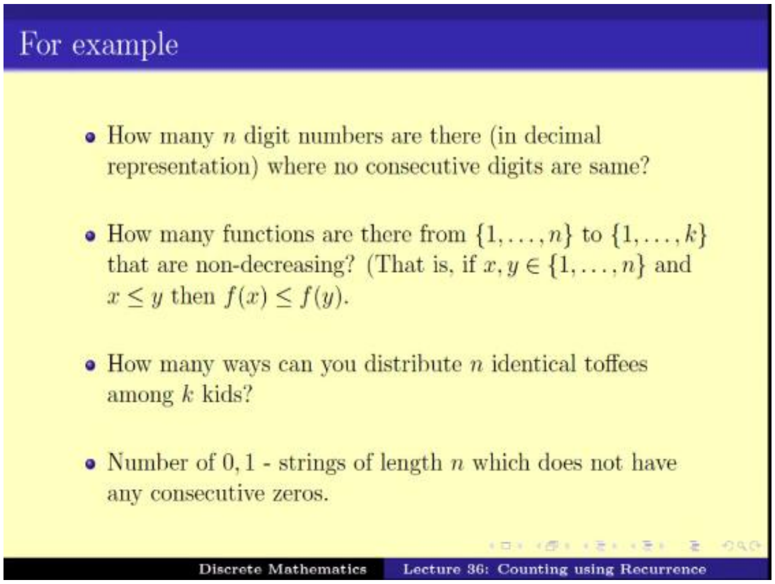
Now the question is that how is the set given. Of course is the set is given explicitly till counting number of elements in math is quite simple. But most of the time the set is described as words or described in some other way. And in that case the set is basically given implicitly and counting the number of element in the set is a pretty challenging problem.

(Refer Slide Time: 01:02)



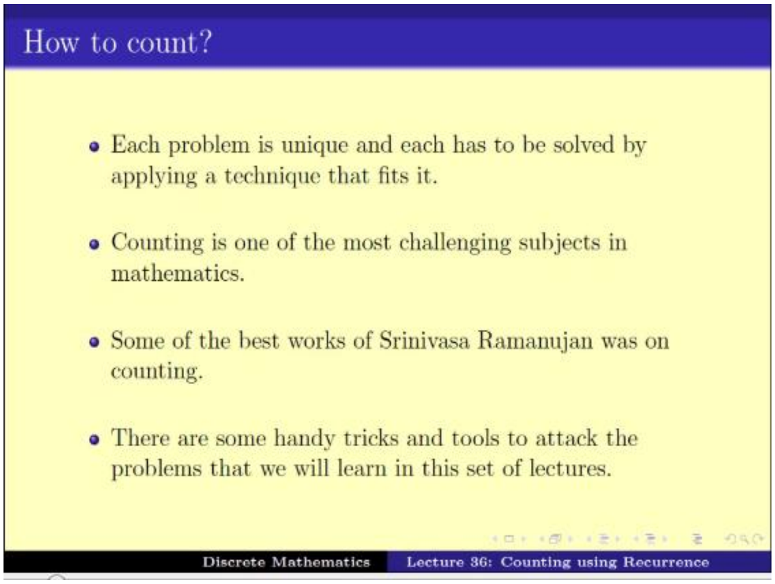
For example, how many elements are there satisfying a particular set of conditions or equivalently how many ways can you draw an element from a set of universe satisfying a certain set of conditions. So these are the kind of questions that usually we will look at.

(Refer Slide Time: 01:25)



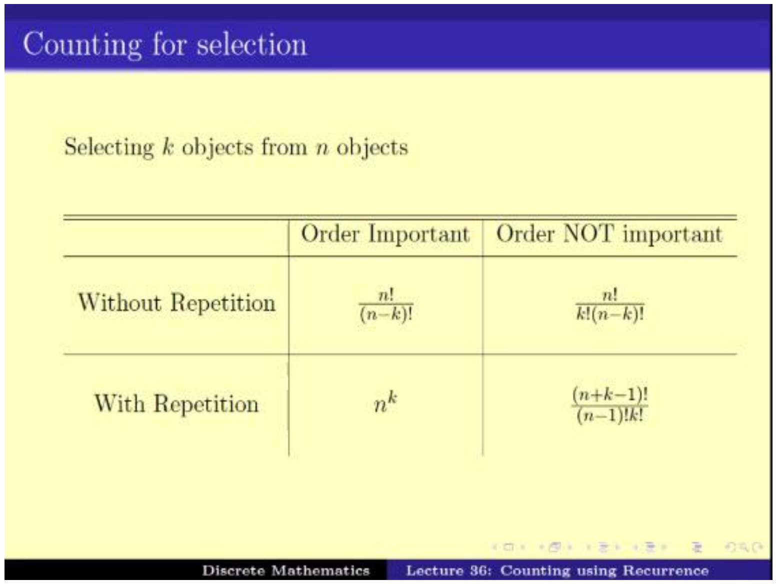
So we have been looking at a few examples, for example how many n digit numbers are there where no consecutive digits are the same. How many non-decreasing functions are there from one to n to one to k. How can you distribute n identical toffees among k kids. And the finally how many 0,1 strings of their of length of n which does not have any consecutive zero.

