IIT Madras

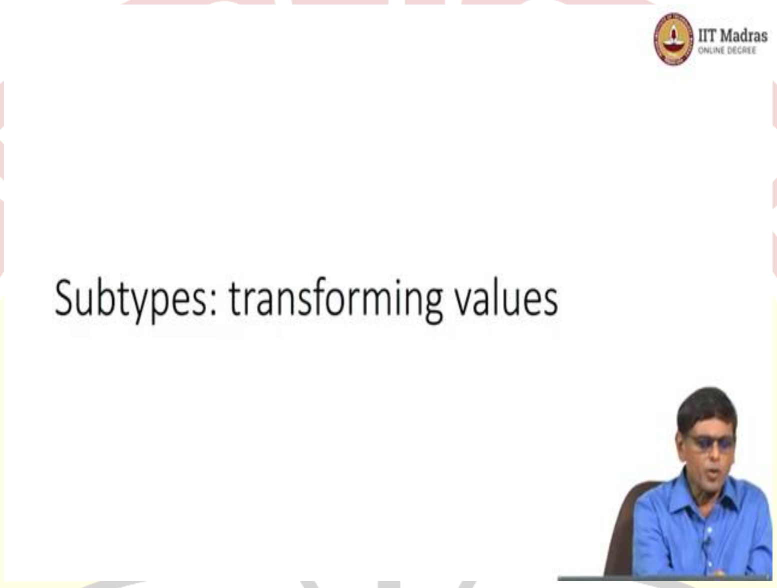
ONLINE DEGREE

Computational Thinking

Professor Madhavan Mukund
Department of Computer Science
Chennai Mathematical Institute
Professor G. Venkatesh
Indian Institute of Technology, Madras

Transformation of Sub Data Types

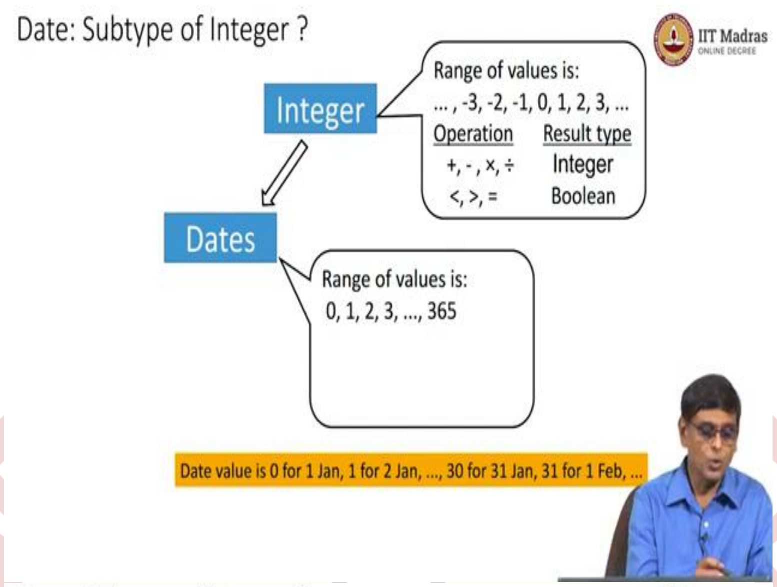
(Refer Slide Time: 00:14)



Subtypes: transforming values

Now, sometimes it is not so obvious how to create a sub type, I am going to give you a few examples of subtypes where we may have to do a little bit of work to create the subtype carefully.

