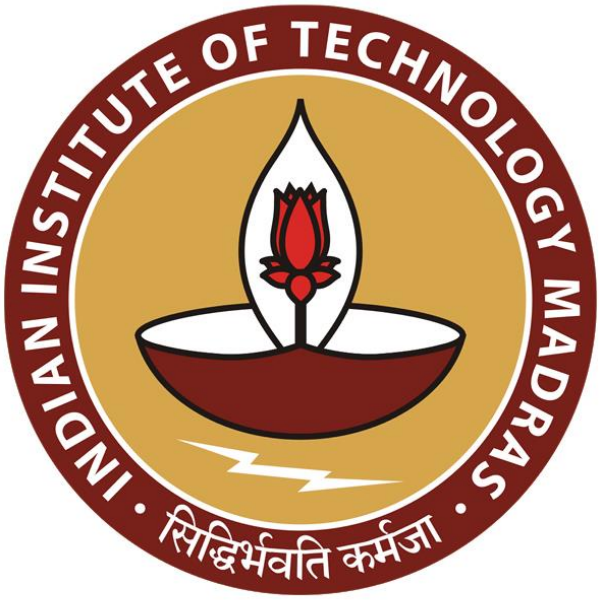


IIT Madras

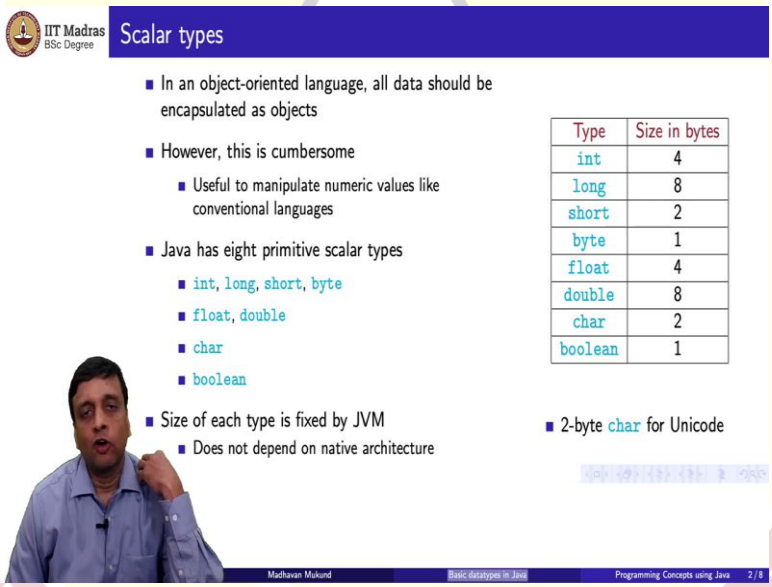
ONLINE DEGREE

Programming Concepts Using Java
Online Degree Programme
B. Sc in Programming and Data Science
Diploma Level
Prof. Madhavan Mukund
Department of Computer Science
Chennai Mathematical Institute

Lecture 8
Basic Datatypes in Java

So, after that quick introduction to Java through one example let us start looking at Java a little more systematically. So, we begin look by looking at the basic data types that Java provides us.

(Refer Slide Time: 00:25)



So, in a truly object oriented language everything should be an object every piece of data that you handle should be an object which is encapsulated with functions that operate on this. But to do this in real life is very tedious and remember that Java came into being in a context where people were already programming in languages like C and C ++. So, it was important to for adoption of the language also to make it a little friendly to use.

So, as a result Java has at its base some conventional scalar types. So, this primitive scalar type in Java consists of the usual types we associate with numbers. So, we have integers we have floating point numbers then we have characters and we have Booleans. So, all of these except characters are also there in a language like python. So, among integers we have 4 varieties.

So, we have the normal int then we have a larger version of an integer called a long we have a smaller version of the integer called a short and then we have an explicit type which takes exactly one byte of memory called byte. In float we have a normal float. So, float is being is used for fractional values and we have one which can accommodate larger magnitudes and a larger range of exponents which is called double.

Char as the name suggests is a character and it stores one character and that is used for text as usual and Boolean as usual is our Boolean data type which can take values true and false. So, one difference between Java and the languages that preceded it is that the meaning of these types is fixed once and for all by the JVM. So, it does not depend on which architecture you are running it as we discussed earlier.

So, the table on the shows us that an int is always 4 bytes. So, remember 4 bytes is 32 bits and if you take one bit away for the sign then you have roughly magnitude 2 to the 31 which is roughly speaking. So, 2 to the 10 is 1000. So, this is roughly one with nine zeros. So, and a long will be now 8 bytes a short will be 2 bytes. So, there is something like 65000 or something like that.