(Refer Slide Time: 02:03)



Now, the problem which counting is that every problem is unique and it requires a different technique to solve it. It is one of the most challenging subject in mathematics from big name like Srinivasan Ramanujan is also work on counting. And there are some handy tricks and tools to attack but they are just some kind of a small tools it does not exactly give a standard way of solving all the problems.

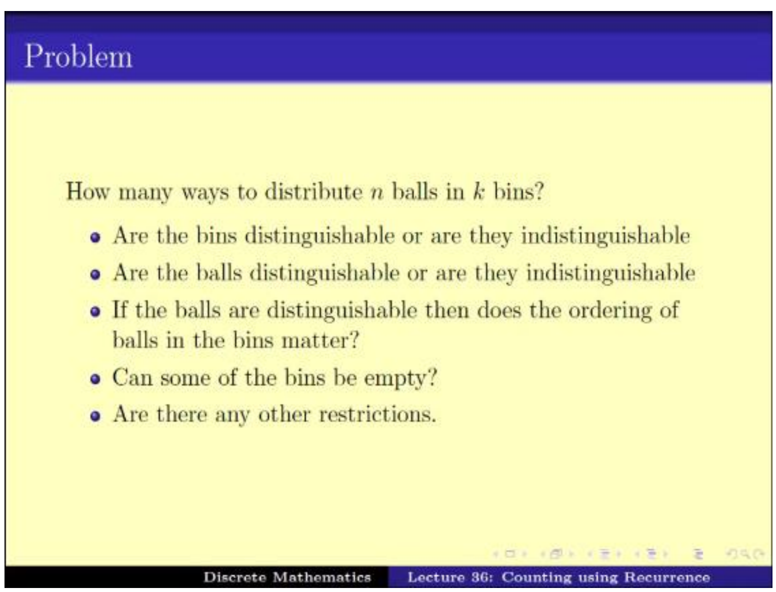
(Refer Slide Time: 02:42)



So once the tools are fixed was this particular special case where we call see how many ways can we select let k objects from n objects. So there are two different cases that we have to take in number one is whether the objects are – whether repetitions are allowed. In other word, can I pick a same object from the n objects multiple times? And the second case that we should look at is whether the ordering which the elements are picked matter.

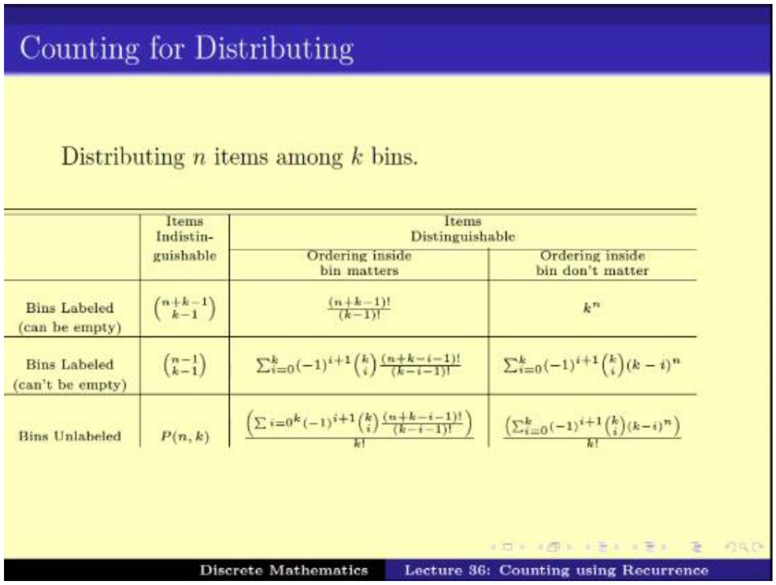
And this gives us the four cases and we have seen how to solve these four cases.