(Refer Slide Time: 00:24)



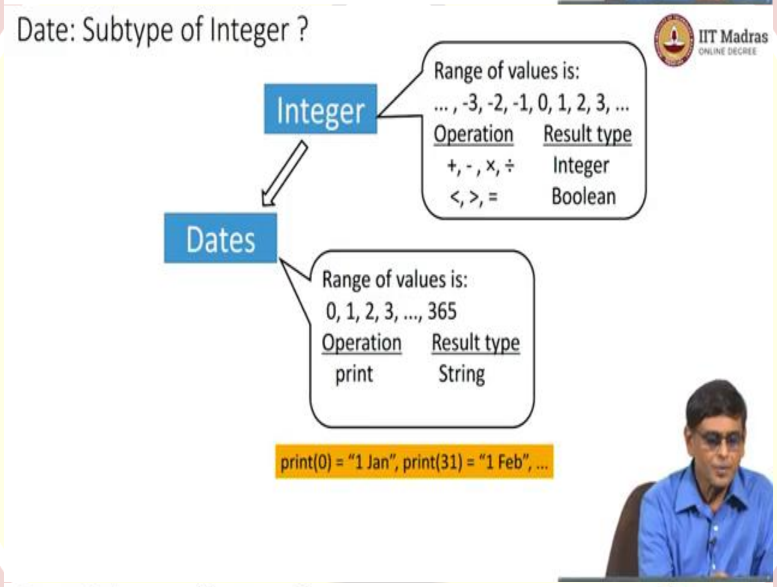
Date: Subtype of Integer ?

Range of values is:
… , -3, -2, -1, 0, 1, 2, 3, …

Operation	Result type
+, - , x, ÷	Integer
<, >, =	Boolean

Range of values is:
0, 1, 2, 3, …, 365

Date value is 0 for 1 Jan, 1 for 2 Jan, …, 30 for 31 Jan, 31 for 1 Feb, …



Date: Subtype of Integer ?

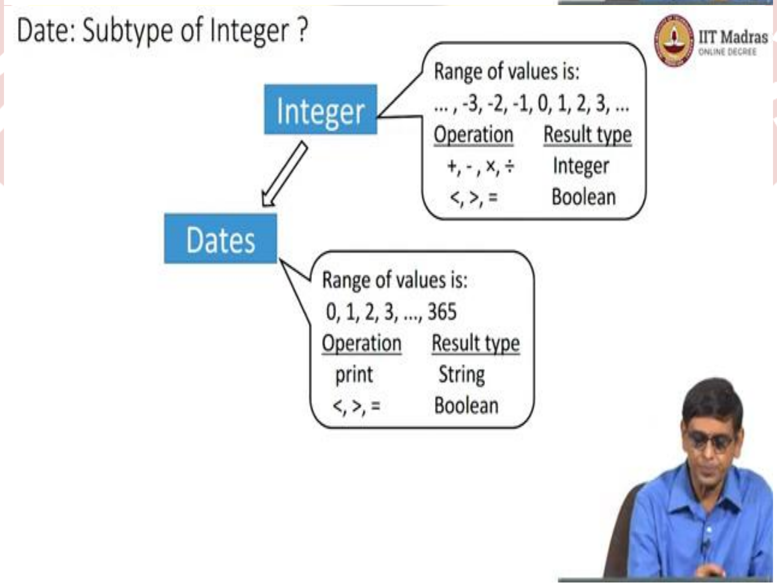
Range of values is:
… , -3, -2, -1, 0, 1, 2, 3, …

Operation	Result type
+, - , x, ÷	Integer
<, >, =	Boolean

Range of values is:
0, 1, 2, 3, …, 365

Operation	Result type
print	String

print(0) = "1 Jan", print(31) = "1 Feb", …



Date: Subtype of Integer ?

Range of values is:
… , -3, -2, -1, 0, 1, 2, 3, …

Operation	Result type
+, - , x, ÷	Integer
<, >, =	Boolean

Range of values is:
0, 1, 2, 3, …, 365

Operation	Result type
print	String
<, >, =	Boolean

So, the first thing is what is date? We have this thing called date on the cards. So, the date has something like say for example 4th June or a 31st January or something like that. Now, what is that? Is that a string? I mean, one can think about 4th June as actually a string it should, you can think about it as a string. But, the problem is if you write it as a string then many operations that we want to do on dates, so we want to compare for example, one date with another, want to check whether one date is less than other date.

