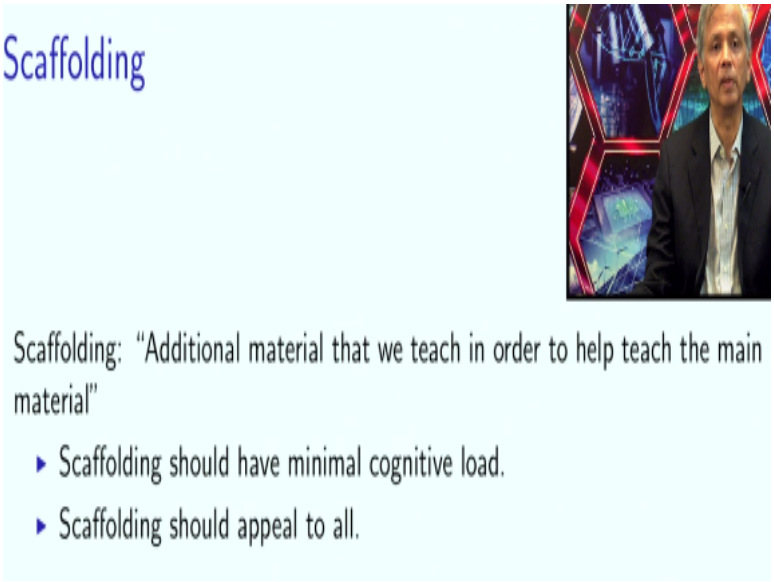
Design and Pedagody of The Introductory Programming Course

Prof. Abhiram G. Ranade
Department of Computer Science and Engineering
Indian Institute of Technology – Bombay

Lecture – 13
Pedagogy.1: Scaffolding, Lesson Plan

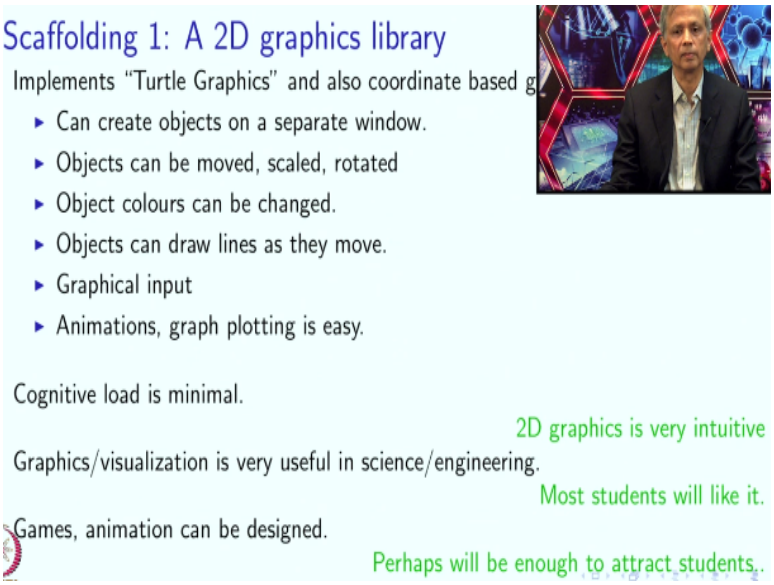
Welcome back, we have been discussing how to deliver our material or the pedagogy of the course and the topic in this sub lecture is scaffolding.

(Refer Slide Time: 00:31)



Scaffolding as we have mentioned earlier is additional material that we teach or somehow provide to the student in order to help teach the main material. So, this scaffolding is not the main material. It is the material which helps us. Scaffolding should have minimal cognitive load. It is should not really burden the student in anyway. The student should not think of it as something heavy, something that makes the student nervous, not at all. Everybody has to learn the scaffolding, so it should be very appealing to everyone.

(Refer Slide Time: 01:08)



