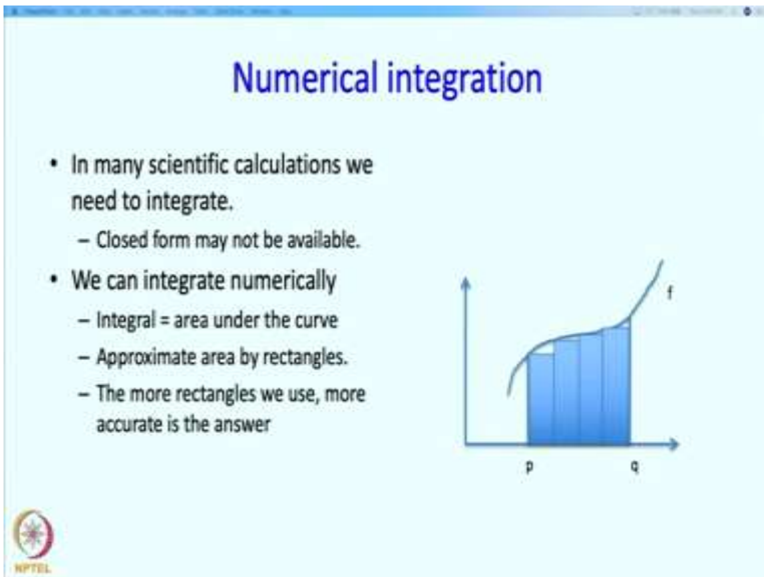
An Introduction to Programming through C++
Professor Abhiram G. Ranade
Department of Computer Science and Engineering
Indian Institute of Technology Bombay
Lecture No. 8 Part – 2
Computing Mathematical Functions
Numerical Integration

(Refer Slide Time: 0:32)

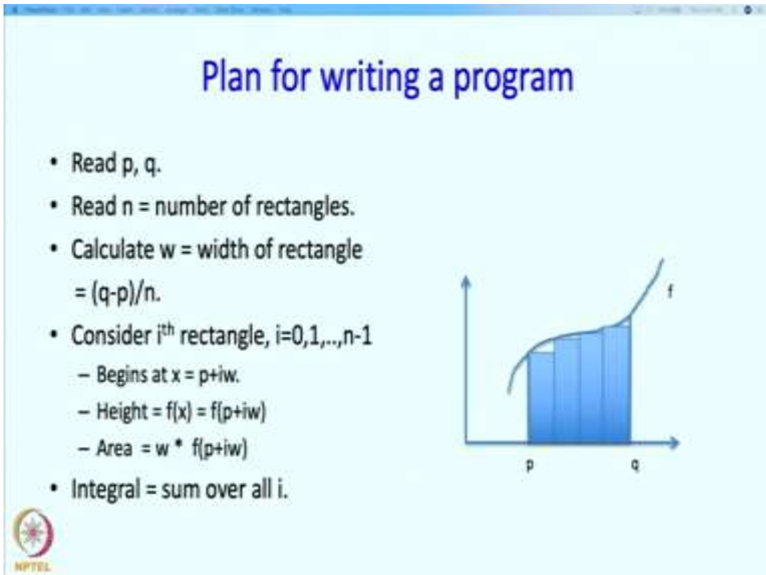


Welcome back, in the previous segment we discussed Taylor series and its use in evaluating mathematical functions. Now, we are going to look at numerical integration. So in many scientific calculations we need to integrate and if the closed form is not available, which is often the case we can integrate numerically. So here we just go with the definition which is that the integral is nothing but the area under the curve. So here is the curve for f and we want the integral from p to q. So it is just the area between in this inside this region from p to q. So underneath so this area, so this is the area that we want to find which is the integral from p to q of that function f.

