

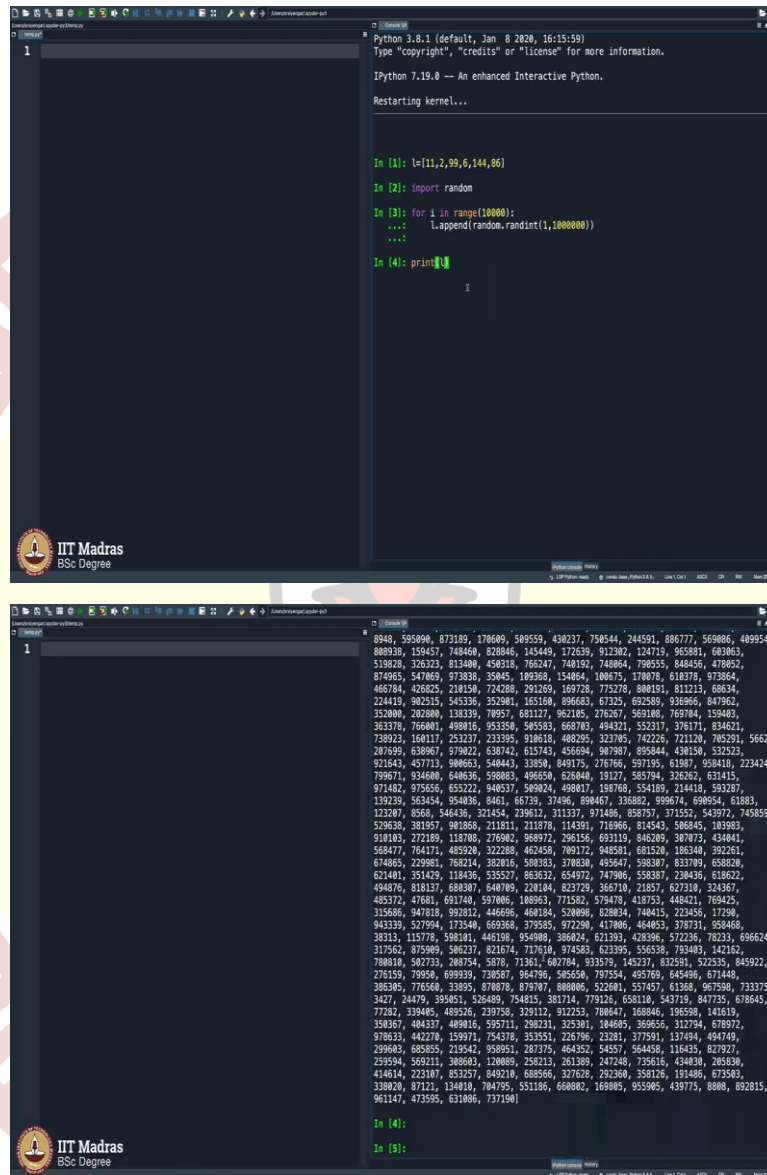


IIT Madras

ONLINE DEGREE

Programming in Python
Professor Sudarshan Iyengar
Department of Computer Science and Engineering
Indian Institute of Technology, Ropar
Naive Search in a List

(Refer Slide Time: 0:16)



The image shows two screenshots of a Jupyter Notebook interface. The top screenshot shows the initial code cells: a list `l = [11, 2, 99, 6, 144, 86]`, importing `random`, and a loop that appends random integers to the list `l` for 10,000 iterations. The bottom screenshot shows the output of the code, displaying a large list of 10,000 random integers. The IIT Madras logo is visible in the background of both screenshots.

```
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]: l=[11,2,99,6,144,86]

In [2]: import random

In [3]: for i in range(10000):
....:     l.append(random.randint(1,1000000))
....:

In [4]: print(l)
```

