

IIT Madras

ONLINE DEGREE

Programming in Python
Professor Sudarshan Iyengar
Department of Computer Science and Engineering
Indian Institute of Technology Ropar
The Obvious Sort

(Refer Slide Time: 0:16)

```
1 l=[112,10,7,18,6,42,8,35]
2 #no to this : l.sort()
3
4 print(l)
```

```
Python 3.8.1 (default, Jan 8 2020, 16:15:59)
Type "copyright", "credits" or "license()" for more information.
IPython 7.19.0 -- An enhanced Interactive Python.
Restarting kernel...

In [1]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
[6, 7, 8, 10, 12, 18, 35, 42]

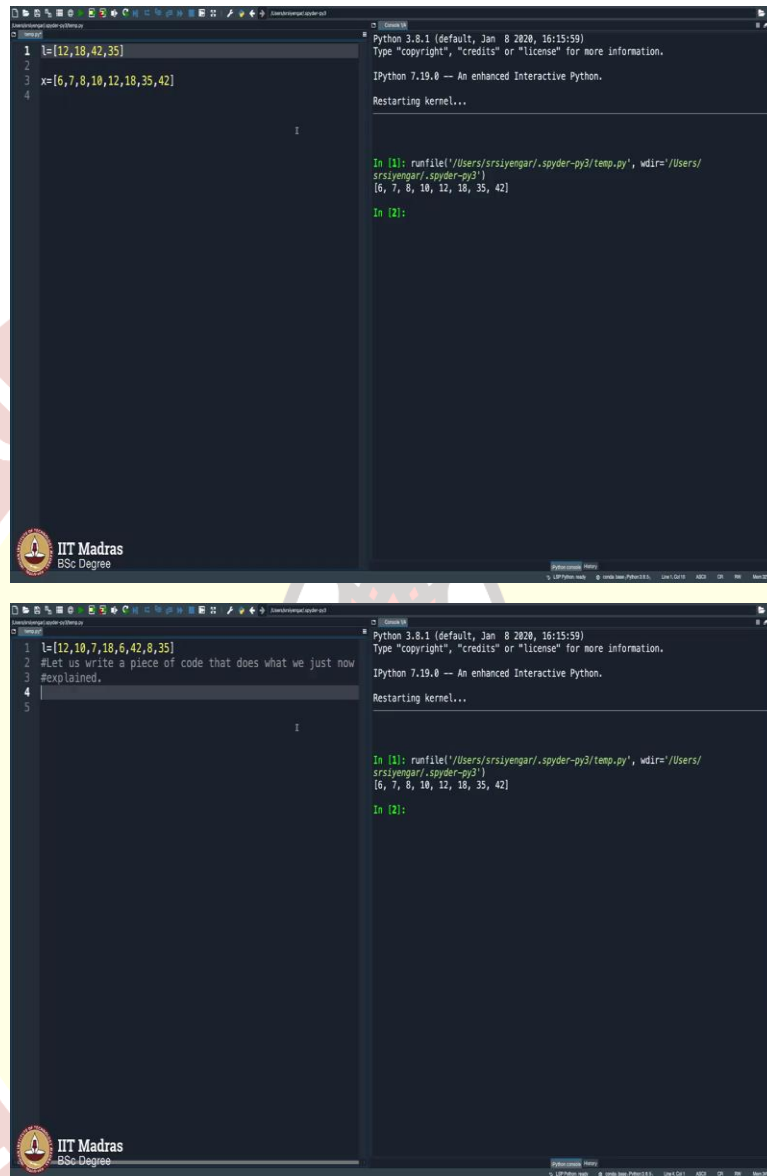
In [2]:
```

Here you see is a list with some eight elements and I do not like this list because it is not sorted and I would like to go ahead and then sort it. One way to do it, of course, is simply say `l.sort()` and then `print(l)`. Perfect, you see that the list is sorted and the entire course is less to do with you using built-in functions and more to do with you understanding how you can write a piece of code all by yourself.

