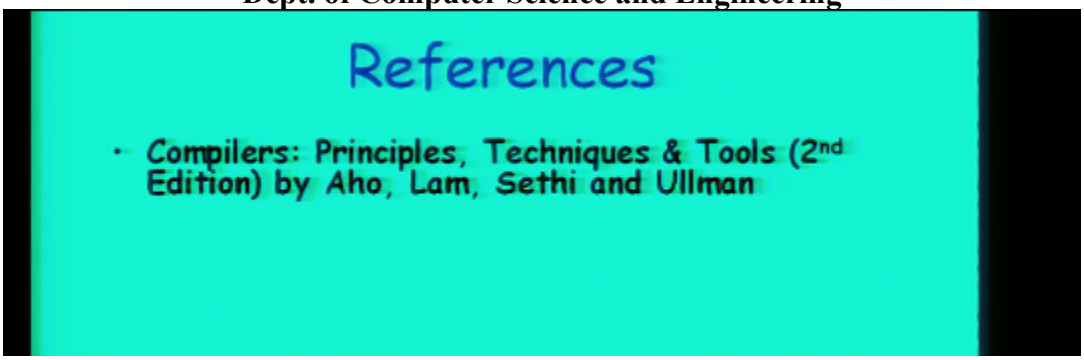
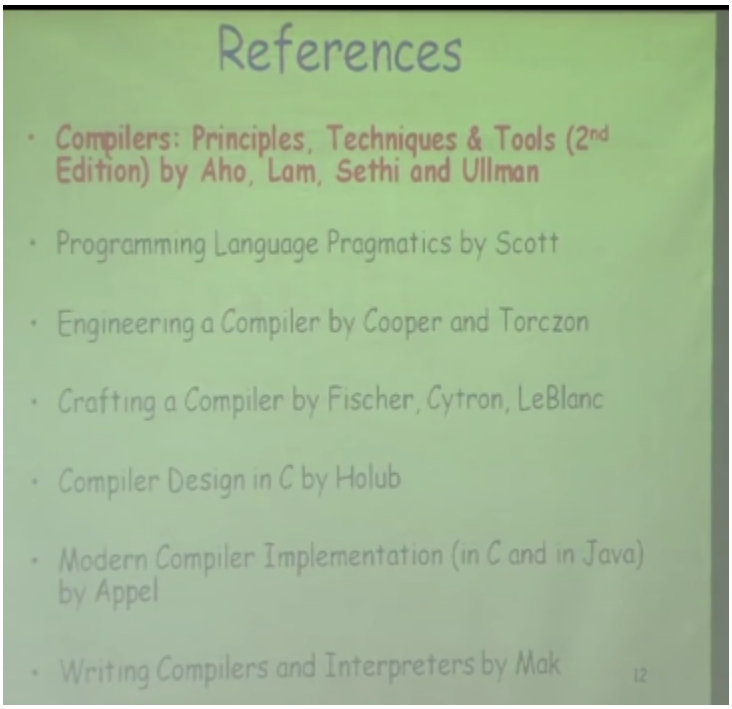
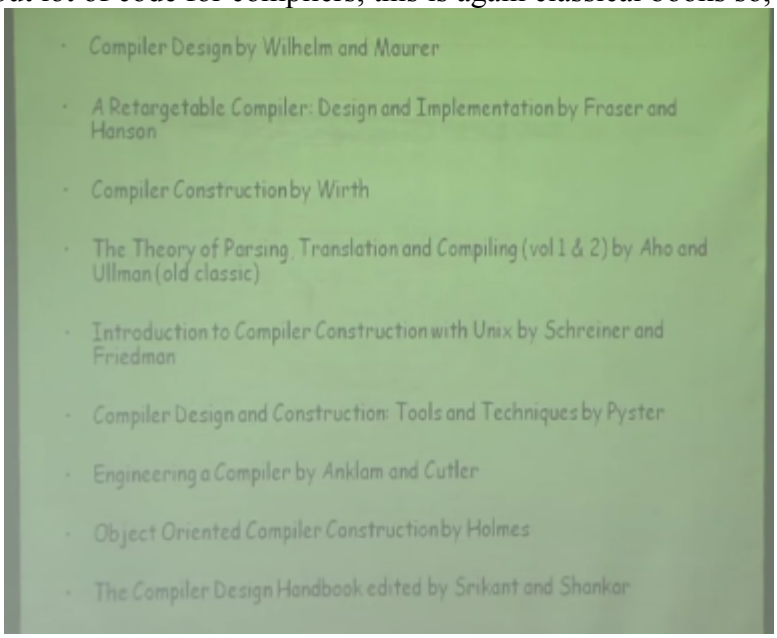
sIndian Institute of Technology Kanpur
NP-TEL
National Programme
On
Technology Enhanced Learning
Course Title
Compiler Design
Lecture – 01
By
Prof. S.K Aggarwal.
Dept. of Computer Science and Engineering



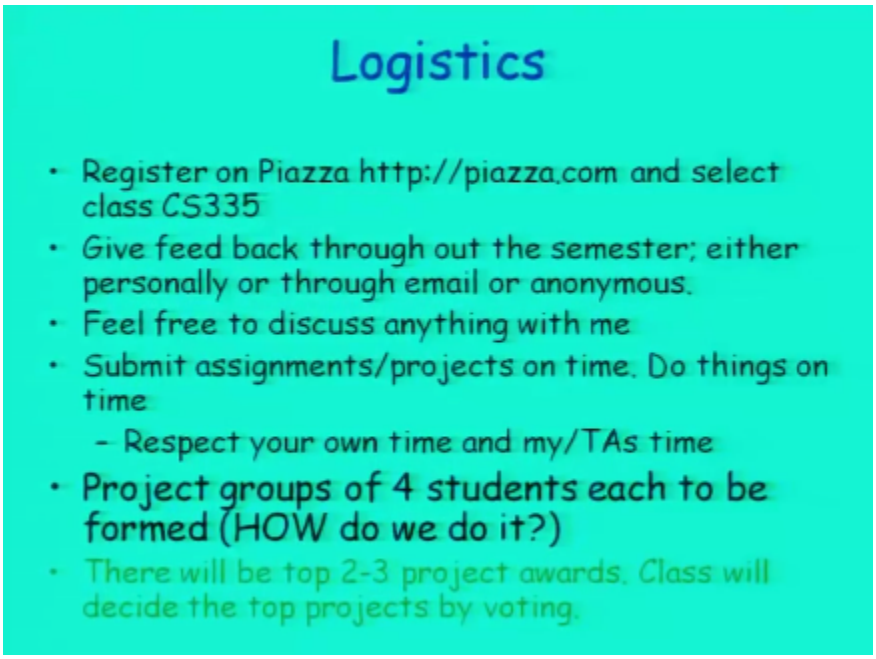
So the some parts we or discussed in previous class, so we before I start on that, okay, talk to each other, yes, just a little students, it is fine but I cannot reveal that thing, I still hold that or third point, I'm not going to choose not to accept your papers, it's a no, if you want you give preference, first one as I said, I'm not going to change a group of two, but I may not, you might give more than one, but I'll decide how to make much, so this is not actually the prospections, first I could not change anything, so this is going to read the main text for us, compiler, principles, techniques and tools, okay, by Aho, Lam, Sethi and Ullman, this is the standard set of rule which used in compiler also, popularly known as the dragon book okay, you can find lot of copies of this in the library, if you want to buy personal you can just go to topic, and while typing here, I think it's available in Chief Indian Edition, but I have also listed lot of other books of which at least first I also sort of use for taking some materials so as I said this remains the base text and some material will also be taken from here which is the book by stocks and book by input, and rest of the books I have listed but these are good references for compilers, but mostly I'll not use that, okay, so as I said first one and to



some extent number 2 and 3, but this is good to know that they are other books, this is another good book than this book really talks about lot of code, so this may not go deep into the tools but it will talk about lot of code for compilers, this is again classical books so, and here is much