Now, how do we estimate this area? Well we are going to approximate it by rectangles, so as I have shown over here. So we sort of try to put down the rectangles and the way we are going to put down the rectangle is that we are going to draw these vertical lines. So at this point we draw a vertical line which hits the function and then we put down the rectangle. Then at this point again we draw a line which hits the function and we put down another rectangle. Again at this point we put line and hit the rectangle and so on.

So we put down the rectangles in this manner and our error will be something of this order. So the difference between the rectangles and the area under the line, so say this will be the error. But it should be clear that if we put down the rectangles we are going to get a reasonable estimate and the more rectangles we put down the more accurate our estimate is going to be.

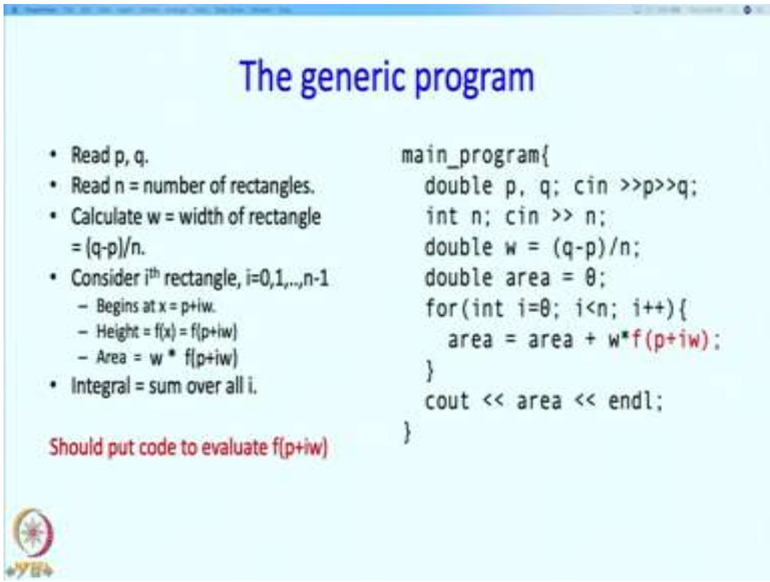
(Refer Slide Time: 2:22)



Now we can make a plan to write this program. So first we should read p and q. So these are, these are the limits of the integration and in this picture so this x-coordinate and this x-coordinate. Then we should read how many rectangles we want, so let us say we want n rectangles. As we said the bigger the value of n is the nicer the fit of the rectangles into that area is going to be.

So then, we calculate the width and for simplicity we are going to assume that all rectangles have the same width. So our width is going to be q - p, so this distance from 0 minus this distance so this width q - p upon n. So, now we want to know the areas. So let us consider the ith of these rectangles. So it is convenient to call this the 0th rectangle, this is the first rectangle, this is the second rectangle and so on. So what is the height of this 0th rectangle? Well, what is this coordinate? So this is p. So the height of this is simply the point at which this vertical line meets the 'f' function. So that is simply f(p). What is the height at this point? It is p + w which is the x-coordinate over here. But not p + w, f applied to that. This is going to be f(p+w) and so on. So in general you can see that the ith rectangle is going to begin at p+iw. And its height is going to be f(x) equal to f(p + iw). So this point is going to be p + iw and this height is going to be f(p+iw). So if f is the function that we want to integrate, we had better be able to calculate this. And of course the area is what? The area is the width of each rectangle. So this is w which we calculated over here times f(p+iw) which is the height of that rectangle. So the integral is just the sum over all i.