So, what I will do is let us say, no to this, let us say no to built-in functions and let us do it the hard way. I am sure many of you are already realizing that I am being a big fan of writing long codes while we can actually do it with a very, very short snippet of code. So, I am doing it purposefully just to ensure that we get our hands wet and do it from first principles.

So, what is the fun in simply saying that everything exists? You see the whole course, the whole of your BSc in data sciences and programming will be a waste because you need not have to write a single piece of code because everything that you will write probably is already there. For practice sake, I think we should try to sort of reinvent the wheels. So, let us reinvent this wheel of sort, let us try to emulate sort, let us try to redo, how one can sort these numbers.

(Refer Slide Time: 1:52)



The image shows two screenshots of a Jupyter Notebook interface. The top screenshot shows a code cell with the following code:

```
1 l=[12,18,42,35]
2
3 x=[6,7,8,10,12,18,35,42]
4
```

The bottom screenshot shows the same code cell, but with additional comments added:

```
1 l=[12,18,42,35]
2 #Let us write a piece of code that does what we just now
3 #explained.
4
5
```

Both screenshots show the Jupyter Notebook interface with a dark theme. The top screenshot shows the kernel restarting and the bottom screenshot shows the kernel restarting again. The bottom screenshot also shows the output of the code execution, which is a list of numbers: [6, 7, 8, 10, 12, 18, 35, 42].

So, let me remove all the other things except for the list l, how would I sort this the obvious way? In computer science the moment you say sort there are some dozens of ways in which you can sort a bunch of numbers. Let us do it the common sensical way.

Now, what is that common sensical way? Let me try to reinvent my own sorting technique. So, I will call a new array x and what I will do is, for a moment I will not do programming, I will use this place as a slate. So, I will try to see through l and try to see what is the least element here. In fact, but the moment you see it, you see 6 is the least element but machine wise how will you find out what is the least element? What if this list were to be a very big list and that just by

staring at it you could not figure out that 6 is indeed the least element if there were to be some thousand elements here, how do you go about it? You will say.

You will see the first element and you call this the least element and go ahead and you see, I now have 10, I will call 10 as the least element, I will not call 12 as the least element. So I will say initially I will say least equals 12 but then no, least was 10 and then I see 7 here, I see 7 here and say, 10 is not the least, 7 is the least then I see 18 here, I say 7 is still the least so far as I am reading through this little line sequentially from left to right, 18 is not less so 7 is the least seen so far, but then you have 6 so then you remove 7 and say 6 is the least seen so far and then 42, still 6 is the least seen so far 8, 35, 6 is the least seen so far.

The moment you do that, you put 6 here and then remove 6 from here and then repeat this process. I mean what is, this appears very complicated when we try to do it this way but all I am saying is pick the least element, put that to x and remove that least element there. As you can see you go through it, the least was 12 and then you saw 10 and the least became 10 initially it was 12 and then it becomes 10, I am sorry 10 and then it becomes 7 and 18 is not less, 42 is not less than 7 it is not less than 7, 35 is not less than 7, so 7 is the least so what I will do is I will remove 7 from here and plug that in here, you see I am getting a sorted list slowly in the making.