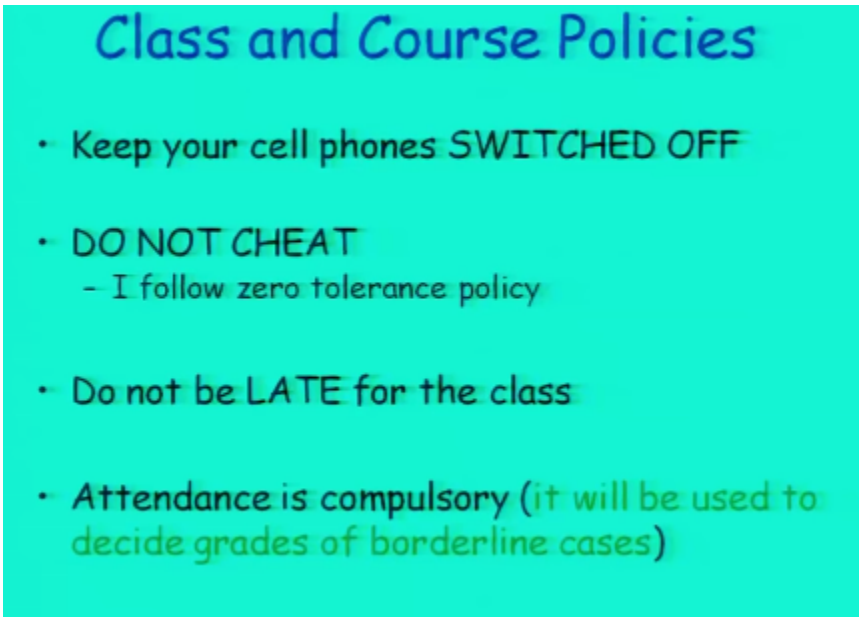
longer list, okay, if you just want to see these books you can always, many of these are not in the library you can always check them in my office, okay, so at some point of time at least I have scanned all these books to find the relevant material, I have found the first one it seems to be where, which can be used really for course like CS335, okay, other books will have some different focus and I may not want to use that, this is very once again I am not sure how many



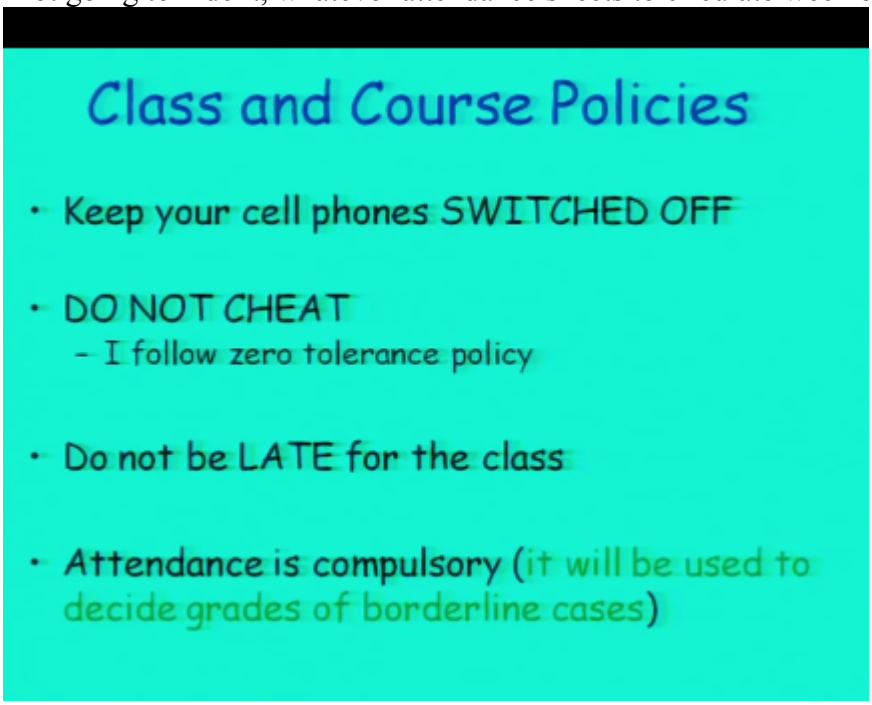
of you attended the previous class, because they were some 20 odd guys who were missing in the previous class, so all of you have to register on this.

Now there is something wrong with the registration I got a message from the other team you are actually going and registering on the wrong books, so please look for I dictate within that select CS335 and look at winter 2013, okay, this session. So you have to look at this keywords and then register for this course, I have asked the administrator, some of you have actually register the wrong course and I have asked that administrator to shift it from that course to the whatever reason course but at least check that you've registered in the right books, okay and also as I said you can leave feedback throughout the semester, you can use any forum you wish, personal discussion and anonymous email, email whatever you want, okay. Do things on time and this part you have already sort of threshold, in case you are going to make groups, and we are going to have two to three top project of ours, which are going to be in kind, not in cash, okay, unlike all the student activities, these are going to be, this are academic awards so it's not certificate you may get a book or some catalog material, okay, but these awards I will decide but those who are the award winners will be decided by you, you choose which are the three project best projects by voting and whosoever gets the maximum number of votes, okay, that project will be dealt as best project for award, may not be for the great, okay.

Your phones must switched off while you are in the class, please make sure, no cheating, and come on time, okay.



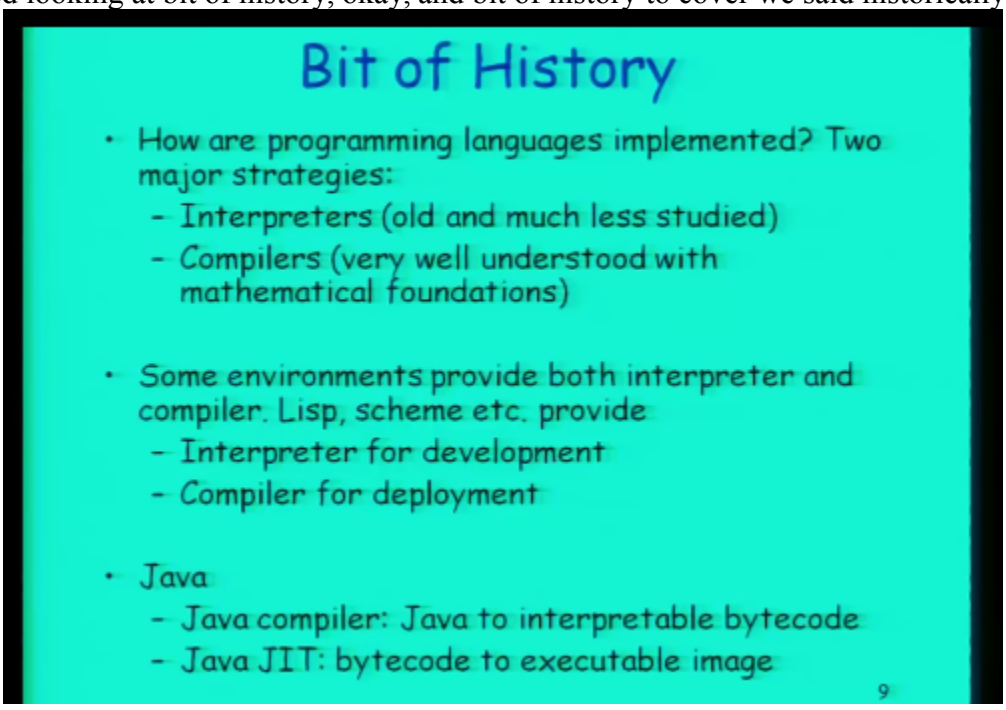
Today also we saw influx almost up to 97, please avoid that and as I said attendance is going to be compulsory in this course, I will regularly post the attendance data on public forum where all of you can see that who else is attending how many classes, okay. This data is going to be public and I'm not going to hide it, whatever attendance sheets to circulate week by the