You cannot do that on strings, at least we did not define what less than means for strings we only defined an equal to for strings. So, therefore it might, because we are doing comparison you might add something to a date also, you might check what is 3 days if today is 2nd of February, you add 5 days then it becomes 7th of February. So, you may want to add some number of days to a date. And then get some other date. So, operations on dates are looking like of course, multiplication does not make sense, but some operations you may want to allow for dates which are like integer operations.

So, it would be nice to make date into an integer. But, how do you do that? Because, what you have written there is, 31st of January which looks like a string. So, a simple way of doing that or a trick, to use to do that basically is to let the day basically, the date basically carry a value which is the day in the year.

So, assume for example, the January 1st starts at 0, then January 2nd would be 1 and so on, you want like that and you would get 364 for the last day in a normal year. But, if you have a leap year as we discussed you could get 365 also. So, let us, let the range of values to be 0, 1, 2, 3 up to 365 and then for January 31st for example, you will have 30 as the value and for February 1st you will have 31 as a value and so, you can write one value which for all, the date.

Now, what is the nice thing about this is, it becomes an integer and because, it is an integer, you can do integer operations on them. Like, you can add for example, two dates now you can subtract two dates you can do comparison, you can check whether one date is less than other date, you can do all that.

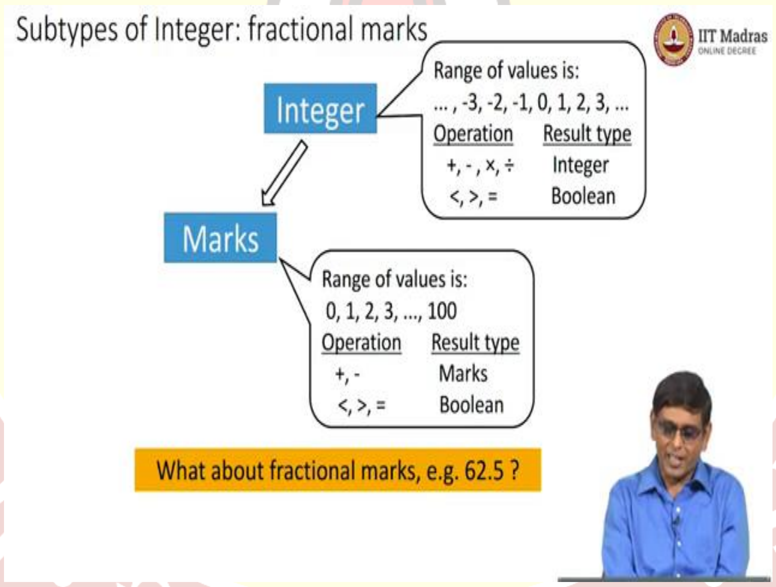
So, but you know when, it is not very nice because, if I print 0 for example as the output somebody asks me what is the date I print 0? Nobody will there, we have to interpret it so, I must be able to convert this number, integer back into the thing that we are conventionally used to.

So, let us say we have an operation called print, that takes the value that we have defined integer value and converts it into the string that we want. So, print 0 will produce 1st January, print 31 will produce 1st February and so on because, we know that, 30 is 31th of January and 31 will be 1st of February.

So, we can do print, we can write print of the integer and it will produce the string. So, any given point in time you want to convert the string into a integer you can create something which converts the string into integer and operation and you have print which converts the integer back into the string that you want to see which is display.

And we can now allow addition, we can allow though I am not allowed it here but, you could in principle allow addition, you can allow subtraction and definitely you can allow comparison because so far we never did addition and subtraction in our operations, but we did compare, we did compare the dates, so you can compare with less than greater than, or equal to.

(Refer Slide Time: 03:55)



Subtypes of Integer: fractional marks

Integer

Range of values is:
..., -3, -2, -1, 0, 1, 2, 3, ...