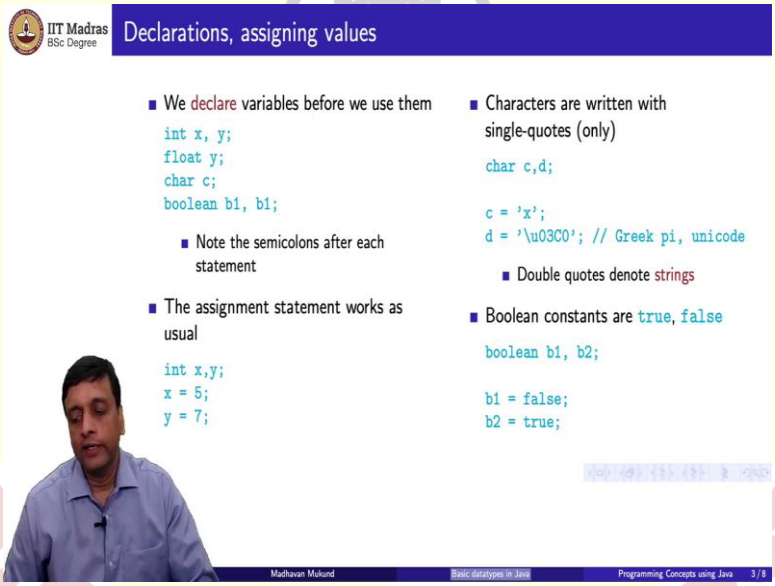
And now we have a single byte variable which is obviously one byte a float will be 4 bytes a double is 8 bytes. Surprisingly and we will ask why a character is 2 bytes. So, normally a character is encoding some value through some numeric table. So, the normal char and conventional languages is one byte but here it is 2 and we will see why. And Boolean of course is just a value is either true or false but we a minimum unit of storage is a byte.

So, Boolean will be one byte. So, the reason that Java chose to have 2 bytes as a default for character is that remember that Java came and be at the time when the internet was already on its way. So, in the mid 1990s the internet had started to be used so there was a need already to display material say for example web pages in multiple languages. So, traditionally different scripts and different languages had developed their own encoding for usage on computer screens and printing and so on but Unicode was an attempt to provide a single encoding for all the scripts and all the symbols in all the languages in the world.

So, it took some time of course to arrive at the particular encoding that people wanted for their language but there was space for everybody because Unicode expanded the space for encoding from one byte to 2 bytes. So, these greatly increase the number of symbols you could represent. So, to be conservative the second byte of Unicode is the same as ASCII which is what we normally use for English for the roman alphabet and then an additional first bit allows us to go to non roman letters.

So, therefore Java uses 2 bytes for a character because by default everything in Java is in unique code. So, this is particularly important as I said to write internet based applications where you might want to render pages written in Korea and Japanese in Indian languages but also in European languages with peculiar accents and so on.

(Refer Slide Time: 04:56)



So, we will just look at code snippets these are not real pieces of Java code because we have not declared all the paraphernalia that goes with Java code as we remember. So, the first thing is that every variable in Java needs to be defined before it is used. So, it has to be declared in terms of its type. So, the typical declaration consists of the type name like int followed by one or more variables which are of that type.

So, this first line says x and y are going to be used in my program and they are both going to be of typing second lens is y is going to be used and will be of type float and so on. So, c will be a character b1 and b2 should be b2 b1 and b2 will be Booleans and so on. So, notice that these are treated as the equivalent of statement. So, each definition or

each declaration ends with a semicolon just like an assignment statement or any other actual actionable statement in Java.

And you can see that you can declare multiple variables of the same type as part of the same declaration. So, you can take b1 b2 and declare them both to be Boolean without writing 2 separate lines Boolean b1 Boolean b2 that will also work but this is sometimes convenient. So, then having done that the assignment statement is the usual statement you use the equality symbol to actually mean variable assignment.

So, the left hand side is a variable the hand side is typically an expression which reduces to a value of the time. So, here we have defined x and y to be integers and we assign the integer 5 to x and the integer 7. Now characters are something new we do not have individual characters as a separate type in python in python we only have strings. So, we have one character strings but we do not have character as a distinct type but in python in Java strings are different and characters are different.

So, we write a character symbol by using a single code it is a single quote because double quotes actually would denote strings in that in Java just like that. Now inside the single quote if you are just using a normal character like a b c d 1 2 3 4 you just write it as you should. So, this will assign the character x to the variable c which is defined as a char. This funny notation allows you to assign Unicode symbols which you may not be able to type on your keyboard.