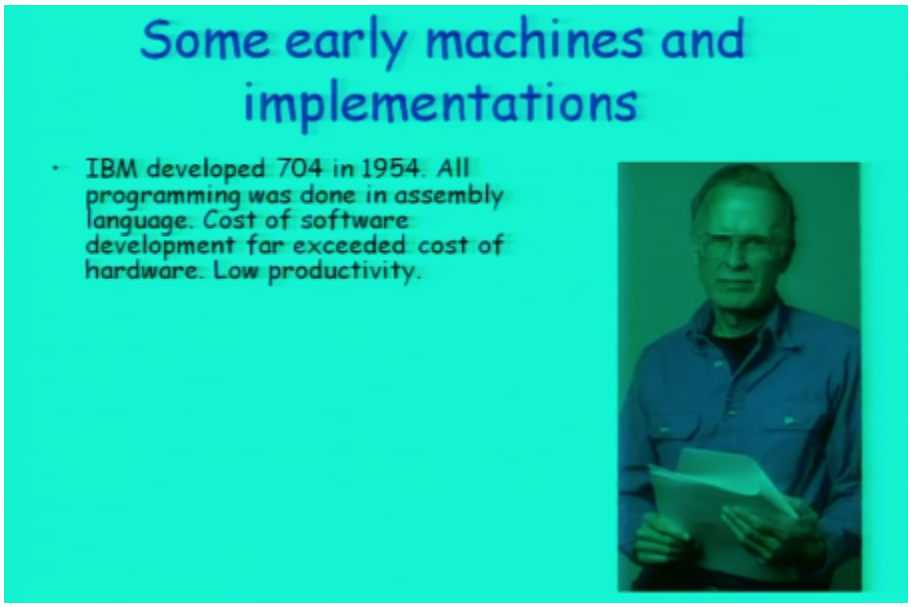
weekends your data will appear on the website, course website, okay, everyone will know who are attending, okay.

- Compiler Design by Wilhelm and Maurer
- A Retargetable Compiler: Design and Implementation by Fraser and Hanson
- Compiler Construction by Wirth
- The Theory of Parsing, Translation and Compiling (vol 1 & 2) by Aho and Ullman (old classic)
- Introduction to Compiler Construction with Unix by Schreiner and Friedman
- Compiler Design and Construction: Tools and Techniques by Pyster
- Engineering a Compiler by Anklam and Cutler
- Object Oriented Compiler Construction by Holmes
- The Compiler Design Handbook edited by Srikant and Shankar

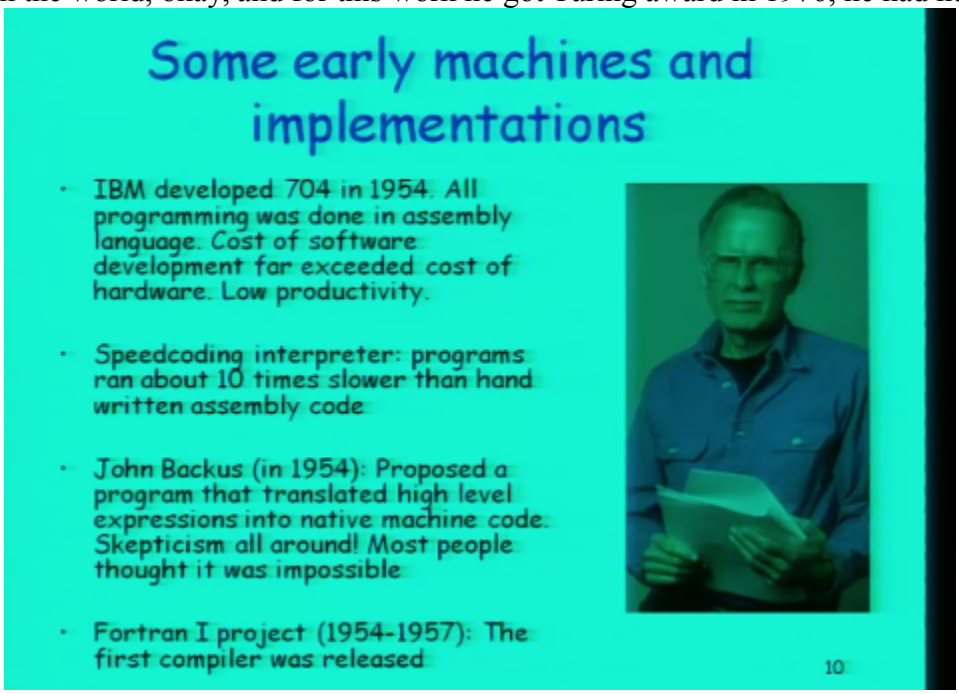
Now going back to something we started discussing towards the end of the previous class, okay, we started looking at bit of history, okay, and bit of history to cover we said historically



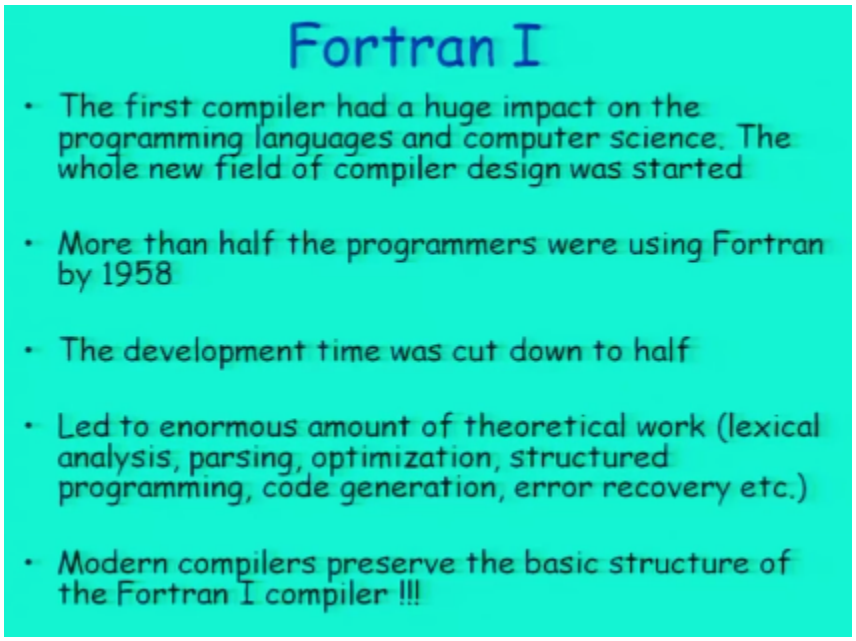
we do not have compiler still medical fees, and people were using machine coding directly, but the two approaches normally for interpreting high level language, one is you want to interpret another is you want to compile and then their languages which provide you and with support. And so this person this gentleman is John Backus, who actually developed the first compiler,



okay and for that he got the Turing award, Turing award is the highest award, a computer scientist given this is almost equivalent to what is known as the Nobel Prize in Sciences, so if you say there is somebody like Turning award, in computer science that is almost considered at that level in the world, okay, and for this work he got Turing award in 1976, he had huge



impacts on how the languages were developed, how the languages were implemented and this project which most people believe could not be done, he really made it possible 50 years back, more than 55 years, so 1957 which was 55 years back the first compiler was written. An interesting part was the impact of this compiler, okay, so that really is a huge impact and what it led to was that all these so interesting part was there was no theory of compilers at that point of