Operation	Result type
+, -, ×, ÷	Integer
<, >, =	Boolean

Marks

Range of values is:
0, 1, 2, 3, ..., 100

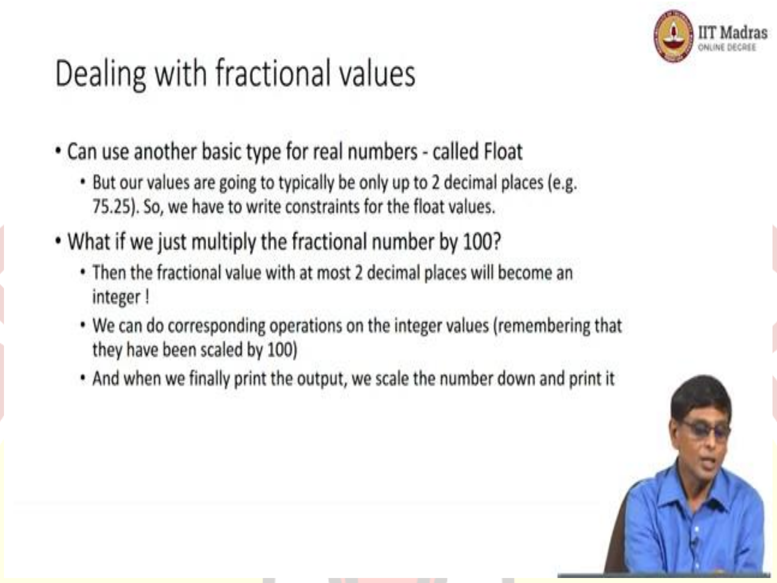
Operation	Result type
+, -	Marks
<, >, =	Boolean

What about fractional marks, e.g. 62.5 ?

A bit more complicated is, fractional marks so, for example if you know teachers use to giving marks like 62.5 and so on, they like to give. But, they do not want to give full marks all the times, they want to give somewhere 62.5 and so on. How do we do 62.5? 62.5 is not an integer, one way is to say that I will turn 62.5 into 62 and store it as 62, or round it off, which means make it 63 and store it as 63, but then some information is lost because, 62.5 is not 62 or 63.

So, we have lost that 0.5 that information of 0.5 and that might mean something later on if you going to award the best rank or something to some student, that might be a problem because, you would, this information of 0.5 then that might create issue. So, you want to store 62.5 as it is but, you cannot do it as an integer then what do you do? Now, one way is to store it as a string, but again the same problem then I cannot add it because, I cannot add marks then I cannot do all that.

(Refer Slide Time: 04:54)



Dealing with fractional values

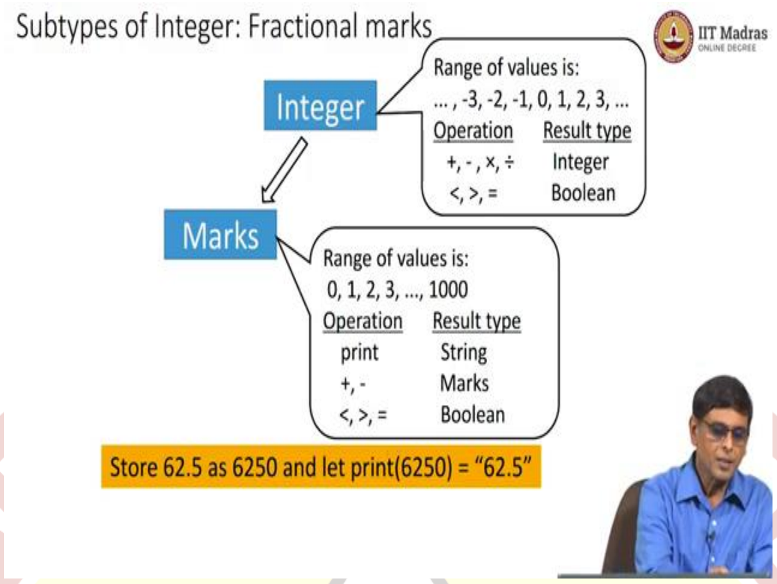
- Can use another basic type for real numbers - called Float
  - But our values are going to typically be only up to 2 decimal places (e.g. 75.25). So, we have to write constraints for the float values.
- What if we just multiply the fractional number by 100?
  - Then the fractional value with at most 2 decimal places will become an integer !
  - We can do corresponding operations on the integer values (remembering that they have been scaled by 100)
  - And when we finally print the output, we scale the number down and print it