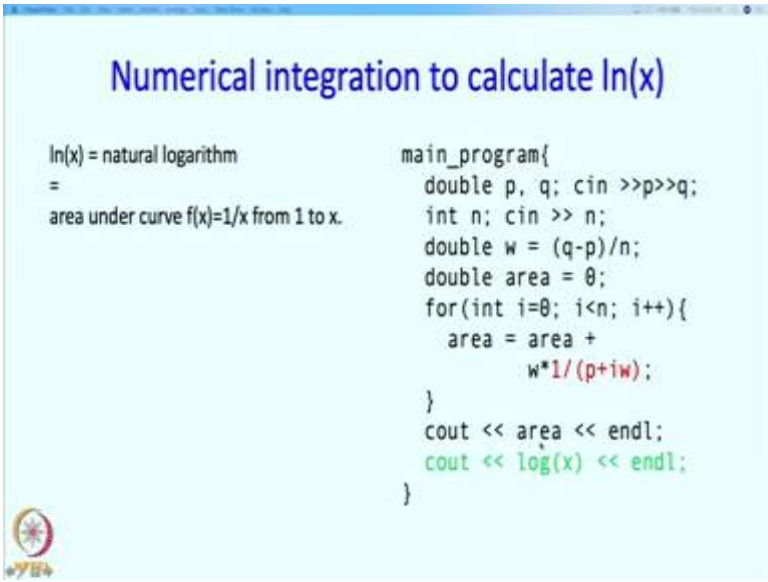
(Refer Slide Time: 4:38)



So this is our generic plan, I have just rewritten it over here and now we can write the program. So our program goes as follows, we are going to declare variables p and q and we are going to read them. So these are the limits of the integration. Then we are going to read in the number of rectangles that we desire, so that is n. Then we will calculate the width of each rectangle that is going to be (q-p)/n.

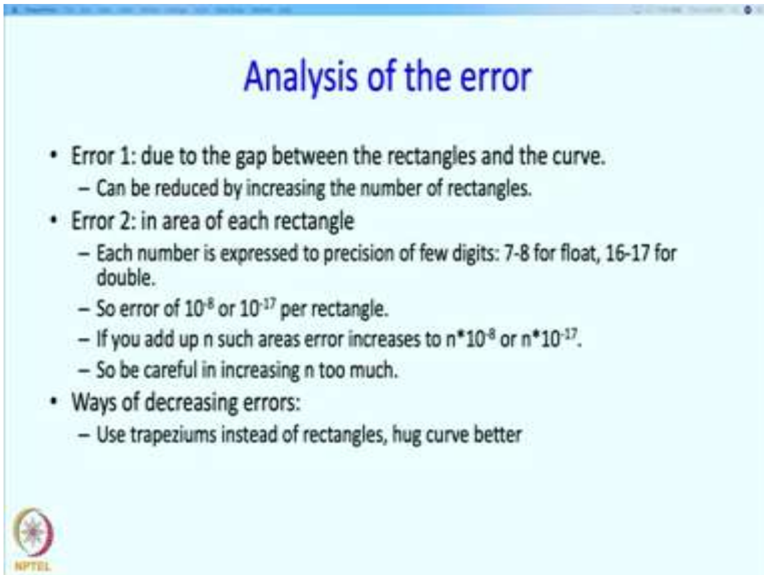
So I am really just following along what I have written over here. So after that we are going to do have a loop in which we are going to look at all those rectangles. So of course, we want to have a variable to accumulate that integral and so that is the variable will call area. So now we have the loop and in the loop we are just going to account for the areas of all these rectangles which are numbered 0 through n-1. So that is what this loop is going to do. So the numbering of the rectangle is exactly going to be the control variable inside this loop. So, what do we do inside? Well we are going to take this old area and to that we are going to add the area of the new rectangle. And, what is the area of the new rectangle? It is w times the height of that rectangle which is f(p+iw). So right now I am writing a program which is generic in the sense that it will work for whatever function it is. But you will have to put in code over here which calculates this function. So at the end of it we are going to print out the area and that is about it. So we just need to put in the code over here to calculate this area and we will be done.