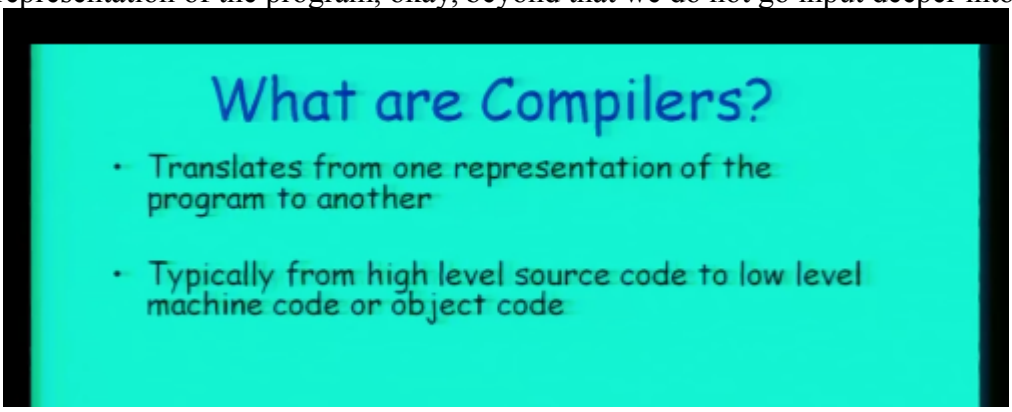
time they just wrote a compiler, and subsequent to that people started developing theory and all the theoretical work which was lexical analysis, parsing, code generation, optimization, it really happen after that, after they had developed the compiler and interesting part is we still have almost all compilers today, at the same structure as the original compiler which was written 55 years back, and it was released 55 years back the same structure is present, okay, so you can see that kind of insight he has into compiler design in to implementing languages and kind of phase they were able to do without having a formal theory into in bringing something into implementation and then subsequently during interval, so this is something you have to keep in mind that kind of things which were in compiler and historically what analyzer compilers, okay, so this is where I would like to close the discussion as far as introduction to the code is concerned, okay and get on with the compilers, right, unless you have any questions or comments, alright.

So let's then move on and let’s try to see what do we mean by compilers. So what is the compiler? There'll be using compilers for at least two, two and a half years, right, so what is the compiler, what do you understanding by that? So this understanding is something which is going to lay the foundation for subsequent things we do in course, technically if we don't know what a compiler is, we will not know how to build a compiler, so first thing we need to understand is what a compiler is so that we can made a compiler, okay, so anyone? What is the compiler? Okay, so let me just write down some of these suggestions and then we can understand it, okay so the compiler is it is program in high level language, and what it gives me is which executable which executes on the machine.

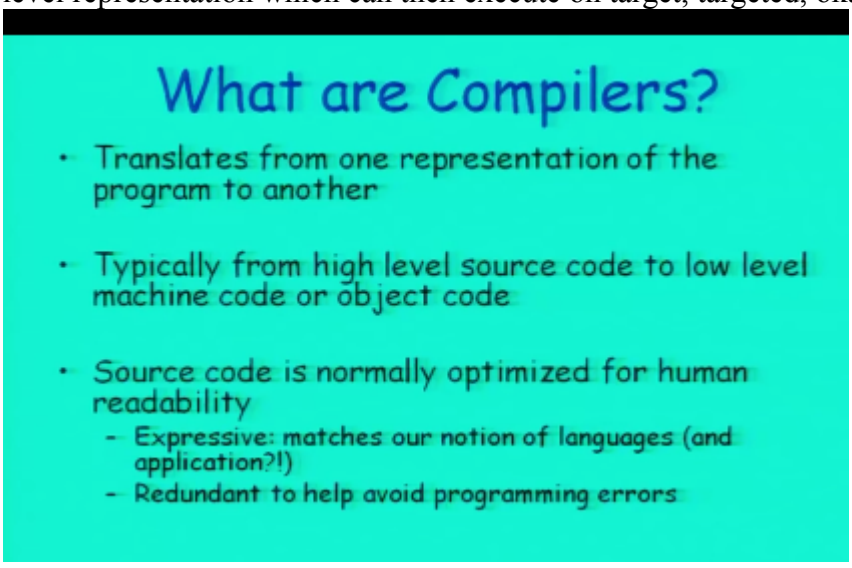
So instead of executable it can also give me a low level language, right. It can also reject compilers, so is that an outcome of compiler? Okay, so it can also identify errors in the program, right, is that what you were trying to say, okay, any other suggestions? Somebody is speaking, yeah, into any target language, so you are saying it need not be an executable or a low-level language, it could be any target language, okay, anything else? Can optimize, what can compiler do? So if the certain property which is being preserved when I am going through these steps, I'm starting with the program and I'm getting some outcome I am doing something I may preserving certain property in the process of the source, but something, so what is the

essence of the program I want to understand. Not, it's a serious issue, I may stop the jokes or something, I mean what you have raised is the important point but you need to articulate, so when you say essence of the program, what do you mean by essence of the program?
Okay, so let me again give you certain scenarios, so I have this program P1 in some valuable language, and then I have the compiler here, compiler is mean the Fortran P2 in some language and what you're saying is P1 and P2, is that doable? So first is this doable? Remember, remember theory of computation, can I take those two programs and prove that they're equivalent, okay, another solution which came was that if P1 and P2 produce same output from the same input then they are considered equally, right, is that what you are trying to say, okay. And what is wrong with that, because we also said that is going to do something, alright, because we already said that once to optimize right so maybe the process which was being used at some level in a and therefore compiler now says that I can change the process itself to make it more efficient so perhaps that seems to be acceptable, so we need to talk about process a little more.
So first is this sort of acceptable definition? That if two programs when they are executed on the same input they will produce the same output then I say that P1 is equivalent within codes or in terms of theory of computation, but P1 is compiled correctly into P, is that okay? What do we mean by executing P1? I never said execute P1, I said P1 produces the same output when the input is given, so maybe there is no machine but I can do a manual execution right, I can go through a flow of the program I can take the C program and say go through the manual execution of the program, right, I may not have a C machines on which the C code without the X, because all these points, okay you are bringing out all good points, okay they're relevant but we need to say what is more relevant and what is it that we picked up as far as compilers are different. So we will not know whether two programs are equivalent, but what we can obviously tells is that P1 and P2 are as far as input and output are the same, okay, but then there are certain limitations, so for example if I say suppose P1 is doing sorting, okay and this does giving bubble sort and this does quick sort, okay, compilers don’t have enough intelligence built into that, so then that they can actually find out what the algorithm is and then replace that by an equivalent efficient algorithm, compilers do not do that, they cannot do that, okay, so they do not understand that obligation, okay, and therefore when we talk of translation we need to understand that when we talk of preserving the semantics when you say it should preserve the meaning what that means is that it is really changing the representations, so for the time being let's not worry about optimization, okay, but what is happening is that it is, when really translation happened, what happens is that you have certain representation in high-level language, okay, so C uses a representation where you say that I'm going to declare certain variables of certain type, okay and then I’ll have some data structures and I’ll have control flow that is how the computation is represented, okay.
When it comes to machine level or when it comes to assembly language or when it comes to different language my representation may be different, but they are essentially computing the same thing as far as input and output is concerned so compiler is only going to change the representation, is not going to try to understand what the algorithm is, and when it comes to optimization it will not try to then replace one algorithm with another algorithm, but we will try to make sure that the same computation by using the different representation of the program or by using a slightly different order of execution can be done more efficient. So for example it may decide that part of the code never gets executed, while somebody may write a code like this if true then S1 else S2, okay and compiler may figure out that this part is always true and

therefore this will never get executed and therefore decide not to email generate code for this part, okay, so efficiency will come only in terms of eliminating part of the code which may not be properly executing or maybe changing certain representations which will say that can I achieve the same computation by doing fewer computations, okay so it may decide that if you say multiplied by 2, and I say oh there's no point multiplying by 2 because that is costly, I may as well add it, okay, so only these kind of changes, these kind of optimizations that can do, it cannot work at the level of algorithms that you have to understand very clearly that you are only dealing with some kind of representation and not some application and try to understand what the application is that is beyond the scope of the compilers, so that is what you have to keep in mind and therefore when we start talking about what a compiler is, we all the time deal with representations and what we say is it translates one representation of the program into another representation of the program, okay, beyond that we do not go input deeper into it and