So, one way to handle this kind of thing is by using a new data type for real numbers, called float so, typically, in most programming systems you have this data type called float which allows you to write very complex real numbers with any number of decimal places and can be, the number can be really really very large, like can be you know hundreds of billions or several trillions also.

Can be very very large, a very very small for example, it can be very small fraction also, that also is okay, but, we do not need those kinds of extraordinary numbers, we only need to basically have two decimal places. So, I do not want to use float so, if I do not want use float then what is a simpler method? A very simple method might simply be, why do not we just multiply the fractional number by 100, because we know that, they are going to have mostly only two decimal places.

So, we will take the fractional number which has only two decimal places and multiply it by 100. If you multiply a number with two fractional decimal places by 100, it will become an integer. So, now I turn this into an integer and then I store only integers in my data set, and I do all the corresponding operations on integers. And finally when I want to print the output at this time I can convert the integer back into a fractional number and print it as a string. Just like we did for dates, so, I can do that.

(Refer Slide Time: 06:15)


Subtypes of Integer: Fractional marks

Range of values is:
... , -3, -2, -1, 0, 1, 2, 3, ...

Operation	Result type
+, - , x, ÷	Integer
<, >, =	Boolean

Range of values is:
0, 1, 2, 3, ..., 1000

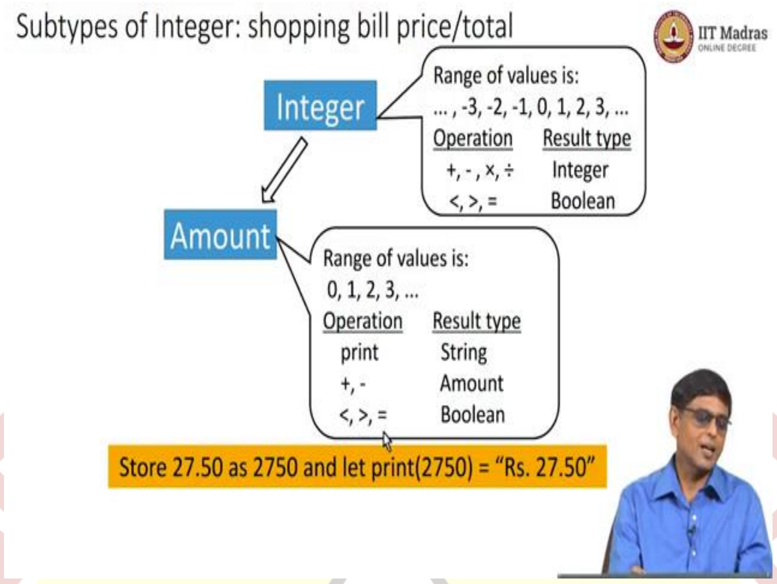
Operation	Result type
print	String
+, -	Marks
<, >, =	Boolean

Store 62.5 as 6250 and let print(6250) = "62.5"

So, here is the description of this data set so, what I am doing now is, I am taking the range of values as it should be 10,000 not 1000, so this is an error here, so because I multiplied by 100 the range of values is now 0, 1, 2, 3 up to 10,000. The operations I can do basically are print so, print basically is, where it will take the number like 62.5 is represented as 6250, 6250 so, print of 6250 will print the string 62.5 so, I can print out the value in the way I want to see it, after I have finished all my operations.