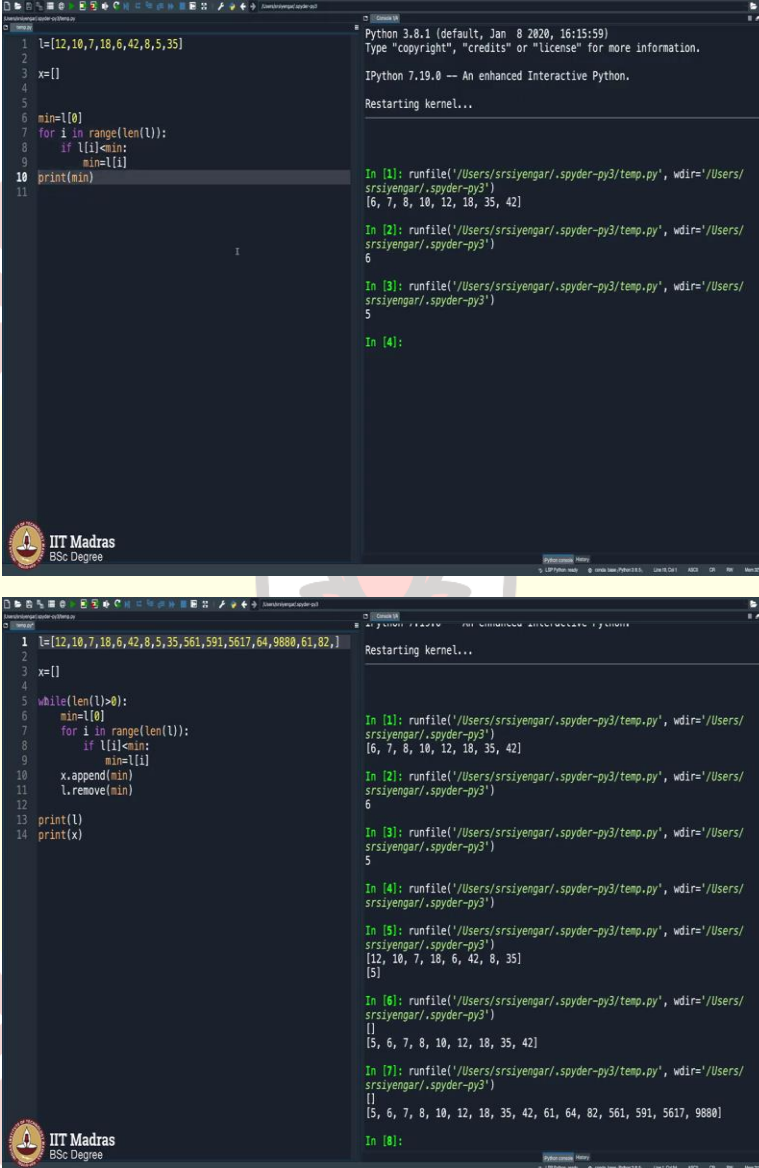
So, then once again, you can just see it and then simply plug in 8 here but you must understand the process with which you pick that; you start from the left, you call the first one as the least and then keep seeing something lesser than what you have seen so far as the least and then change your least value as you get better and better terms, that way you will get the minimum most here.

So, you will get 8 here you remove 8 and plug that 8 in here and then the next one as you go ahead like this list is 12, no 10, no 18 no it is not least, 22 no, 35 no, 10 is the least so you plug in 10 here and you remove 10 here, you see the list is slowly getting sorted, you are removing entries of 1 and plugging them into x and surprisingly for you, they are getting sorted similarly 12 will come here, 18 will come here, let me just remove this line, you understand what this means.

So, we will write a code using this very idea 18, then 35 and then 42 as this happens the list gets deleted, I skipped a lot of steps here, I suppose you understood. So, let me undo everything and go back to the beginning. So, all that I typed I am undoing it so that just the list is there. Now, I

am going to write a program here. Let us write a piece of code that does what we just now explained. So, let us go ahead and do that now.

(Refer Slide Time: 6:38)



The image displays two screenshots of a Jupyter Notebook interface, likely from a video recording. The background features a large, semi-transparent watermark of the IIT Madras logo and the text "INDIAN INSTITUTE OF TECHNOLOGY MADRAS".