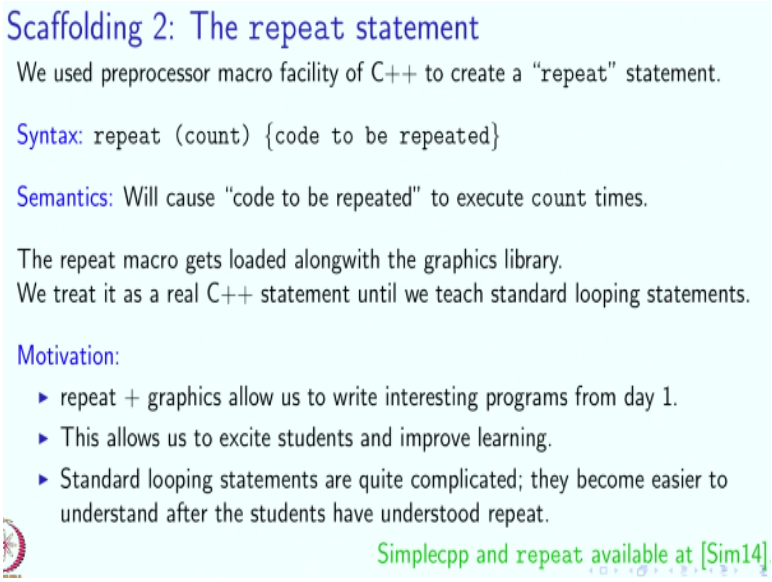
The first scaffolding that we use is a two-dimensional graphics library. One part of this graphics library implements the so called turtle based graphics or turtle graphics and another part uses coordinate based graphics. What does this allow us to do? This library will allow us to create a window and it will allow us to create objects on that window. Objects can be moved, scaled, rotated. Object colors can be changed, all the usual things, but all this is two-dimensional.

The objects are sort of like rectangles and squares and circles which move on paper. They are not like cubes or anything like that. Objects will draw lines as they move. We are going to see examples of this in a few minutes. We will also have graphical input. So, the user will be able to click on our graphics window and our program will be able to respond to this input as well. What does this do for us, it will allow us to create animations, also it will allow us to do things like graph plotting, which is very, very customary, very, very commonly done in science and technology like subjects.

We will see in a minute that the cognitive load of this graphics library is very, very little. This is because 2D graphics is actually a very intuitive subject, we know it quite well. We will talk about square, circles and everybody understands these things quite well. 3D graphics on the other hand is tricky because when things rotate and all of that it is a little bit more complicated. But, 2D is really simple. Graphics and visualization is very useful in science and engineering.

Students already have some kind of skill for the topics that we are going to talk about. As I said, games and animations can be designed and students will love it for that reasons.

(Refer Slide Time: 03:21)



The second part of our scaffolding is the so called repeat statement. This is not really a statement in C++. We are going to use the preprocessor macro facility of C++ to create a repeat statement. Many of you may know that there is a preprocessor macro facility. Anyway, the textbook that has been mentioned in the course does have a discussion of the preprocessor macro facility of C++, which actually is there in C as well. It is a very simple thing.

When we load the graphics library, this macro will also get loaded. Here is a syntax. If you say repeat followed by a count, a number or a numerical expression, then code following that in braces will get repeated count number of times. Code to be repeated will execute count number of times. This macro gets loaded along with the graphics library, so the user does not need to do anything special.

In fact, the graphics library will get loaded without the user really knowing it because we will provide a compilation script which will do the compilation for you or there will be an IDE, which you can use a simple CPP library, the library that we are talking about. That IDE will be setup so that if you hit the compile button, then things will get compiled and all the libraries will get linked without you having to do anything specific.

Now, as far as this repeat statement is concerned, we are going to treat it as a real C++ statement. On the first day we are not going to tell the student that it is not sort of basic C++. But, when we teach the standard looking statement then we will say that look the repeat statement was like training wheels that we put on, on you bicycle so that we could get going very quickly and you will see why we are saying that. But, until then we will just say that it is a standard C++ statement.

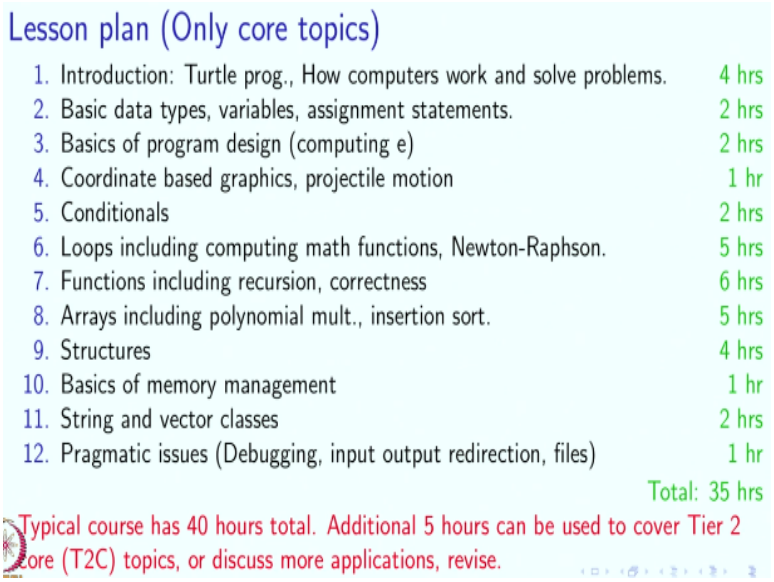
By the time we talk about standard looking statements, the students will not need to have the repeat. Why are we doing this? Repeat plus graphics allows us to write interesting programs from day 1. That is the idea. That is why we are doing it. Otherwise, we will see that there is great amount of difficulty. We mentioned the slow start problem and I will explain that in the next sub-lecture as well. But, that slow start problem can be nicely overcome and we can give student something very interesting and yet something very substantial, something very much connected to our subject.