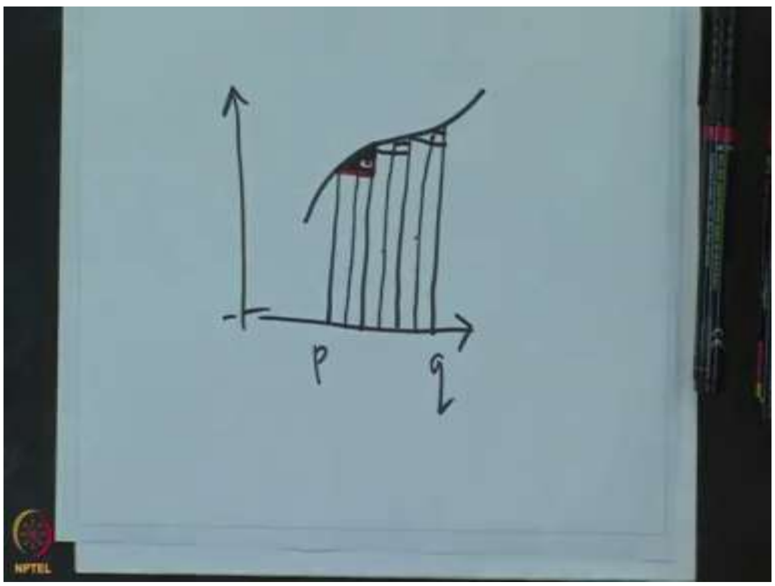
(Refer Slide Time: 6:44)



Now we are going to apply this to another specific problem which is the problem of numerical integration. So here we are going to find ln(x) or the natural logarithm of x. And as you might know this is exactly equal to the area under the curve f(x) equal to 1/x from 1 to x. So in other words, what we have over here is that f of x is 1 upon x. So here is our old program and in over here we had f(p+iw). But instead of that we know now that what a specific is, so that is just going to be f(p+iw) is just going to be 1/p+iw. So our program for calculating ln x is going to be exactly this. Now since we are calculating ln x, we can actually print out over here the official value or the C++ library log function value, which gives us ln of x. And then if we compare these two things we will know how good our calculation is.

(Refer Slide Time: 7:42)





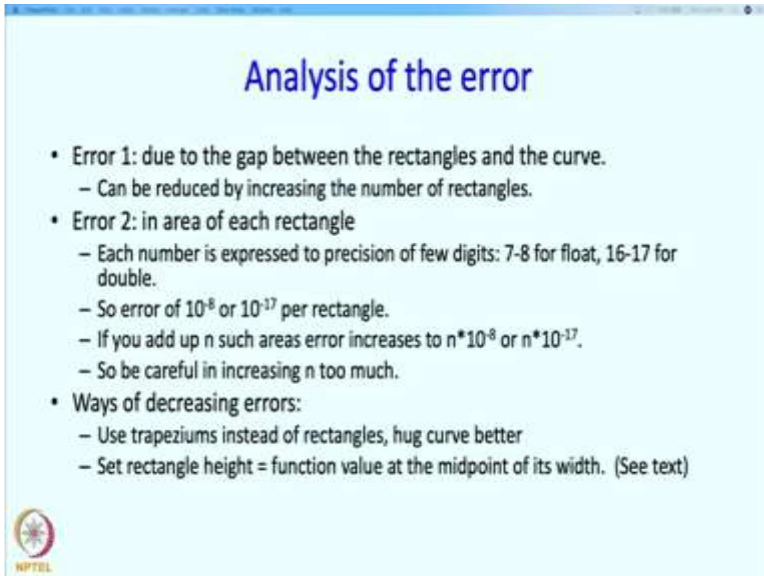
Now, we can do that and it turns out that that gives fairly good results. And now here is some analysis of the error that we can do. So the first error is due to the gap between the width of the rectangles and the curve. So we said that we can reduse the gap by increasing the number of rectangles. So maybe I should draw a picture just to see how that might happen, why that might happen?