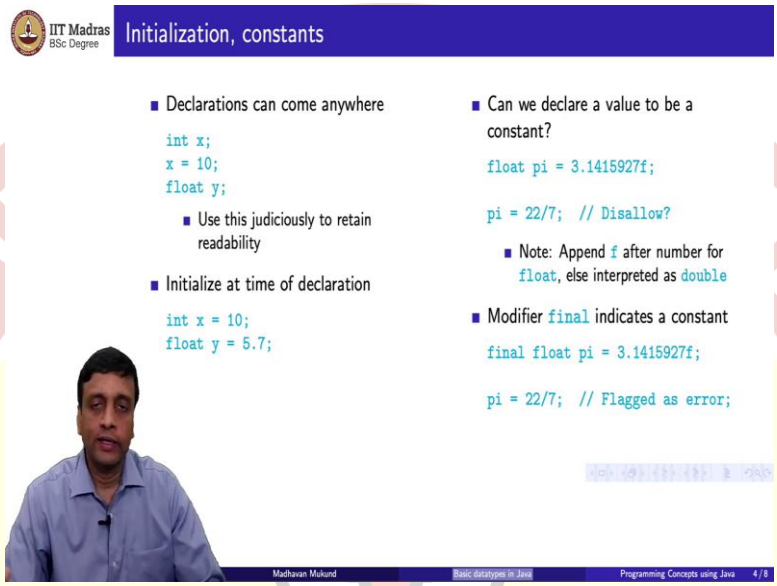
So, this is Unicode this is 2 bytes. So, these are actually written in terms of. So, a byte can be broken up into 4 + 4 bits and then 4 bits you can write a hexadecimal number a base 16 number. So, 2 bytes is a sequence of 4 base 16 number. So, this is 03C, C is the number that stands for 12 and base 16 10 is a 11 is b 12 is c. So, this is base 16 this is x. So, this particular code for instance 03C0 in Unicode stands for the cap letter pi that we are familiar in Greek.

So, you can assign to a character variable a character from any valid language which has a Unicode representation and if you are using conventional things you just use a convention. So, Unicode see you notice that the description is a long thing it is backslash u followed by 4 hexadecimal digits but denotes one character then it will always take 2

bytes. So, the Boolean constants in Java are called true and false but they are written with a small t and a small f.

So, these are different conventions in different languages. So, Java uses small cap small letters for true and false.

(Refer Slide Time: 08:36)



In many languages all declarations have to come at the top of the program in Java actually you can intersperse and introduce declarations in the middle. So, you can write something like this you can define an integer x assign it a value and then introduce a variable float. Now this is not necessarily a great programming practice from the point of view of readability but you can choose to introduce variables where you think they are needed.

So, sometimes I mean it is like you are reading a book you are reading a book and suddenly you come across a character and you do not remember this is a new character or it is a character rule introduced in chapter one you have forgotten about. So, some books will actually provide you like an index of characters and say this person first came on page 4 like. Similarly here after 100 lines of code if you suddenly see a y the question is, is this a y which is always there and this is being used for the first time or is it I mean.

So, what is the status of y. So, in that case it might be useful to actually define the y there with the comment. So, that you know that at this point I needed a new variable called y and so I am using it. Of course Java will not allow you to redefine the same

variable. So, if you have already used a y you have to use something else but you can postpone the definition or the declaration of a variable until it is going to be used to make it easier to read.

But sometimes can be counterproductive because up front you do not know what we deserve. So, it is a question of taste which you prefer to do. So, one useful thing you can do is we can combine something like this into a single state you want to declare a variable of type integer and you also want to initialize it to a value right up front and this is very often the case for example when you use a counter or something we want to declare a counter and say set it to zero.

So, why should we do it in 2 parts right up front when we declare it we can assume that it is initialized. So, that is how this initialization does. So, you can take you can combine an assignment statement with a declaration and this becomes an initialization. So, this is actually a small typo as we will see. So, sometimes you want to declare something initialize it and say it does not change.

So, for instance suppose you are going to use some constant like the mathematical value of pi in your program you would not like later on for pi to be redefined ok accidentally or intentionally in the program. So, how do we make sure that this pi that is initialized here cannot be redefined later. How do we declare that this is actually a value which should be remain a constant.

Once it is defined it can only be re-initialized by rewriting the program and recompiling it but in the course of running the program it should not be assigned and do that a small aside. So, this is why I said it is a typo there should be an f. So, Java has some conventions about floating points versus doubles. So, if you write a number which just has a decimal point in it by default Java will assume it is a double.

So, it is at least the current version of Java the one we are using Java 11 assumes it is a double. So, if you want to insist that it is a 4 byte float then you have to suffix it with the small f yeah otherwise the compiler will come. So, that is what the small f is doing its telling Java to treat this number which is ambiguous as a float or a double not as a default double which is 8 bytes but as a float which is 4 bytes.