Top Screenshot:

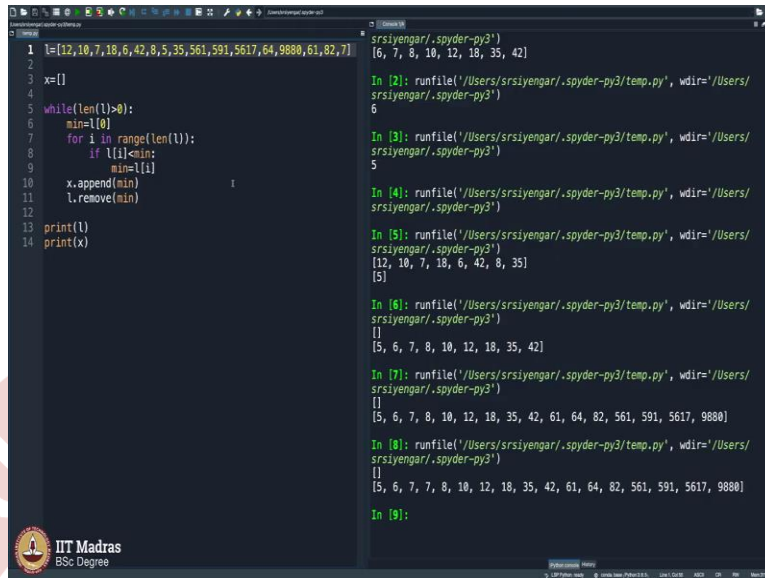
- Code Editor:** Contains a Python script to find the minimum element in a list `l`.

```
1 l=[12,10,7,18,6,42,8,5,35]
2
3 x=[]
4
5 min=l[0]
6 for i in range(len(l)):
7     if l[i]<min:
8         min=l[i]
9
10 print(min)
```
- Output Console:** Shows the execution of the code, resulting in the output `6`.

Bottom Screenshot:

- Code Editor:** Contains a Python script to find the minimum element in a list `l` and then remove it, appending it to a new list `x`.

```
1 l=[12,10,7,18,6,42,8,5,35,561,591,5617,64,9880,61,82,]
2
3 x=[]
4
5 while(len(l)>0):
6     min=l[0]
7     for i in range(len(l)):
8         if l[i]<min:
9             min=l[i]
10            x.append(min)
11            l.remove(min)
12
13 print(l)
14 print(x)
```
- Output Console:** Shows the execution of the code, resulting in the output `[5, 6, 7, 8, 10, 12, 18, 35, 42]` for `l` and `[6, 7, 8, 10, 12, 18, 35, 42]` for `x`.



```
1 l=[12,10,7,18,6,42,8,5,35,561,591,5617,64,9880,61,82,7]
2
3 x=[]
4
5 while(len(l)>0):
6     min=l[0]
7     for i in range(len(l)):
8         if l[i]<min:
9             min=l[i]
10    x.append(min)
11    l.remove(min)
12
13 print(l)
14 print(x)
```

```
srsiyengar/.spyder-py3'
[6, 7, 8, 10, 12, 18, 35, 42]

In [2]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
6

In [3]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
5

In [4]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
[12, 10, 7, 18, 6, 42, 8, 35]
[5]

In [5]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
[6, 7, 8, 10, 12, 18, 35, 42]
[]

In [6]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
[5, 6, 7, 8, 10, 12, 18, 35, 42]
[]

In [7]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
[5, 6, 7, 8, 10, 12, 18, 35, 42, 61, 64, 82, 561, 591, 5617, 9880]
[]

In [8]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
[5, 6, 7, 7, 8, 10, 12, 18, 35, 42, 61, 64, 82, 561, 591, 5617, 9880]
[]

In [9]:
```

So, let us try writing a piece of code, so what did you have there? You had x being empty, initialized and you should find the minimum most element in l, how do you do that? Initially you declare least or I will call it minimum it is easy on a mind or even least any variable will do, minimum to be the first element in l.

And then I will go through the entire list l for i in range len of l will go through the entire list and as it goes to the entire list I say if l of i is less than minimum, just in case l of i is less than minimum, only then you call minimum as that l of i, obviously here as you enter the for loop the first one will be l of 0 is less than minimum, obviously not because minimum is already l of 0, 12 is not less than 12 so this will not happen, but then 10 is less than 12 so minimum will be equal to 10 and then 7 is less than 10 so minimum will be equal to 7, so on and so forth and finally 6 will get stuck here as minimum.