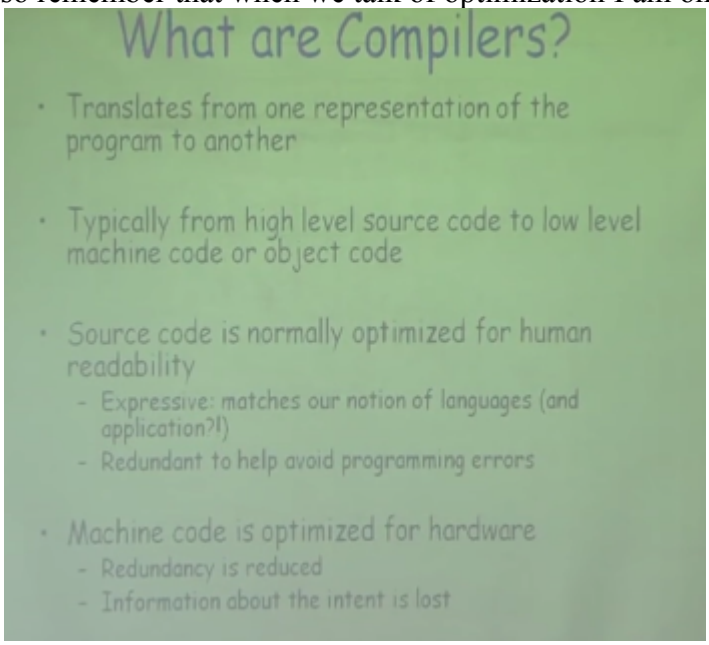


typically most compilers what they will try to do is they will take the high level source language and then translate that into either machine code or object code but that does not exclude the scenario where I can be translating to any higher level language, so I could be writing a compiler say for Pascal to C, okay or C to C++ or C++ to C, okay those will also be called compilers, but traditionally if you look at compiler there is this term is sort of reserved for the scenario where you say that I am taking the height of the language and translating that into some low-level representation which can then execute on target, targeted, okay.

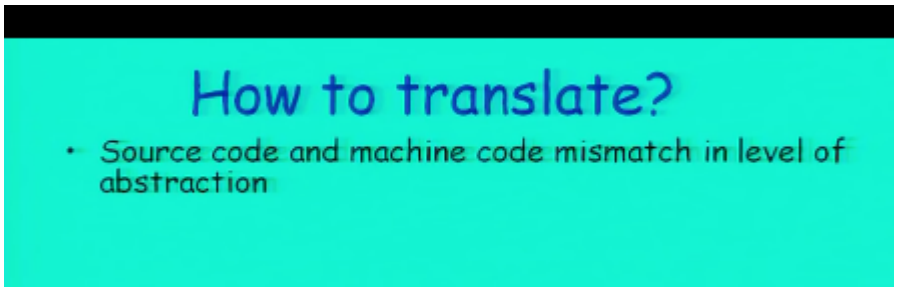


And source code is normally when we write source code, so now we are coming to this point so what happens is when I am writing source code most of the time what I have in mind is that a

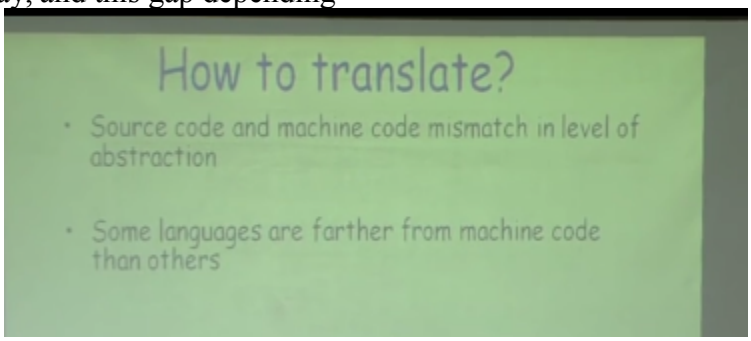
human should be able to read that code and understand it, because we are going to deal with the source code as human beings, okay and any application, okay so we are going to make it very expressive, we will not worry about any kind of optimization, we will not worry about even efficiencies most of us will not, okay, and we will also see of redundancy in the code just to avoid all kinds of programming errors, okay, so you want to write very expressive code but what we would like to do at the machine end is that I don't want any redundancy, I want to remove redundancy so remember that when we talk of optimization I am only trying to remove



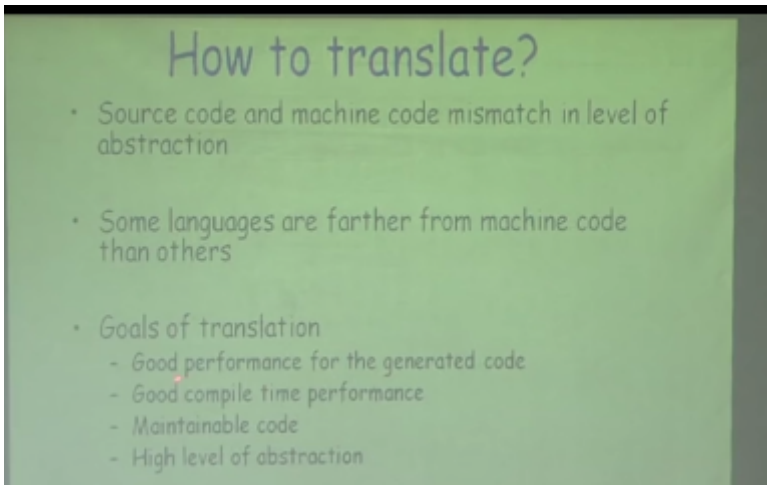
redundancies from the representation and not trying to improve the algorithm, okay.
And information about intent, so whatever information I had about the intent on the application, most of the time we will find that in source code I'll not be able to figure out that intent, so many times when you read the source code you will be able to see oh this program does sorting, okay, when you read the machine code most of the time you will not be able to figure out what this program is, so that intent will be lost in the process of changing representation from high level language to a low level language.
So this really tells you about the functional specifications of what a compiler does, these are the kind of things you have to keep in mind, so how does it translate, okay, now abstractions at the source and machine level are very different, so what are the abstractions typically you may have at the source level? When you deal with the programming language and you're trying to write the program in the high-level language, what are the kinds of abstractions you'll have, you'll normally start with variables, and you will have operators using that intent then the expressions. And once



you have expressions then you would like to put conditionals into it, you would like to put iterations over it, and you would like to have functions and so on, okay, so depending on the richness of the language we keep on making more and more complicated, but basically this is how we're going to stop. What about machine? If you are writing machine code directly, what are the kind of abstractions which are available to be at that programmer? What are the resources you have? So you will have certain locations, so you will have memory and you will say that I am going to put certain variables in it, okay, you may also have resources like registers, you may decide to put two things on the stack, okay.
And then you have opcodes, so opcodes are going to say that I'm going to take certain data from memory, certain data from register, some something from the stack and so on, okay these are my opcodes and then I'm going to have various addressing moves, so I'll say that I'm trying to get certain data from memory but it is indexed with separate register and that information, indexing information is in the register, or I may say I will know the base address and then I'm trying to get something from the base address with respect to or I'm trying to get something from memory where the location is given with some offset with respect the base ends, alright.
So these are kind of abstractions you have, and what the compiler has to do is, it has to translate representation which is at the program level into something which is at this level. So source code and machine code they are going to mismatch and normally some languages, depending upon the language you have some languages are close to the machine abstractions, some languages may be very far from the machine language, okay, so you're going to have a mismatch here, okay, and this gap depending