Anyway coming back to our question our question is how can we stop this from being redefined. How can we stop this float from being redefined. So, what we have to do is we have to up as in everything else in Java there is for every problem in Java there is one way to solve it which is to add one more word. So, we saw that we could not create a function without creating an object. So, we said okay let us call it static.

So, here we want to say here is a variable that I want to define I want to assign a value but I do not want anybody to touch it. So, I want to mark it as being something which is not to be modified and that is given by this thing called final. So, you might think that you could have used the word like const or constant or something but anyway. So, this is a choice which Java designers made and we are stuck with it.

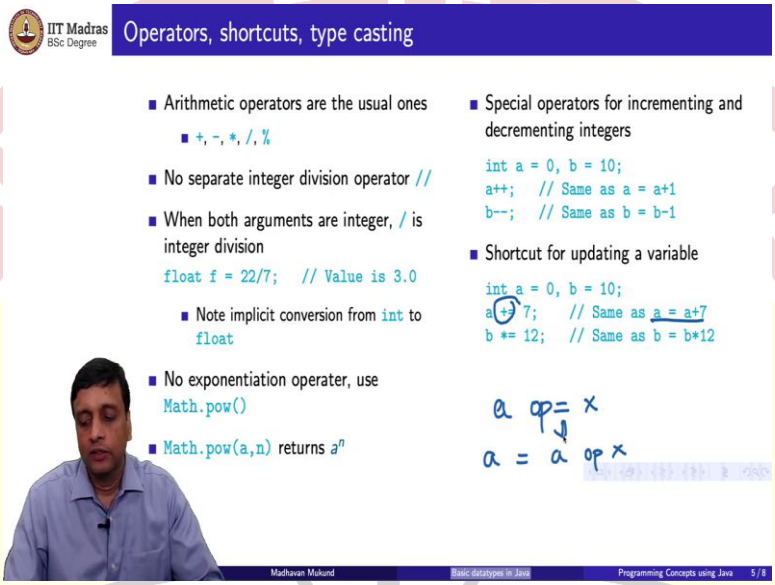
So, final means that this is in some sense the final value of this variable. So, final is a modifier it is not telling us anything about the type of pi it is just saying that whatever is being assigned here cannot be further modified. So, you combine a declaration and initialization and this modifier final and you get a value which is effectively a constant and now this is quite useful.

So, pi obviously is a usable constant we are going to use pi all over the place in mathematical formulas you do not want to keep writing this long expression at the same time you do not want somebody to unintentionally update pi to be 3 somewhere in the middle and then you get. So, this is undoubtedly something which is useful. So, now what happens is that if you declare the final then indeed when Java comes to this line you can try this out you can just embed this in that nothing else that hello world code which was there in the first module. So, that works.

So, you can just embed this definition in that hello world module and try to compile it and the version. With this version will if you have this reassignment pi after it was declared final the compiler will actually give you an error. So, this is the key, the key is that you want these errors to be caught before you run your code when you run your code if somebody accidentally updates a variable you have to go back and debug that and you have to understand what happened.

Whereas the compiler is told this, this variable should never be modified then whenever the compiler sees an assignment which attempts to modify it, it can catch it and tell you something went wrong. So, as far as possible this is the kind of philosophy of languages like Java put a lot of machinery into the language. So, make in some sense make the program writing part little more cumbersome in order to make it easier to make sure that the code adheres to what you want to make it easier to control.

(Refer Slide Time: 14:40)



So, what can we do with numbers we can do the usual thing we can add subtract multiply divide and this as usual is modulus or remainder which is also there in other languages but one thing which is not there in Java is a separate integer division. So, python has this double slash which is integer division. So, in Java actually if both arguments are integers and you put a slash it is treated as integer division if it is not is treated as float.

So, in fact when you do 22 by 7 you might have thought that you would have got an approximation of pi but because Java treats this as an integer division it actually thinks of this as 22 by 7 as a quotient 3. So, the value that is assigned to this float f in this particular assignment is actually 3.0. Now there is something else going on in that because 22 by 7 is an int.

So, it is the integer 3 is being assigned to a value of type float. So, there is an automatic conversion from into float. So, in general int in the; context if it is supposed to behave like a float. So, for example when you multiply and int by a float it will convert into a

float. So, in any context in which because in a sense value wise every end can be thought of as a float not every float can be thought of as name.

So, if you are requiring of an end and you pass a float you will need a trouble but if you are requiring a float and you pass an in then Java will happily convert it similarly you have long and double. So, both are 8 bytes but now the problem is that sometimes I might end up having problems like I take a long and I use it with a float. So, this conversion will happen but the value may not get preserved.