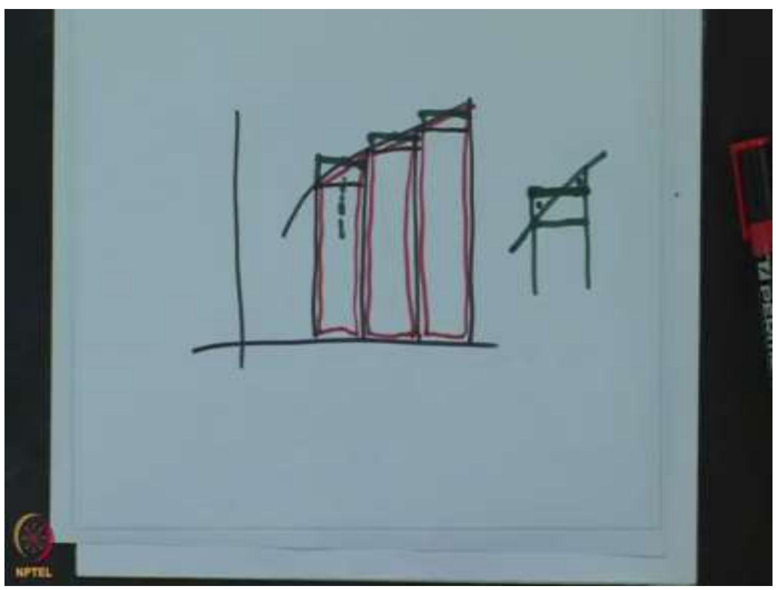
So let us say this is our x axis, this is our y axis and say this is our function which we want to integrate between this p and this q. And what we did was, we put in these rectangles and that area of these rectangles was approximating the actual area. Look what will happen, if we had twice as many rectangles but of half the size. So if we did that these rectangles would get closer to the actual curve. So originally the error would have been this this red triangle. Now instead of that, the error is limited to this triangle and this triangle. So that extra line that we put, knocked off this part from the error. So that is one way to reduce the error. But that is not the only error.

In the area of each rectangle also we have some error. Why is that? Because each number is expressed to precision of a few digits. So if our numbers are floats then only 7-8 digits we get. If we are, if our numbers are doubles, then we only gets 16-17 digits. And furthermore when we multiply two numbers, we lose additional precision. So the end result is that every time we add up two numbers, the error that we have in each of those numbers is going to accumulate.

So in each area we have an error of about 10 to the power minus 8 fraction. Because only the first 8 digits we have. So our error is going to be 10 to the power minus 8 or 10 to the power minus 17 per rectangle. So if you add n such areas, the error can increase to n times 10 to the power minus 8 or n times 10 to the power minus 16. So it means that you should not add too many too many rectangles, certainly not for float. So, if you are having double then adding many rectangles is ok. So, how do we decrease the errors? So the way to decrease the errors is first of all use trapeziums instead of rectangles.

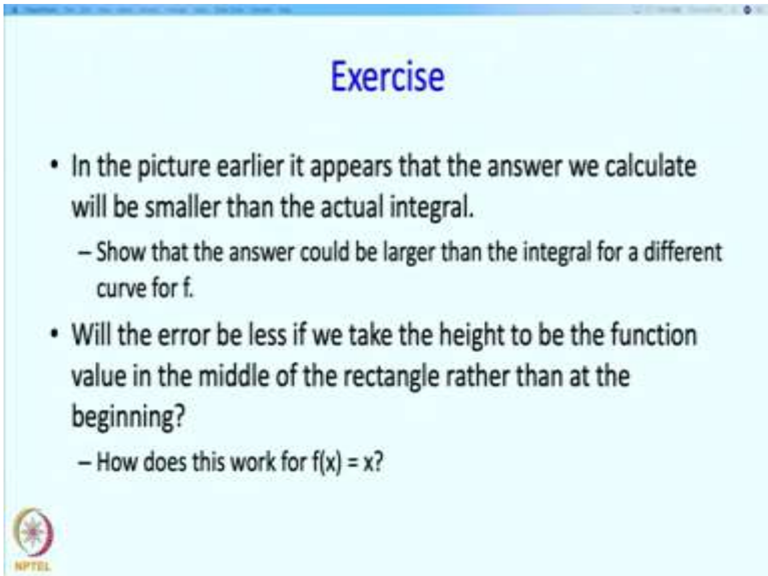
(Refer Slide Time: 11:16)