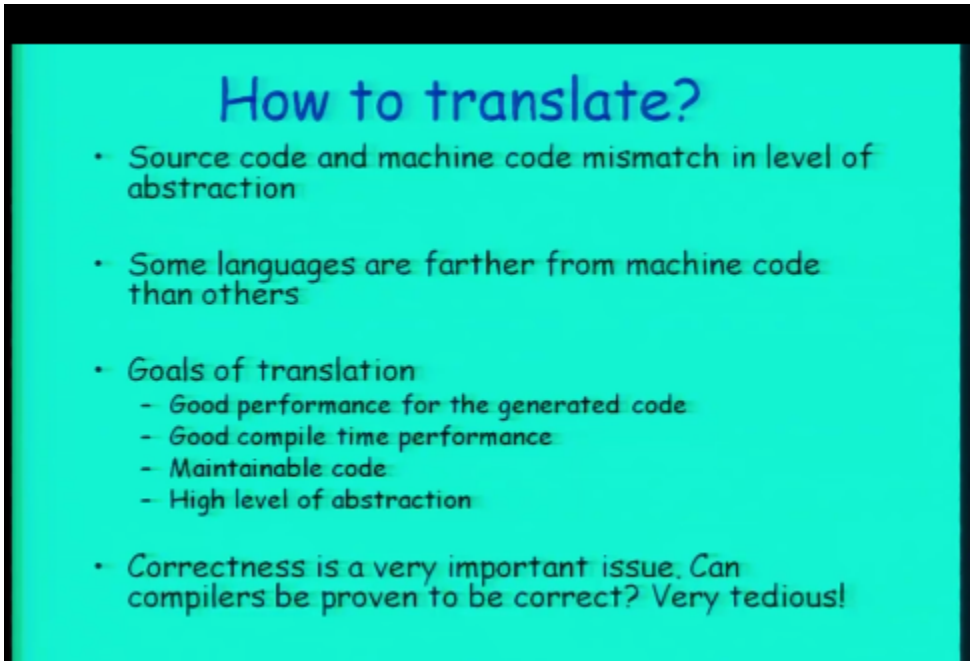


upon kind of language and machine you may have this would be narrow or large, and the question is, how do we then translate it, okay that is what the compilers supposed to be doing so we want to do this translations, okay.
Now obviously it looks like that if I just write it let’s say in one go it may not be possible, so normally we would like to do is when we say it says big jump okay, and therefore before I can make this big jump can I just take a small step and then as I keep taking these smaller steps I finally end up with translation, okay, now each small step maybe a simple step but maybe logically coherent step, and normally compilers are designed so that at time you take one step but keep on moving towards this direction till you have finally you reached there, okay and therefore I need to understand this what these small steps are, okay, if I can understand each of these small steps then I can reach the destination, okay, beginning to make sense, okay.
So what is this board of translation? This is the translation process I will follow and what is the board of this? Okay, one that generated code should be good, it should give me good performance, okay, obviously good compile and performance because if you say that this translation process itself is so tough that it is going to take long time, I mean imagine the



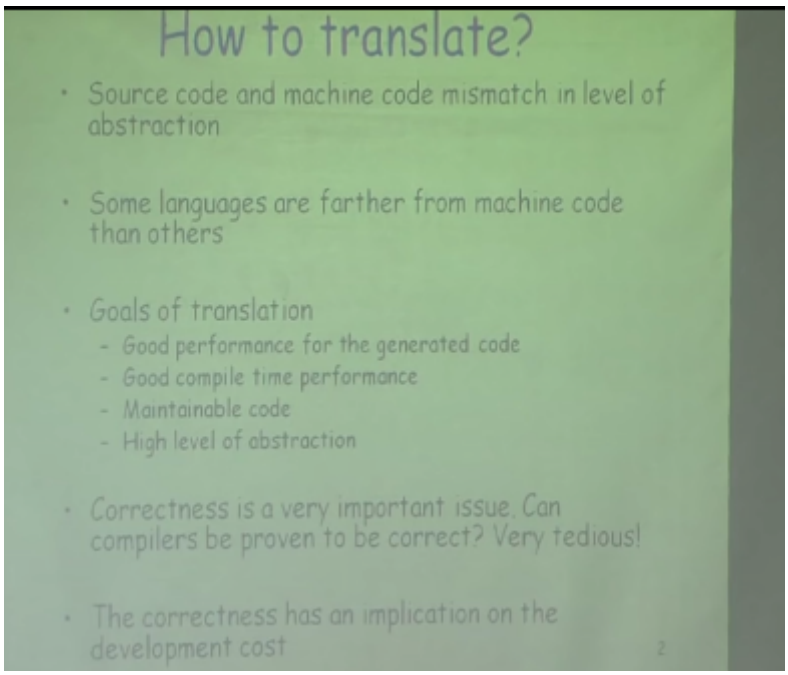
situation where you sitting on the terminal you have finished writing your program or you want to execute it then you compile in the next stage I'm going to take 30 minutes to compile till that situation could be accepted, now obviously no, right, similarly if it compiles very fast but says now we're going to take 30 minutes to execute, okay, same scenario, this scenario will also not be acceptable, so we want good performance for both the generated code and for compile time code, okay.

Now these two issues, we'll take up slightly later, what this means is that, I want to have maintainable code because I may not be generating machine code all the times, suppose I am doing the high level translation, okay, and I say that I want to convert Pascal to C, I don't want to generate C code which further cannot be maintained, every time I don't want to go back and start maintaining my source code, okay, so I should be able to maintain both generated codes as well as I want to maintain my compiler code. Now as a user I may not be able to change compiler but as a compiler writer if a bug is reported then I should be able to handle that, many times you'll say I'm using a compiler than suddenly you find the bug, okay, I can show you even in GCC, we have found bugs, or somebody has to go and type it, so obviously you report it back, but now they say your code is so complicated I don't even remember what to do with it that's it, again that kind of scenario is not acceptable to us, so one should be able to maintain code and one should be able to maintain high level of abstraction as far as compiler is concerned, so as I said these two points will become more clear as we go along, okay, but another very important issue is the correctness, I don't want the situation when I say I compile the program my intent was something so I have written the program with certain intention in mind and when

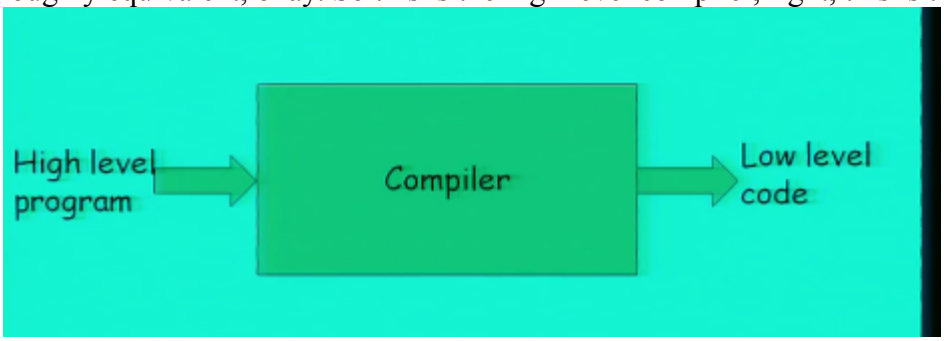


the results come they are entirely different, and why they are different, not because my program has a bug, but because compiler has a bug, compiler is giving me some arbitrary code, right, so that also is a scenario which is not acceptable to us.