So, it is not always guaranteed that the integer value. So, an 8 bit 8 byte integer value can be represent, faithfully in a 4 byte float okay. So, there will be some loss of accuracy you have to be a bit careful but in general if the numbers are compatible in an arithmetic sense Java will do an automatic type conversion. So, Java you would have noticed here does not I did not mention an exponentiation operator Java does not have one.

So, you have to use a power function called pow which is there in this math class. So think of this as a static class like system dot out so, system. So, it is a class which exists. So, math dot pow you do not have to call it separately you do not have to import it or anything is there. So, math dot pow will take 2 arguments the first argument is the bottom part second argument. So, a comma n means take a and raise it to the power n.

So, Java inherits from languages like C and C++ these shortcuts for some of these arithmetic operators. So, one thing that we often do is to increment a counter count equal to count + 1 or in a while loop we might say i equal to i + 1 or j equal to j + 1. So, this happens sufficiently often that the authors of C decided there should be a shorter way of doing that. So, they introduce these 2 operators for incrementing and decrementing called ++ and

So, if I define a as an integer and b as an integer and I say initialize them it does not matter what I initialize into if I say a++ it is just a shortcut for a equal to a + 1 and b-- is same as b equal to b 1. So, you can sought of think of now what the language C++ means it means the language which is one step beyond C okay that is the kind of inside joke in the name C++. So, this is one shortcut that we have we have this automatic incrementation operator.

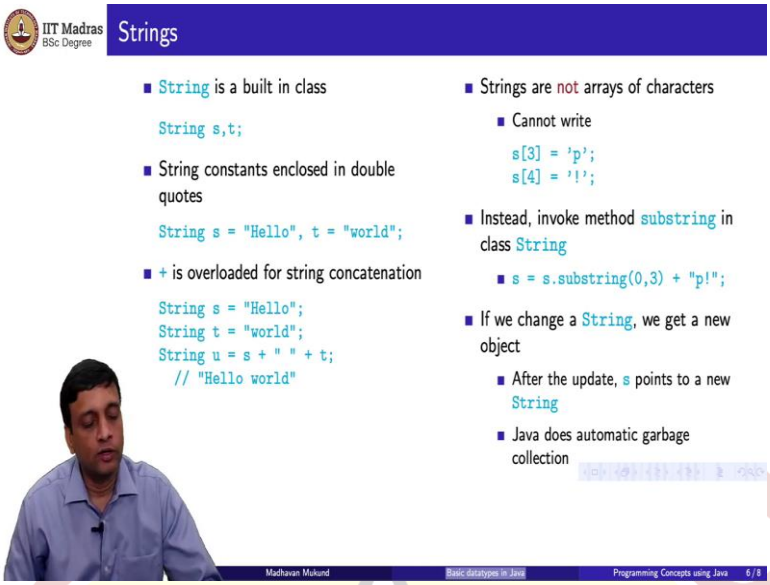
Now the other thing that we often do is to take a variable and update it. So, if you are for example scanning an array then you might say sum is equal to sum + a of i if you are trying to add up all there. So, the same sum here and the sum there. So, you want to take the current value of sum and do something to it apply an operation with a second term. So, this is now shortcut like this. So, if I say a equal to a + 7 which is like sum equal to sum + a.

I can combine this collapse it will say a + equal to seven. So, a + equal to 7 is just a shortcut for a equal to a + m and the reason I am doing all this is because very often in Java examples you will come across the shortcuts and you should not be I am not necessarily saying that you should use the shortcut. So, this is good style but this is frequently used by people who write code.

So, if you look at examples which you will see in the internet or some of the examples that we might see in this course this kind of thing might come up and you should not get confused. So, a + equal to 7 says that + is the operation that is being applied to a and the second argument is 7. So, it is just. So, you should unpack this. So, you have a opt equal to x. So, this means a equal to a of x where op is the operation it could be times it could be it could be any binary operation that involves a on the left hand side and argument x on the hand side.

So, this is the short term a of equal to x is same as a equal to a of x. So, for example b star equal to 10 is b now the operation is star. So, b equal to b star 12. So, this is how you should read it. So, for any binary operation of this kind you can collapse this update when you are updating one variable by applying this binary operation with another quantity you can collapse it into a single step.

(Refer Slide Time: 20:31)



So, those with the scalar type, so now let us start looking at built-in objects. So, the first built-in object that we use and which is something quite useful to us is string. So, remember that a string is a sequence of characters normally. So, we had a separate character type char which was a kind of scalar it was not an object. Now we have string which it is an object it is built in but it is given to us it is a public class and it is an object.