(Refer Slide Time: 03:35)



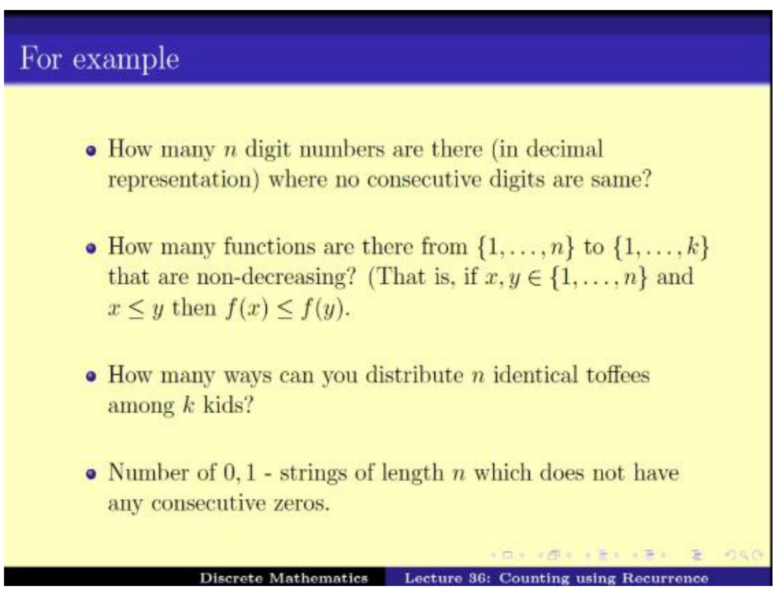
Another problem of this kind is how many ways can one distribute n balls into k bins. And there are certain cases which to be handled whether the bins are distinguishable whether the balls are distinguishable. If the balls are distinguishable does the ordering in the bins matter. Can some of the bins be empty and are there some other restrictions.

(Refer Slide Time: 04:02)



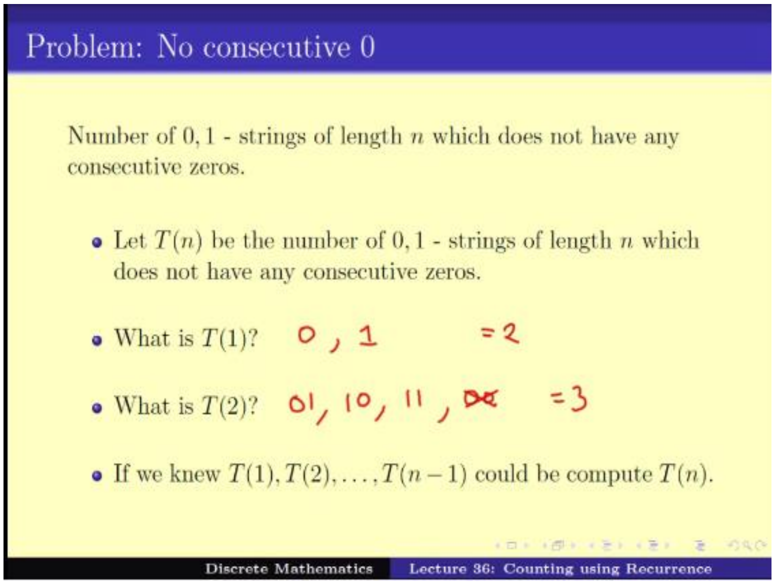
And here also we kind of solve them or kind give you the idea of most of the various cases except for this particular case which is a P(n, k) which is a pretty complicated case by itself and we saw it last video back that this is one of the—these are very challenging problem something that Srinivasan Ramanujam had worked on.

(Refer Slide Time: 04:33)



Now using all these tools and tricks we did look at these problems and we got to see how to solve these top three problems. Now the question that I have is how to solve the last problem. Namely number of 0,1 strings of length n which does not have any consecutive zero. Right.