So correctness is very important issue and people have tried various approaches to this, one approach in 70s and 80s was they want to prove compiler set, they want to prove that the code they have written is correct, now that kind of thing has not really taken off, people were able to prove small programs correct, but typically how large the compiler is, how many lines of code typical compiler we have? So there is C compiler with runtime system in full support, how many lines of code we think it's there? About lakh, okay, and if I go for something more complex like C++ or Ada I can get a million and if I go some for something more complex like saying I want to support a full integrated program develop in the environment where multiple languages can be supported then I may go into several media, okay, now with that kind of code size, okay, today's technology doesn't, does not permit us or it does not support that I can prove that what I have written is correct. And we have to go through for code testing and this obviously when you try to do a correctness is going to have an impact on the development, because we have to ensure that we have a very high degree of confidence in what we have it written, okay,



something that happens in compiler, so we talk about how compilers are tested, how do we know that the code which is generated from compilers is correct, how do we ensure that this P1 and P2 are roughly equivalent, okay. So this is the high level compiler, right, this is the box, we



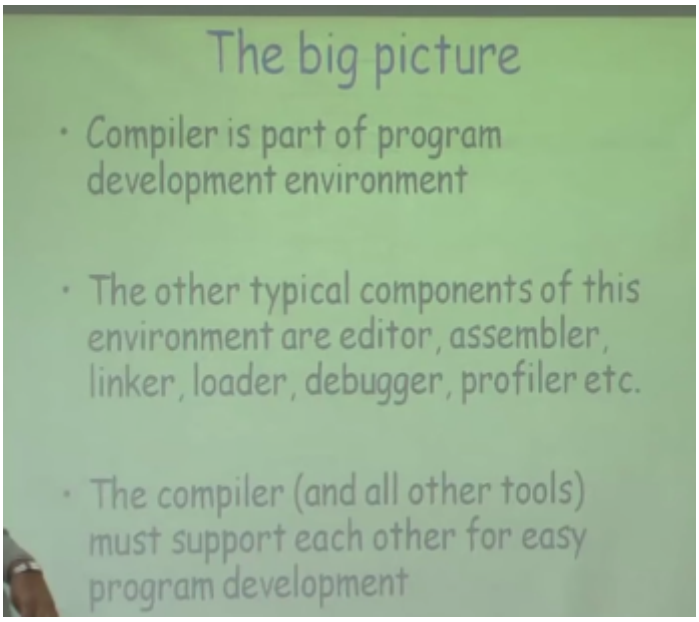
do not know what is inside this box but apparently we know what is the input to this box and what is output of this box.

And what we now need to understand is so here is the high level representation, here is low level representation, this is the box, and I need to understand what are these steps I need to take, these small steps I need to take so that rather than jumping straight from here to here, I can say I can jump from one level to another and slowly keep moving towards line, right. So what are those steps? So let me again give you a scenario, okay and just look at this word, what is the meaning of this word? Backwards or compiler, okay, let me give you simpler thing, so here is a small language, okay, we are in CS335, do we understand meaning of this sentence, how do we do that? So what is that structure? Okay so let me mess up the structure first, you still understand the meaning of the sentence, roughly now there's an error, right, and what is the error? There is a character that we don't understand, okay this is not in English, okay, therefore for any language understanding first thing as far as so I am not using this or human go out to make it but I am trying to say how compilers would like to do it, first thing we need to understand is what is the work of it?

So I immediately have set of alphabets, so when you say that when it comes to English normally I will say that I have these 26 alphabets both in lowercase and uppercase and then I can also use numbers and I have some punctuation marks, so I have this full stop, semicolon, colon, brackets and so on, and that really gives me the character set, okay. And anything which is outside this character set first thing I says is not acceptable, because this is not English, okay, so immediately you put flag an error saying oh I don't know what this character is, this is not in the characters that I know and therefore is an error and you make sure that you have all proper characters of this language, okay, so to understand a language, to understand the representation first I was defined I correctly said, right, what is the next thing I'll do? Spellings, okay so if I gave you a spelling like this, you'll immediately say I don't know what this word means, okay now how do I know spellings? Before spelling do I need to do something, before checking, you did something which are not able to articulate so let me say, now, what happens now? Or what happens now? So somewhere I was able to, while reading this I was able to break this into words okay, I knew what the word boundaries were, okay and what boundaries here was you were saying that whenever there's a blank there is a word boundary, whenever there is a punctuation there is a word boundary, okay, so if I immediately put a character after dot, you did not mind it, you knew that there is a word boundary but if I do something like this, then you will say there is boundary, okay. I need to figure out what the boundaries are, once I have figured out all the word boundaries then only I'll be able to check whether this is a valid word or not, because if I have not even identify word boundaries how can I say this is word or not, okay. And how do I find out words, okay, I have a dictionary, and look at Oxford Dictionary, if any English Dictionary proper nouns may not be there like CS335 which may be a proper noun here, okay it will not be in traditionally but rest of the words will find it, okay.

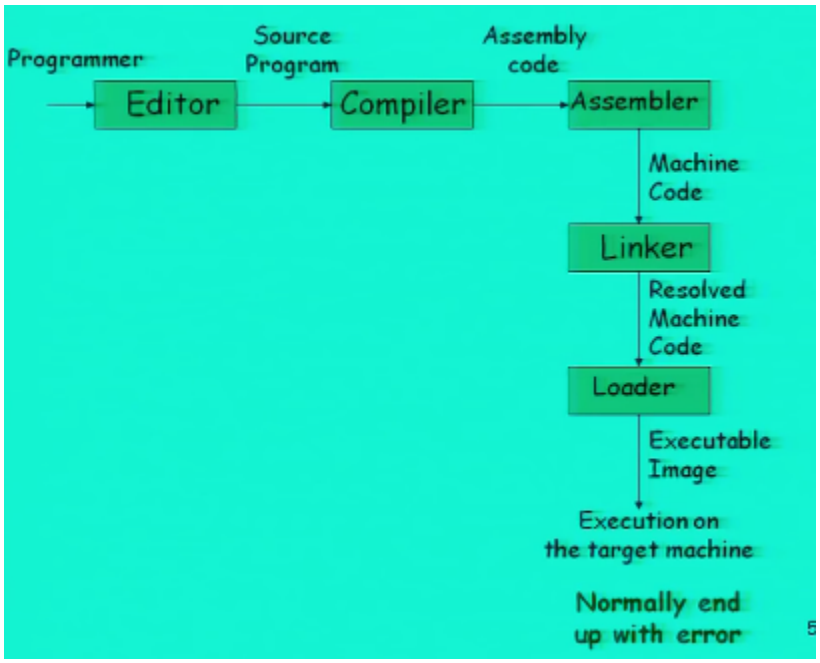
Now do we have so let's go back to programming languages, do I have a set of characters or programming languages? We have, right, do I have rules for finding out what are the word boundaries? Do I have a dictionary of C? Yes, so please give me a reference where I can look at dictionary or C and I can find whether this word is in the dictionary or not. Everybody in the dictionary have seen? Keywords, and so we don't have a dictionary, what we have is a set of keywords which is in dictionary, but then we have rules for construction of valid words, so whatever is not in dictionary I will apply certain rules to this and I'll say that whether this word conforms to the rules I have, so I'll have rules for constructing what are valid numbers and I have rules for construction what are valid words, okay, then I'll say count is valid, this is not, okay, because it does not match any of the rules I had, okay, and that will give me the words, so first thing I need to do is the first step when I start changing representation from here to here first step I need to do is check whether all the characters are valid, break those sequence of characters into a set of words and check whether those are valid words or not, okay. If I can do that then it doesn't matter whether it's a valid sentence or not but at least I have achieved something, right, so this is really the first step we follow but before we move into this first step, okay we also need to look at that compiler really is part of, why do we write programs they're trying to solve the problem and when I try to solve the problem compiler only plays part of the role, they have other things which have to be working with compiler, so let's look at a big picture whereas compiler is part of a program development environment, so after all if I'm writing program I need to do lot more things, okay.