But we it is slightly different from other objects we will see because you can just define a string like any other type by using the name of the type. So, you just say string and you will assign it in the usual way by using double click. So, this is how you create a string. So, you do not have to go through a kind of object oriented exercise of creating an object like when we did it even in python when we wanted to create a point we had to call it with the bracket to indicate that there is some process of constructing a point.

So, it is very much like normal strings the way we do it anywhere we assign the name to a value of type string and a value of type string is a sequence of letters in double quotes. So, as in many other languages including in python you can combine 2 strings together using plus combine in this case means concatenate put one after the other. So, s is hello and t is world then if you concatenate it and add an extra space in between then you will get this output hello followed by space.

So, plus for strings is string concatenation very much like it is in python. So the paradox in Java is that though a string is a sequence of characters you cannot extract characters

very easily from a string you cannot index it. Like even in python you can take a string and think of it as a sequence of letters and you can take out the letter at I and so on. it will be a one element string but nevertheless you can index out parts of the string in python this is not in Java.

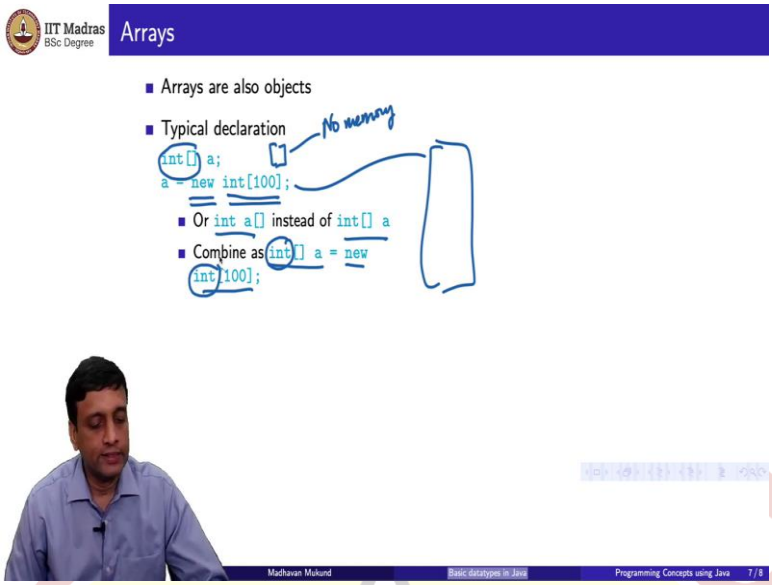
This is not allowed you cannot take s and say that position 3 is a p you cannot say that s4 and you cannot change it you cannot extract it both of these are not. So, if you want to actually update then you have to do something like even in python this is true that strings are immutable you cannot take a position in python in a string and update you can do it in a list you can say l square bracket i equal to something a new value you get there but you cannot take s square bracket I equal to something and something that is immutable.

So, similarly in python if in Java if you want to actually update a string you have to reconstruct it. So, if you want to convert hello to help then you take some part of this thing and then you add the rest so you use the substring operation to take out the parts that you want and you use the plus operation to add the parts that you want to add. So, what has this done this has taken an s pulled out a substring and done something to it and reassigned it back in x.

So, this is really created a new string object. So, in python speak strings in Java are immutable. So, what we will get after this is we will get a new object. So, s will now point to a new string in memory it is not the old string. So, this will of course mean that whatever was there in the old string is no longer reachable from our program. So, this is what constitutes unreachable storage what we call garbage but remember that Java has a garbage collector.

So, it does automatic garbage collection. So, this kind of accidental wastage of memory is not a problem because it is designed to be restored automatically by the garbage collector.

(Refer Slide Time: 24:24)



The other thing that we are used to is some kind of a sequence. So, in python we start with lists in almost every other language we start with arrays. So, the difference between a list and an array is an array is usually a fixed length sequence of a fixed type. So, all the elements in an array must be of the same type a list in python in general is flexible you can grow it you can shrink it and you can put different things.

So, the first element of this could be a float the second could be a tuple the third could be a dictionary and so on. So, here an array is like in many other languages or by definition what an array is a block of code a block of memory containing objects of the same type. So, in Java these are objects. So, you would first define it and then you would instantiate it. So, you define the type of a and then you instantiate it.

So, this is one way of saying that the name a points to an integer array. So, int with so this is an open bracket square bracket it may not be close very clear but it with a pair of empty bracket. So, a is not an int but it is an array of int and then when you actually want to allocate an array to a you use this thing. So, this new we will see the general way in Java of creating objects but in particular we create a new array of integers of size 100 and assign a to point that.