(Refer Slide Time: 04:57)



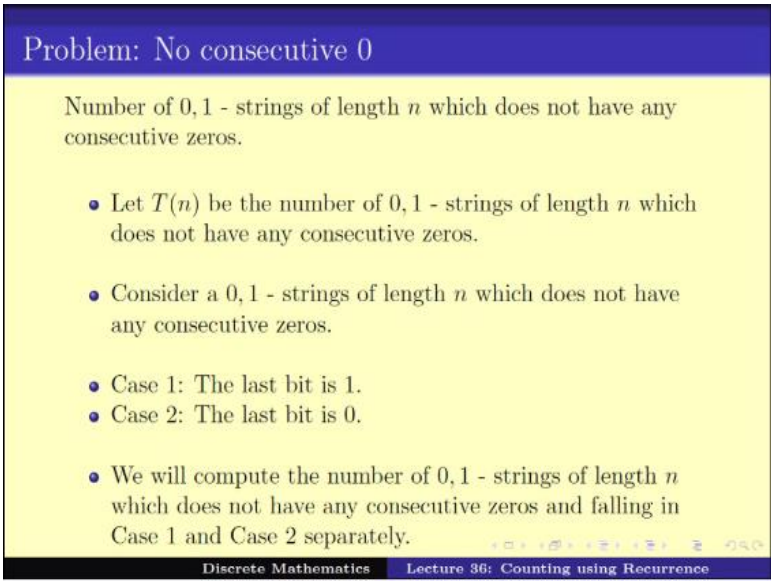
Now you can try out by yourself and realize that none of the standard tricks actually help here. It is that quite complicated to count them count the number of 0,1 string of length n which does not contain any consecutive zero. So let us try to see how can we break this one. So let us T(n) be the number of 0, 1 of length n which does not have any consecutive zeros. So it is parameterized by the number n.

Question is that can we answer this T(n) for small values of m? So what is T(1)? So T(1) of course says that it is the number of 0,1 string of length one that does not contain any consecutive zero. Of course there will be two them namely 0 is one of them and 1 is the other, so the answer is 2. What about T(2)? Again I can have 01, 10, 11 but 00 I cannot have right because 00 have consecutive zero.

So the length of number of strings if length 2 which does not contain consecutive zeros is 3. So like this we can possibly even count what is T(3), what is T(4) and so on. But that is a very bad way of doing it. The question is that if somehow I know how to count T(1) to T(-1) is there a way of computing T(n)? So it is kind of like the induction hypothesis or induction step. To count

T(n) first count T(1) to T(n-1) and then using them can we somehow get an estimate of T(n). So let us see.

(Refer Slide Time: 07:41)

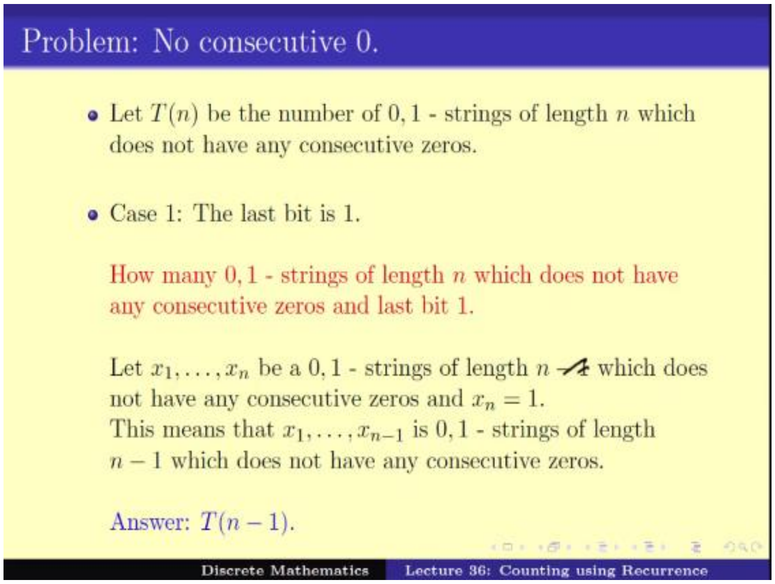


So this is the problem and T(n) is the number of 0,1 string of length n. Now consider a 0,1 string of length n which does not contain any non-zero any consecutive non-zeros. How does it look like? It is something of the form x1, x2 till xn right. It does not contain any consecutive non-zero. So I will break it up in two cases. Of course the cases will be depending which are the last number it is same is 1 or 0. So let us see.

So Case 1 is the last bit is 1and Case 2 is last bit is 0. And then what we will do is that we will try to compute the number of elements satisfying Case 1 and number of elements satisfying Case 2. In other words, we will count the number of 0,1 string of length n which does not contain any consecutives zero and last which is 1. That is basically the size of Case 1. And similarly for Case 2, we will compute the number of 0,1 string of length n which does not have any consecutive zeros and last bit is 1.