And finally, if I say print min, let us see what happens at this level let me just stop coding and execute this and see what I expect here, I expect what to be, what to get printed here? I expect 6 to get printed here. Let us see if that is happening, yeah that is indeed happening here as you can see just now executed I get 6 here. I am just increasing the font size so that it is clear to you people, so I get 6 here. Very good so assuming I plugged in some 5 here and then execute it I should get 5, yeah I am getting five, this is how you code ,line by line trying to ensure that what you are doing is right.

So, but this is just giving me the minimum most number. I want the entire list sorted how do I do that? So, what I do is I do not print minimum, I plug in minimum to my new list x and then I should remove that element from l, you see we did some warming up of lists in the very first lesson in this week, we did talk about how you can remove an element from a list.

You simply say l remove min and it will remove the first occurrence of that number. Do not break your head, you will see what is happening now so let us print l and let us also print x here. See what happens. So, l becomes 12, 10, 7, 18, 6, 6 is retained, that is because we are included 5, so 5 gets removed, 5 did get removed and 5 has come here. Very good news we are through, but then we need to keep doing this.

So, this entire stuff should be inside a small loop, you see, what is that loop? Figure out, so I will say while l is not empty, l is not empty, of course we are not going to write it like that, we are going to write it symbolically, we will say while length of l is greater than 0 which means length of l should at least be 1 which means l should have at least one element then I go ahead and do this as I know slowly the length of l is decreasing because of this line here where I am removing one element at a time.

And finally, out of outside this while loop I am going to display l and x, I am hoping now that this will take care of the fact that my list x will be the sorted version of l. Why? Because, I am going through this entire thing again and again and again trying to append new elements into x and trying to remove the elements and then moving them to x, so l is becoming smaller and smaller, x is getting bigger and bigger with the values of l entering into x the sorted way.

Hip hip hooray, as usual 5, 6, 7, 8, 10, 12, 18, 35, 42 and we have done it. So, let me try including more numbers here 561, 591, 5617 comma 64 comma 9880, 61, 82, so let me just execute this. Perfect, it is getting sorted. Now, let me try to include a repetition so 7 is here, 7 is here as well, let us see if it sorts, what is your first guess, do you think this will throw an error or will it sort?

Wow, it is sorting, it is showing 7 and 7 twice, sometimes you get free gifts like this, you never fix the code for repetitions but it is taking care of repetitions. Why? It is actually very easy to observe, in fact, whenever you remove an element the first occurrence of that element gets removed.

In the next iteration inside the while loop that very number gets caught once again and then it gets removed. Too many technicalities, again, the entire class I am sure of all you people, a few thousands of you people some of you find this very silly, very straightforward, we have done this already in our school days, why are you repeating it while many, many, many of you this comes as a surprise.

What is happening? This looks so complicated, you see editor with some tab going on the right like this, like a slide, what are you doing here? This is just like a mechanic shop, do not expect it to look like a word editor or your google docs where it all looks very neat and you start typing your letter, it is not like that, it is a hardcore mechanic workshop where you grease your hands, your shirt and pant gets sort of dirty and that is how you write a piece of code.

So, this is how it is, you should get used to it, while you type you will get errors, while you type you will get confused, you must go very slowly. So, before you type the code here the way I explained, I sort of had the experience of how to write the code but then you may not have it. What you should do is you should write it on a sheet of paper, try to figure out what the logic is and then come here and type it.

I hope you people enjoyed the first obvious sorting program which you have already learnt in computational thinking course but I did not use any word here, probably you learned insertion sort there I just did the obvious sort here, I am not even going to give a name for this. In fact you will be seeing more and more and more sorting techniques during your entire program. So, let us go to the next topic. Thank you.