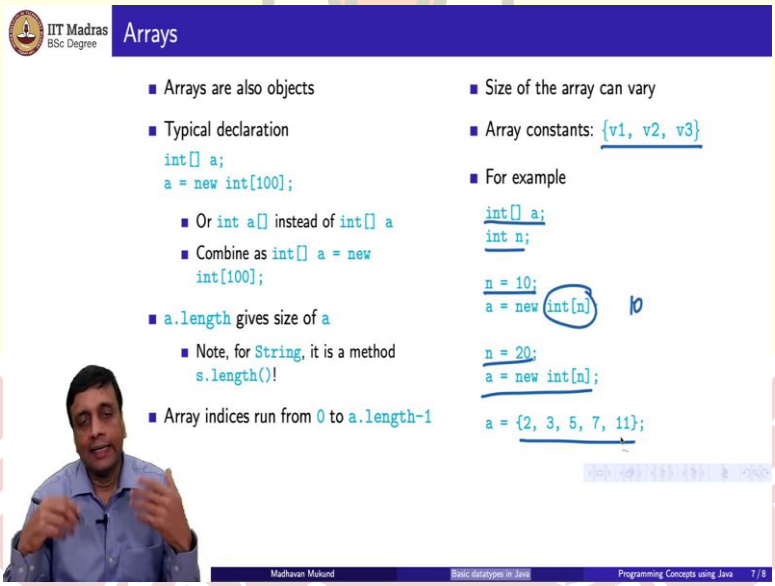
So, at this point so no memory is used and at this point a is pointing to a block of memory at the point when it is created the block size is fixed as 100. So, Java allows this and this has 2 alternative ways either you can attach the square bracket to the name of

the variable or you can attach int. So, it depends on which style you prefer and which is more easy to read for you.

So, if you think that the type is more important, then it is important to distinguish int b from int array a then this is better. As opposed to saying int v and int a b like this where the modifier that it is an array is hidden inside the thing. So, it looks like you have 2 it is declared it is a question of taste which of these 2 you prefer but Java allows both and of course you can combine the 2.

So, this is badly formatted but you can start by saying that int a and then like an initialization initialize it to a new array of size 100. So, notice that there is a little bit of redundancy we will come back to that. So, you have defined it to be an int here and you again say it is an integer. So, I say that it is an array of some type and I want at new array of that type of this size.

(Refer Slide Time: 27:28)



So, you say that type twice and later on when we look at objects and all that we will see why this is the case. So, for an array we can extract its length now this is very useful because if we write a function which sorts an array for instance typically we need to know how many elements there are in it and if you do not have access to the length of the array we have to keep passing the size we have to say sort this array and it has hundred elements sort this array it has 200 units.

So, it is very convenient to be able to take an array in fact find out its size by looking at it. So, in python certainly you can do that for any list you can find its length and similarly here for an array you have this quantity a dot length for an array a which tells us. Now there is a sort of mismatch here because string also you can extract its length and for that you use a function called length.

So, strings have a kind of a method or a function which gives you the length for arrays it is not a function it is just it is like a instance variable you can think of it that way. But whichever it is you can find out the length of an angle and in Java like in most languages you do not have a choice about how to index an arrow. So, if you have. So, many elements in your array n elements in your direct the indices go from zero to n 1.

So, in particular they go from zero to a dot length minus one this is exactly the same in a python discussion. So, because arrays are objects they are actually dynamic. So, the name tells you the type of array but what it is pointing to can change from time to time. So, it is not a static object. So, if you go back to our first week's understanding of where these things are defined.

So, they are not necessarily sitting on the stack perhaps but you have to be able to flexibly define them. So, here is some kind of things that we can use. So, one thing is you can write out an array explicitly. So, just like in you can write out a list explicitly using square brackets and Java allows you to write out an array as a sequence separated by commas with curly braces.

So, let us look at these 2 things in the context of some kind of junkie code. So, here I have defined a to b an array of bin and I have defined an integer a. Now I assign n to be 10 and now interestingly I tell you that a is a new array of size n. So, notice that at this point n is flexible. So, but when I execute this code n is known so before I run this code I do not know how big a is going to be at this point but when I run this code I know at this point that n is 10. So, this will give me an array of size 10.

Now this is useful for inside a function supposing you are writing something like merge sort and you need a new array which will merge 2 lists or 2 arrays that are given to you into a new array. Now the new array size will be dependent on the input array size. So,

you can get the input array size and then declare a new array to be of the fixed size rather than picking some arbitrary magic number like 10000 and saying I will merge it I mean you do not have to be wasteful or you do not have to be kind of predicting what will be.