Note that these two cases are disjoint. Meaning a string can be either in Case 1 or in Case 2 and have to be one of them. Right. So if I can compute Case 1 and Case 2 then I get the whole answer as sum of the number in Case 1 and sum of the number in Case 2. And that is the way we will proceed forward.

(Refer Slide Time: 09:53)



So let us look at the first case. Case 1the last bit is 1. Now the question is that how many 0,1 string of length n at their which does have any consecutive zero last bit is 1. Now let x1 to xn be a 0,1 string of length n which does not have any consecutive zero and xn = 1. So how does it looks like. So this one into one. Now what happens the rest of them. So let us look at let me draw here so I have x1, x2 till xn-1 and xn. Now x1 is 1.

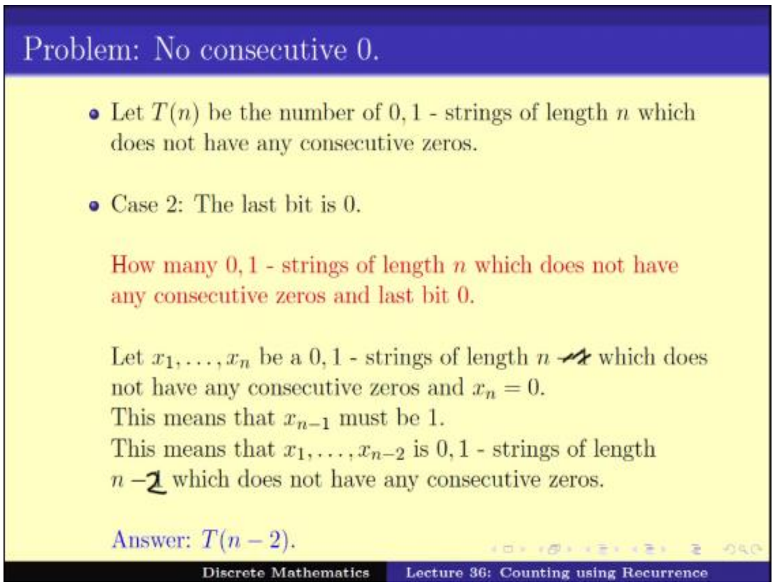
So what are the ways possible? I claim that you put any number x1 to xn-1 such that satisfying the condition that it does not contain any non-zero string I can add a one at the end and I get one of these cases. So clearly one thing is that x1 to xn-1 which is string of length n-1 that does not contain any consecutive zeros because if there are no consecutive zeros between x1 to xn then it is clear there no consecutive between x1 to xn-1, right.

So it is trick. So x1 minus xn-1 is a string of length n-1 that does not consecutive zeros. And in fact it is also the other way meaning gives me any string of length n-1 that does not contain any consecutive zero I can add a 1 and get a string of length n that does not contain any consecutive zero and whose last bit of one. Why it is so? Now see by putting this 1 the only time I can solve a problem is by creating a consecutive zero but since this last bit is 1.

So whatever happened to the bit before that I do not care, these two bit together cannot be a consecutive zero ever because this is one. So this is a-this cannot happen. So in other words, what do we have, in other words, if x1 to xn-1 is 0,1 string of length n-1 which does not have consecutive zero and any such string can be converted into a 0,1 string of length n that does not contain a consecutive zeros and last bit is 1. So the answer for this one is T(n-1).

Note that this is followed, this is exactly the definition of T(n-1), right. So the size of Case 1 or the size of number of satisfying Case 1 is actually T(n-1) correct.

(Refer Slide Time: 13:59)