Now, because it is an integer I can do addition I can do subtraction, I can do comparison, I can do all of that.

(Refer Slide Time: 07:03)


Subtypes of Integer: shopping bill price/total

Integer

Range of values is:
… , -3, -2, -1, 0, 1, 2, 3, …

Operation	Result type
+, -, ×, ÷	Integer
<, >, =	Boolean

Amount

Range of values is:
0, 1, 2, 3, …

Operation	Result type
print	String
+, -	Amount
<, >, =	Boolean

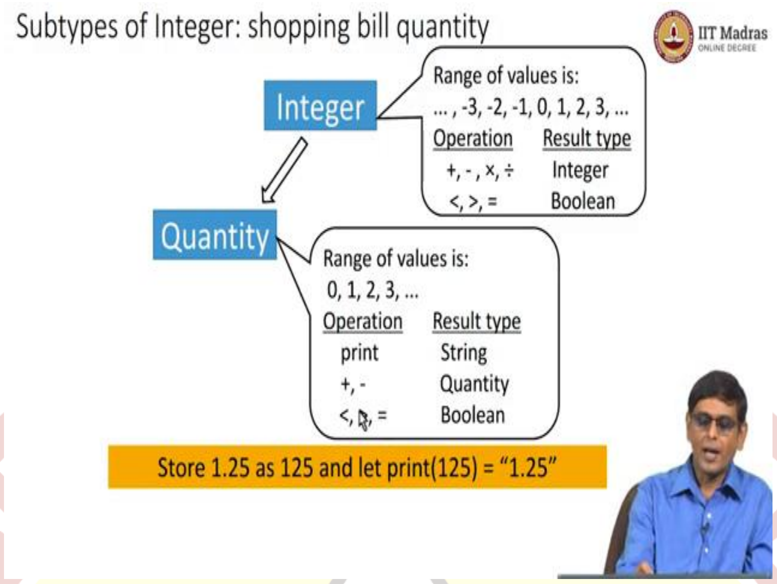
Store 27.50 as 2750 and let print(2750) = "Rs. 27.50"

A similar data type that we saw in the shopping bills, was this amount called, this thing called amount, or price and price and amount basically which we can add all these costs, we could add all these costs and get the total amount of the bill. Those things had decimal places in them so, for example the number could come out as rupees 27 paise 50, 27.50.

So, again 27.50 is fractional number with two decimal places, so easy way to represent that would be to multiply it by 100 so you get 2750 and again the range of values can be 0 1 2 3 up to some maximum number I have not specified what it is, but it could be some maximum number or you could just allow it to range to any positive number and again when you print 2750 you can print it as rupees 27.50 so, that you know it is an amount.

So, the print operation can printed however you want and now because, it is an integer I can do addition, I can do subtraction and I can add to amounts, I can subtract to amounts, I can compare to amount and so on.

(Refer Slide Time: 8:04)



Subtypes of Integer: shopping bill quantity

Range of values is:
… , -3, -2, -1, 0, 1, 2, 3, …

Operation	Result type
+, -, x, ÷	Integer
<, >, =	Boolean

Range of values is:
0, 1, 2, 3, …

Operation	Result type
print	String
+, -	Quantity
<, >, =	Boolean

Store 1.25 as 125 and let print(125) = "1.25"

Same goes for quantity, if you recall the quantity, was either a whole number like 5 or 3, or something like that if you are buying packets cans and so on it was a whole number. Or sometimes it could be fractional quantity like for example if you are buying quarter kilo of tomato, then you would write what 0.25 so, 1.25 you know kilos of something you could buy.

So, again to store this one might multiply it by 100 so, it becomes an integer and store 125 in our data set and so the values can be again any positive integer. And when you print it, you print 125 as 1.25 so, you get the number in the form you want and because, it is now an integer I can do addition, I can do subtraction and I can compare.