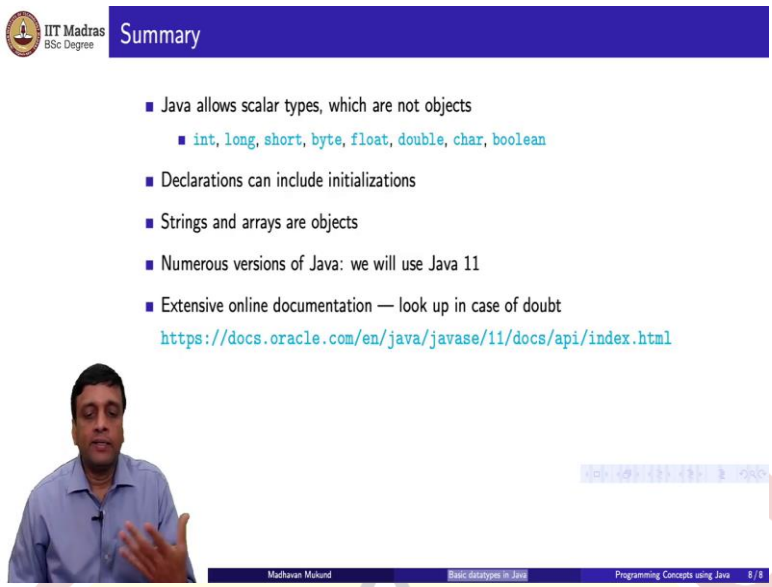
So, it is useful to be able to define these arrays flexibly at the time of this new statement the value of n must be known but other than that it is flexible. So, now I come down and I change n to 20 so I can now reassign a to be a new a different array of size 20. Now again like the string case when we have updated the string this is the case where this will now throw away the old return so the old array is gone.

And now I have got a new array. So, this will of course mean that garbage collection knowledge happens. So, we will come back to arrays in more detail but this is just to show you some basics because it is difficult to talk about for loops and arrays without talking about each other. So, you need a basic understanding of both and then we can do these in more detail.

So, one more thing you can do is like we said you can write as array constant. So, I can take this array and instead of asking for a new block of storage I can say I want this particular block or store I want an array and its length is implicit because they are given five values and the values are also now in some sense pre assigned a0 is 2 a1 is 3 and so on. So, these are different ways in which you can work with arrays some of which are slightly different from other languages.

In particular this idea that an array is statically declared as a type but dynamically allocated as storage. And the dynamic allocation can be of a size which is determined at runtime time. So, these are important things.

(Refer Slide Time: 32:03)



So, to summarize although it is otherwise very faithful to the object oriented world it has made some compromises Java and one of the compromises is to allow some scalar types. So, we have these numeric types int long short byte float double we have a character type we have a Boolean type and these are not objects. These are just this usual kind of scalar types that we see in other languages and we also see them in python. Though in python we do not declare them but they are still there. So, they operate you can use operations on them and so on. Then we have to declare these variables before we use them but we can at least do a combination of declaration initialization.

So, if we are going to use it in a particular context we can combine the 2 statements strings and arrays are objects. So, they are basic types in the sense they are given to us but they are not scalar types. So, in particular we looked at a couple of functions we looked at substring for example. So, string comes with a whole lot of functions and it is very difficult to you know exhaustively go through all the functions without boring you.

So, we will use a function when we need it. So, you need to have this habit of being able to go and look up the documentation whenever you have a doubt you cannot be asking all the time saying how does this work how does that work you have to be able to look up the documentation and figure it out. So, Java is an evolving language and it keeps changing. So, I think the latest I have seen is something called Java 17 and even the numbering convention has changed.

So, the numbering convention is quite bewildering and it is very difficult to keep track of how old or newer version is just by looking at its number. So, we are using a version called Java 11. Now it looks like if we are on 17 today then 11 must be really very old but it is not that old I mean 11 is a couple of years old. So, for each version of Java there is a corresponding exhaustive documentation online.

The changes from one version to another are not huge they are relatively minor and at the level of this course they should not matter. Java 11 is widely available on various systems. So, that is the one we have picked. So, for example if you use a standard Linux distribution on Google 2 or whatever Java 11 is the one that will be installed. So, it is not very easy always to keep the latest version installed because Java as I said keeps coming out with new version.

So, we have for this particular iteration of the course we are sticking to Java 11. And here is the link. So, basically you search for it on the net you will find it but Java was originally a language developed by a company called Sun Microsystems which was a very prominent manufacturer of graphics workstations and so on. But at over a period of time sun workstations kind of went out of business and this has been acquired by oracle the database company.

So, therefore it is in oracle domain that you will find the Java documentation. Notice that you have different documentation for different versions. So, we are looking at Java 11. So, do take a look at this sometime it is not very easy to parse but if you are searching for something you will find it.