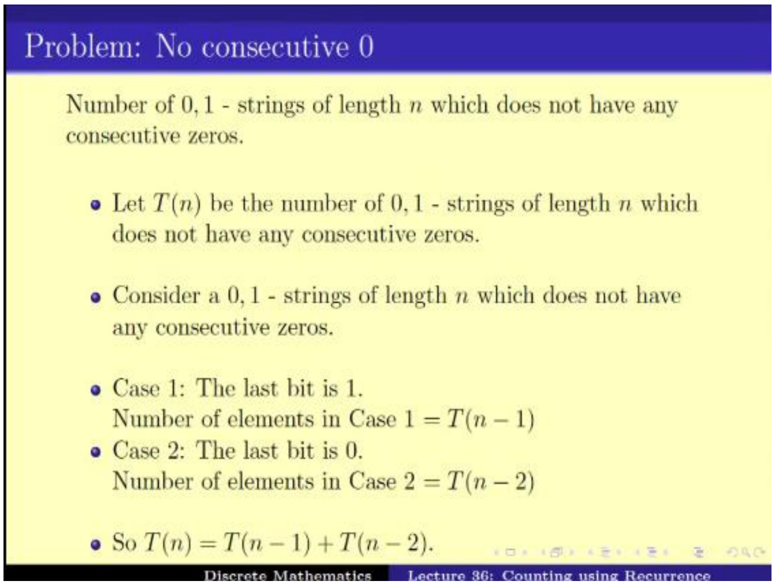


Now let us move to the second case, Case 2. In this case, the last bit is 0. Again, we want to compute now the number of 0,1string of length n that does not have any consecutive zero and last bit is 0. So let us x1 to xn be a 0,1 string of length n which does not have a consecutive zero and the last bit is zero that is xn is 0. Now what happen here let us see? So I have xn-1and xn. This one is 0.

Can xn-1 be 0? No it cannot be. Because if it was 0, then (()) (14:58) about two consecutive zeros xn-1 and xn-2 consecutive numbers that are 0. So from this definition of x1 to xn, I know that xn-1 is 1. Now if xn-1 to one or let us go to xn-2 what happens here xn to xn-2. Now just like we did for the last case, we realize that x1 to xn-2 is now are string of length n-2 that does not have any consecutive ones, these are only the zeros.

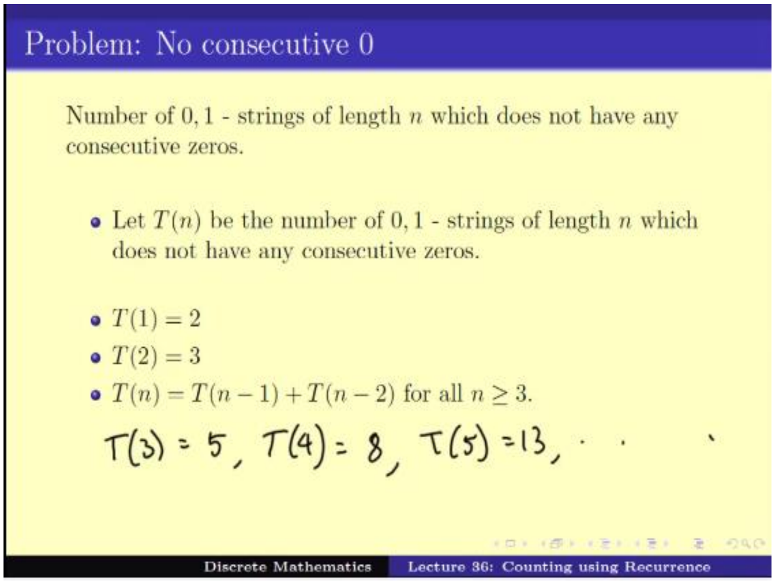
And more importantly any string of length n-2 that does not contain any consecutive zero. I can add a 1and the 0 at the back where the string that satisfies Case 2. So in other words, we have that xn-1 must be 1 which means that x1 to xn-2 is a string of length which is n-2 which does not contain any consecutive zero and hence the answer is T(n-2).

(Refer Slide Time: 16:35)



Now, with this can we do something. Now can we combine them to get some results. Let us see what we have. We started with this value of T(n) and we wanted split up this whole set of strings of length n into two cases Case 1 for which we realize that now that the number of elements in Case 1 is T(n-1) where Case 2 a last bit is 0 in this case the number of elements was T(n-2) and therefore T(n) is equals to T(n-1) + T(n-2). Right.

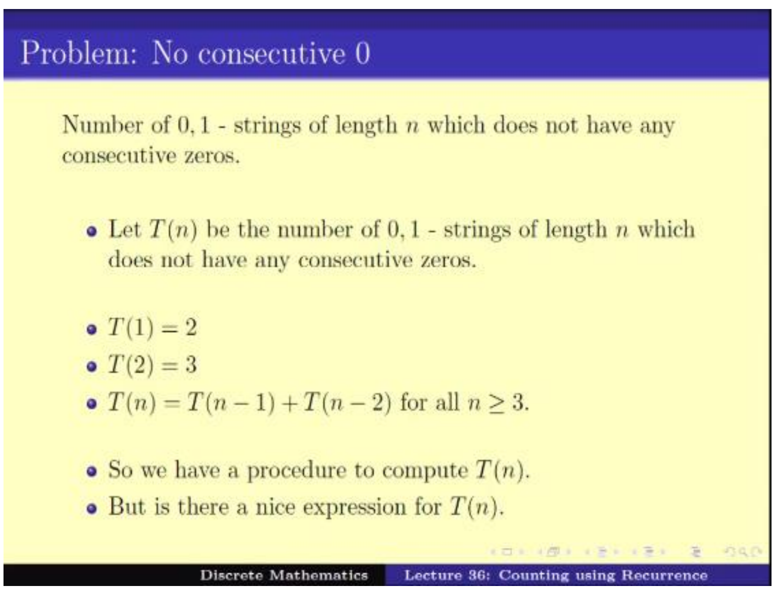
(Refer Slide Time: 17:23)