So let me explain that. So let us say these are our 3 rectangles that we used. If we were to use trapeziums we would use this region. This would be our trapezium. You can see already that the trapezium is going to hug the curve much better and so the error is going to be much less. Another idea is to set the height of the rectangles slightly differently. So we are going to keep the same rectangles but here we use this as the height. Instead of that we are going to take the midpoint of the, the midline of the rectangle and make it hit the function. So this is the rectangle that we are going to use. So even here our line will pass through this middle region, even here this line will pass through this middle region. So if I have to explore this a little bit, so this is our function, these, this is our original rectangle and this is our new rectangle the top of the new rectangle. So what has happened is, that we are overestimating the area in this region but we are underestimating it in this region. So as a result the agreement with the actual area is better for this rectangle.

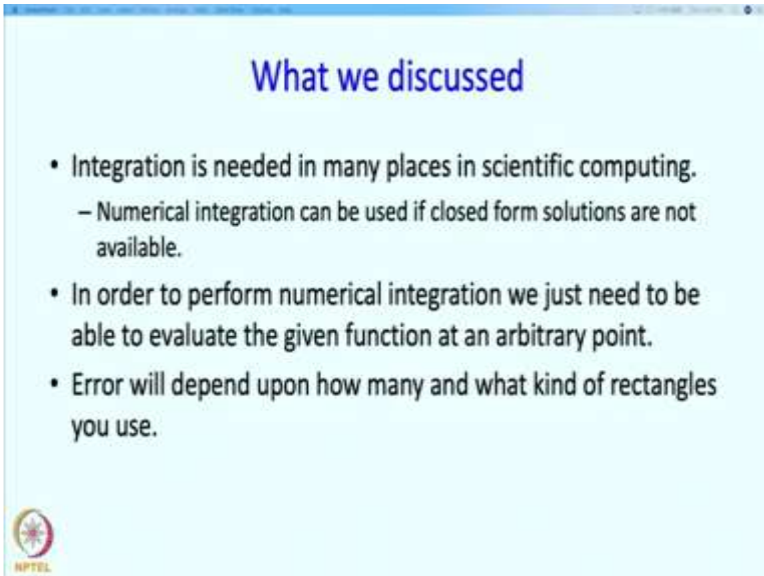
(Refer Slide Time: 12:57)



So some exercises, so in the picture that we do drew it appears that the answer we calculate will be smaller than the actual integral. So I want you to think about that and just observe that just happens because of the way we draw the function. So if I (drew), the way we drew the picture of the function. So if we drew the picture in a different way, so a hint is have a decreasing function then you would see that our estimate is actually going to be larger than the actual integral. So I would like you to do this exercise and persuade yourself.

Then to check whether the error will be less if we take the height to be the function value in the middle, I want you to do the following thought experiment. So suppose I am using this to find the integral of f(x) is equal to x. Of course, this can be easily integrated but we are using this as a thought experiment just to check what happens if we take the midpoint of the integral. You will see that in this case the midpoint that the taking the height at the middle of the rectangle works beautifully.

(Refer Slide Time: 14:15)



Alright, so, what did we discuss in the segment? So we said that integration is needed in many places in scientific computing. And numerical integration can be useful if closed form solutions are not available. In order to perform numerical integration we just need to be able to evaluate the given function at an arbitrary point. And we also said that the error will depend upon how many and what kind of rectangles you use. And of course we use numerical integration to find the value of ln x the natural log of x. So this finishes the segment and will take a break.