With repeat and graphics, we are enabling students to improve their learning as well as get excited about the material. Standard looking statements actually turn out to be quite complicated. There have been studies and have eluded to some of these studies and these studies show that students have difficulty in understanding standard looping statements. Repeat on the other hand we will see is very easy to understand.

Once students understand repeat, they are sort of half way towards understanding more complicated statements like a while or a for. This I believe will improve or will help in making students understand the more complicated standard C++ or C or Java or most languages for example, the statements that most languages have. They will be easier to understand after students use repeat with confidence.

But, simple CPP, the graphics library and the repeat statement are available at the website given by this reference and this reference is at the end of the slides.

(Refer Slide Time: 07:53)



I am now going to tell you about the lesson plan that we are going to use in teaching this course. The lesson plan only talks about the core topics. As I said, there is a Tier 2 core as well. The Tier 2 core we are going to discuss later on. We are just going to talk about a lesson plan for the core topics and then we will talk about some more details about how to teach those core topics. The first topic is the introduction. In this we will use what is called the turtle programming, so we will see that in a minute.

We will also talk about how computers work and how in principle they solve problems. This is going to provide some kind of background material to students to understand the course on the whole. This will take about four lecture hours. Then we will talk about basic data types, variables, assignment statements. This will take about 2 hours. Then we will talk about the basics of program design.

We have talked about this program to compute e, so we will over it and that will take about 2 hours in the classroom. After that, we will talk about coordinate based graphics. In this we will give one example, which will together it should get over in about an hour. After that we will talk about conditionals, which will take 2 hours and there will be loops after that standard loops and they will include how to compute math functions, e is only one example, but we will talk about other math functions as well and we will talk about things like the Newton-Raphson method for finding roots.

We will also talk about style issues, when should you use a for loop, when should you use a while loop and things like that. That will take about 5 hours. Then we will talk about functions including recursion and including arguments for correctness. By the way, arguments for correctness will sort of come in every time. So, starting from basic program design we are going to ask students to be putting in plans. Students should say that look when I write this loop this is going to be my plan.

We will not spend specific lecture on how to argue correctness, but correctness will get discussed. But, recursion when you talk about recursion there is sort of a new way of arguing correctness, so we will talk about that as well. Discussion of functions including recursion is expected to take 6 hours. Arrays and in this we will include uses such as polynomial multiplication, insertion sort and this is expected to take 5 lecture hours.

After that we will talk about structures. This will take about 4 hours and then we will talk about basics of memory management. Very little, so this will take about an hour. I do recommend and we should talk about the string and vector classes in the standard library of C++. This could take about 2 hours and believe me this is time well spend because your students will be able to write programs much faster with these features and with the basic C like features that they get.

Last but not the least, we should spend about an hour telling students how to debug, telling them about things like input output redirection. Files are given an hour by themselves very often. But, in C++ declaring and using files is very, very simple. So, I would say it could be done in 15 minutes and then you could give some code which contains use of files and then the student could just modify and learn.

There is nothing principally interesting or important about files and therefore all these issues I believe can be discussed in about 1 hour. The total this takes us to about 35 hours and since a typical course has 40 lecture hours, this leaves 5 hours. So, those 5 hours can be used to cover the Tier 2 core topics or maybe you feel that some of these topics themselves could be discussed at more length.

You might feel that look I have talked about these topics, I have talked about some applications, but I really want to talk about more applications. I want students to write more programs. I want to discuss more programs in class. That is perfectly fine. So, the 5 hours can be used for many things and of course that is discretion that the teacher has to use.

After this we are going to go through many of the topics which have been mentioned and we will do a quick tour of the course saying how the different topic should be taught if there is anything interesting that I have to say about those topics. So, we will take a short break here.