Now this does not exactly solve the whole problem, so what does it do? So I told this is T(n) I have got T(1) = one I have got T(2) = 2 and I also have got this recurrences that T(n) = T(n-1) +T(n-2). Note that by doing so I can now just keep on I mean I can find out T(n) by picking this problem multiple times for example what is T(3), T(3) is T(2) + T(1) which is 5. What is T(4)? T(4) is T(3) + T (2) which is 8. What is T(5)? T(5) is T(4) + T(3) which is 13 and so on. Right.

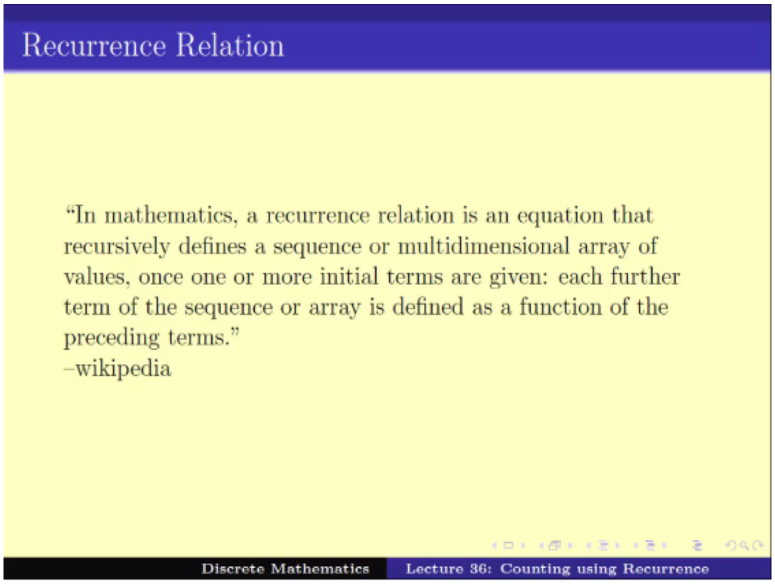
So this way I have been able to count what the value of T(n) is-- I found out a nice way of computing -- accounting the number of elements of length n number of strings of length n of 0,1 which does contain any consecutive zeros. So this is a nice way of counting it.

(Refer Slide Time: 18:49)



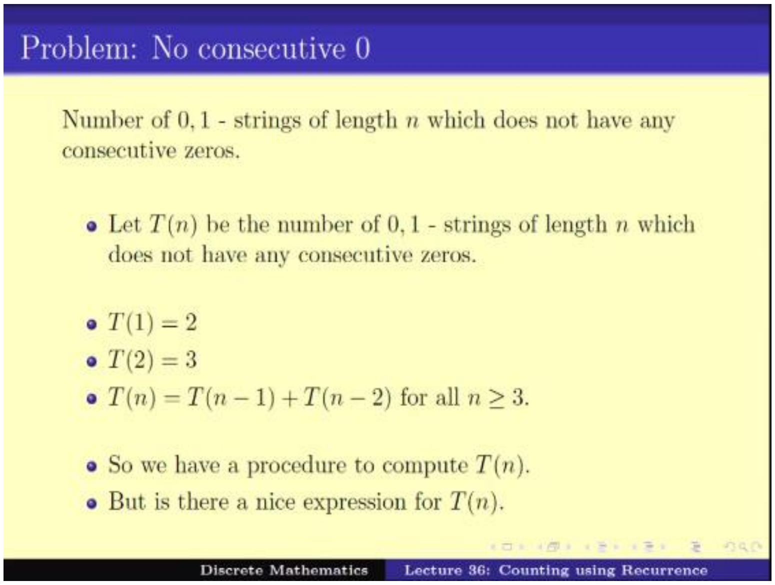
But the question one can ask is can you count with a better procedure of computing T(n) or do I have to keep on apply this process again and again and again n times before I get T(n). One can be get some answer like T(n) = 2 power n by n or something like that. Like we got for the case of the distribution of (()) (19:19) and so on. Cannot we get a closed form expression of T(n). So this brings us to this problem of recurrences relations.