So what are the other typical components? You obviously have to start with an editor after you have reached whatever operating system you have, okay, so you need to have obviously compiler has to work under certain operating system, it needs to have support for that I must

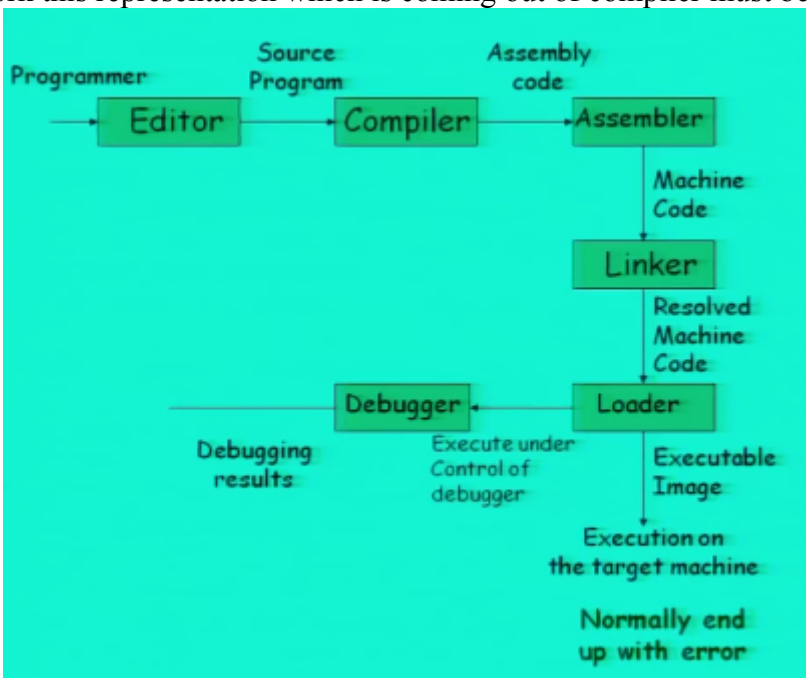


have some editor then I must have assembler, linker, loader, debugger, profiler and so on, okay. So we look at least introduction to all of these and the compiler and all these tools I'm talking about, they must support each other, so for example take a scenario where your editor creates a file and it's not a nasty file, it saves in a different format like this word stuff or take the Microsoft Word, okay, I create something if I try to read that it's not nasty format and now I write the program using say Microsoft Word and say compile this program, compiler will not be able to handle that because it uses some kind of control sequences, some kind of compress representation, okay. So compiler must know this interface, okay and therefore this editor must create files which may compiler can handle, okay, and whatever is the output of my compiler that assembler should be able to handle it so on. So what is the big picture? Big picture is that there is an editor which is being used by a programmer and you have a source program which comes out of this editor but then my compiler should be able to take this source program and should be converted that into an assembly code, so you can immediately see that I am talking of an interface here, okay.

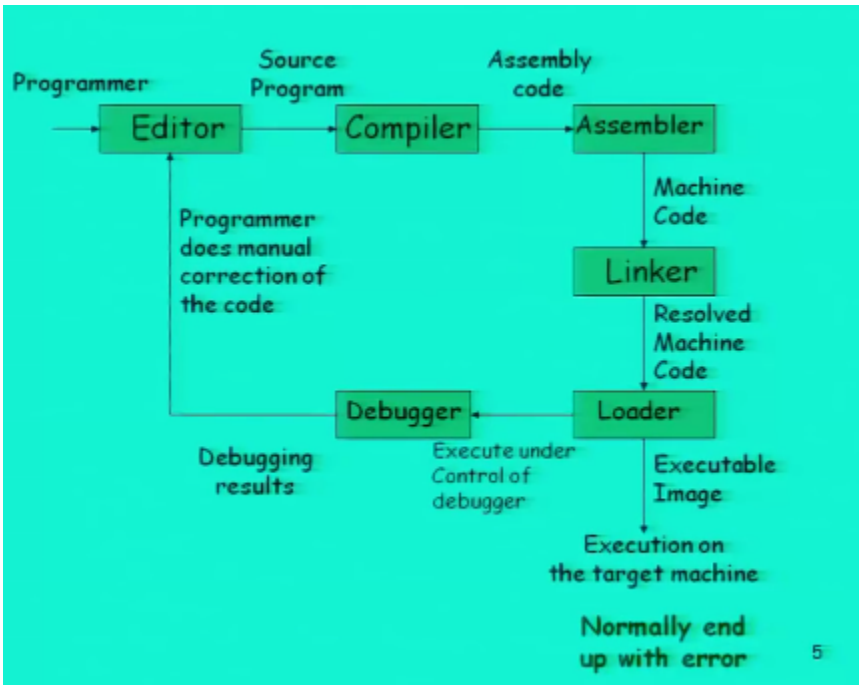
Now what do I do with assembly code, I can assemble it which generates machine code and then what happens I will have to then use a linker because this may be in multiple files and I want to resolve certain symbols so I get resolved machine code and then I am going to load this



into certain machine locations and execute it and what is the outcome of this execution? When I execute the program what happens? There's normally what happens, okay, you get an error, you don't get results, okay, and when you get an error, what do you do? You need to debug your program, so you start using symbolic debuggers, okay, so therefore debugger and compiler again have to work this representation which is coming out of compiler must be understood by



the debugger, okay and then debugger will, the program will be executed under this control and I will get debugging results, so how do I get debugging results, I have certain mental model in mind about how the computation should go about, I have mental model saying and this point of



time value of valuable should have been this, but when I debug it I found this value was different so then I say there is something wrong with the program, I go and start fixing that which is a manual process. And once a programmer has done all the manual corrections then we will go through this, and at some point of time I will get results, okay, but to make sure that I get results I must go through all these sides, okay, so therefore what we need to make sure is that the compiler which is sitting here is able to generate information which all other tools here endues, right, so this is the bigger scenario and what we would like to do is stop here today, and tomorrow start going into this details and variables. Alright, so let's stop here today and again tomorrow the same time.

Acknowledgement
Ministry of Human Resource & Development
Prof. Phalguni Gupta
Co-ordinator, NPTEL IIT Kanpur
Satyaki Roy
Co Co-ordinator, NPTEL IIT Kanpur
Camera
Ram Chandra
Dilip Tripathi
Padam Shukla
Manoj Shrivastava
Sanjay Mishra
Editing
Ashish Singh
Badal Pradhan
Tapobrata Das
Shubham Rawat
Shikha Gupta
Pradeep Kumar

K.K Mishra
Jai Singh
Sweety Kanaujia
Aradhana Singh
Sweta
Preeti Sachan
Ashutosh Gairola
Dilip Katiyar
Ashutosh Kumar

Light & Sound

Sharwan
Hari Ram

Production Crew

Bhadra Rao
Puneet Kumar Bajpai
Priyanka Singh

Office

Lalty Dutta
Ajay Kanaujia
Shivendra Kumar Tiwari
Saurabh Shukla

Direction

Sanjay Pal

Production Manager

Bharat Lal

an IIT Kanpur Production