(Refer Slide Time: 19:36)



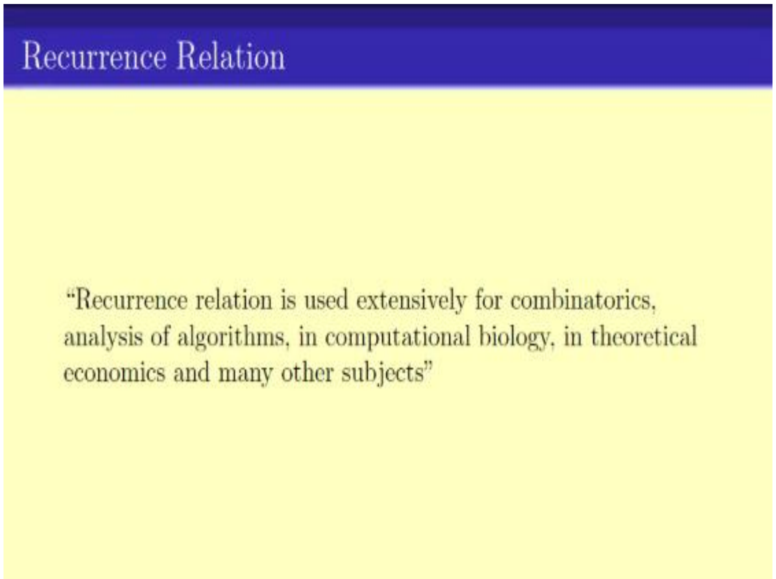
So recurrence relations is a subject, so according to Wikipedia this is a quotation from Wikipedia, "In Mathematics, a recurrence relation is an equation that recursively defines a sequence or multidimensional array of value, once one or more initial terms are given: each further term of the sequence or array is defined as a function of the preceding terms."

(Refer Slide Time: 20:13)



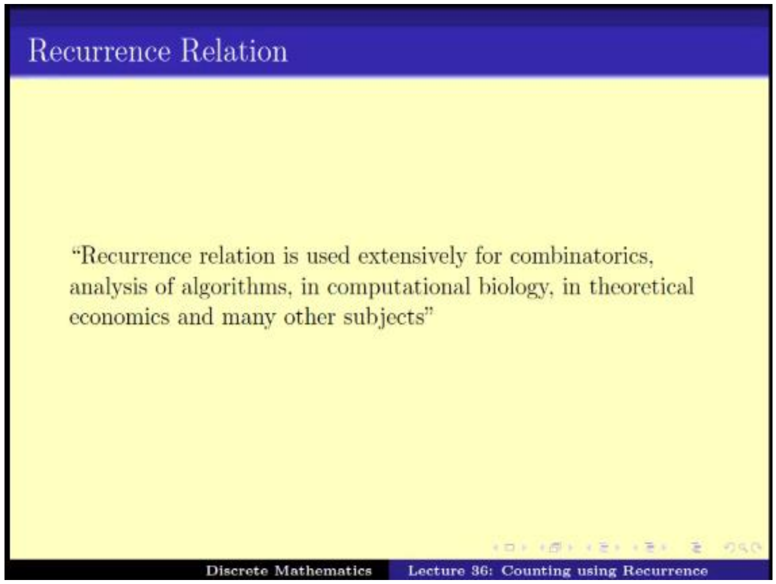
Let us go back and let see that this is actually a recurrence relations. I have been given the initial terms which are these and this is how I solved T(n) by looking at the preceding relations in a preceding numbers, right.

(Refer Slide Time: 20:37)



In fact, recurrences relation is used extensively for combinatorics, analysis of algorithms, computational biology, in theoretical economics and many, many other subjects.

(Refer Slide Time: 20:48)



Recurrence Relations are used for modeling problems particularly counting problems like we sort today. And it is also a big use of recurrence relations is that one can solve recurrence relations easily. Namely one can get some compact form for this T(n) and so on. In the next video lecture, we will see one more example of application of recurrence relations for counting and after that we will be going into how to solve recurrence relations. Thank you.