8948, 505000, 873189, 178609, 589559, 438237, 758544, 244591, 886777, 569086, 489954, 889338, 159457, 748460, 828846, 145449, 172639, 912382, 124719, 965881, 683863, 519828, 326323, 813480, 458318, 766247, 748192, 748864, 798555, 848456, 478852, 874965, 547069, 973839, 35045, 189368, 154864, 180675, 178078, 618278, 973884, 465784, 426525, 218150, 724288, 292359, 169725, 713278, 888191, 812125, 68824, 224419, 902515, 545336, 352901, 165168, 896683, 67325, 692389, 936966, 847962, 352880, 202880, 138339, 78957, 681127, 962185, 276267, 569188, 769784, 159483, 363378, 766881, 498816, 953358, 585583, 668783, 494321, 552317, 376171, 834521, 738933, 158117, 252327, 233355, 918618, 488285, 923785, 742226, 721120, 763591, 5662, 287699, 638967, 978822, 638742, 615743, 456694, 887887, 895844, 438158, 532523, 921643, 457713, 988663, 548443, 33858, 849175, 276766, 597195, 61887, 958418, 223424, 799671, 934680, 648636, 598883, 496658, 626848, 19127, 585794, 326262, 631415, 971482, 975656, 655222, 948537, 589924, 498817, 197468, 554189, 214418, 593287, 139239, 563454, 954885, 8461, 66739, 37486, 894407, 336882, 958674, 698954, 61883, 132887, 8568, 546436, 321454, 239612, 311337, 971488, 888757, 371552, 543972, 745859, 529638, 381957, 981868, 211811, 211878, 114391, 716966, 814543, 586845, 183983, 918183, 272189, 118788, 276982, 988972, 296156, 683119, 846289, 387873, 434841, 588477, 764171, 485928, 222288, 482458, 789172, 949381, 681528, 186340, 292261, 674865, 229861, 768214, 382816, 588383, 378838, 495647, 598387, 833789, 658828, 621481, 351429, 118436, 535527, 863632, 654972, 747986, 558387, 238436, 618622, 494876, 818137, 688387, 648789, 228184, 823729, 366710, 21857, 627318, 324367, 485372, 47681, 681749, 587886, 188863, 771582, 579478, 418753, 448421, 769425, 315686, 947818, 952812, 446686, 468184, 528868, 428824, 748415, 223456, 17288, 943339, 527994, 173548, 669368, 379585, 972288, 417886, 464853, 378731, 958468, 38313, 115778, 598181, 446198, 954888, 386824, 621393, 428396, 572236, 78233, 696624, 317562, 875989, 586237, 821874, 717618, 974583, 623395, 535538, 753483, 142162, 788818, 582723, 280754, 5878, 73361, 68784, 333579, 145237, 825591, 522325, 845922, 276159, 79958, 699939, 738587, 964796, 505658, 797534, 495769, 645496, 671448, 386385, 765688, 33895, 878878, 879787, 888886, 522681, 557457, 61368, 967598, 733375, 3427, 24479, 395851, 526489, 754815, 381714, 779126, 638118, 543719, 847735, 678645, 7728, 338485, 489326, 289758, 329112, 912553, 788647, 168846, 198588, 141519, 358367, 404337, 488816, 595711, 288231, 325381, 184685, 389656, 312794, 678972, 978833, 442278, 159971, 754378, 353551, 226796, 23281, 377591, 137494, 494749, 299683, 685855, 219542, 958951, 287375, 464352, 54557, 564458, 116435, 827927, 229594, 593211, 386882, 128889, 258213, 261389, 247248, 735616, 454838, 295386, 414614, 223187, 852527, 549218, 688566, 327628, 392288, 358126, 121466, 675863, 338828, 87121, 134818, 784795, 951186, 668882, 169885, 855985, 439773, 8888, 892815, 961147, 473595, 631886, 737198

So, we are going to go ahead and try to write a small code to search for an element in a big list. Firstly, let me tell you how to create a big list. So, how do we go about it? You can always type in a big list like this, 99, 6, 144, 86 and so on that is the biggest list I could type in less than 10 seconds, but if you want to type a bigger list than this, there is a way to do it. So, how do you do it?

Say import random and then let us say for i in range 10000, you append to the existing list let us say whatever you have created here or maybe you can remove the numbers here; I append

random.randint let us say 1 comma so much. How much is this? 1000000. From 1000000 numbers, 1 to 1000000 I am picking a few numbers, close the bracket and there you are. As I have been telling you can type the code here itself on the terminal.

But then, when you type it here, you can execute it repeatedly, once typed here you may not be able to execute them in sequence. Let us see what is in l. l has so many numbers. So, you cannot scroll it, it, they are plenty many. So, what you do is, it can be a little buggy a times because you are generating so many numbers scrolling up scrolling down can be a problem.

(Refer Slide Time: 1:56)

```
1 import random
2 l=[]
3 for i in range(1000000):
4     l.append(random.randint(1,1000000))
5
```

```
In [4]:
In [5]: import random
In [6]: l=[]
In [7]: for i in range(100):
...:     l.append(random.randint(1,1000000))
...:
In [8]: print(l)
[531470, 444876, 152045, 486009, 392080, 969188, 648557, 790852, 171208, 274372,
419116, 105229, 573908, 35589, 668838, 211959, 814551, 130736, 993536, 683194,
583246, 794077, 763730, 202772, 712346, 848445, 599836, 359447, 146696, 362375,
458420, 945868, 590160, 281519, 579668, 862466, 445283, 9855, 896293, 136539, 737683,
107327, 648453, 241362, 87839, 633565, 576627, 613344, 651415, 867294, 321846,
984459, 906337, 39104, 689267, 733869, 378955, 195251, 627410, 684176, 205254,
811824, 627132, 648371, 279037, 167813, 165514, 4917, 177858, 796939, 818178, 487456,
661542, 255977, 643617, 637575, 489999, 358425, 906154, 884945, 527553, 195947,
647444, 170187, 442328, 842187, 170123, 925615, 744260, 73076, 488532, 356824,
123216, 158182, 601652, 877979, 50931, 70120, 258616, 379214]
In [9]: runfile('D:/Users/srsiyengar/.spyder-py3/temp.py', wdir='D:/Users/
srsiyengar/.spyder-py3')
In [10]: print(l)
```

```
3456995, 6139980, 1342711, 9969617, 2179951, 3540751, 6481277, 3222746, 1453167,
6127099, 6153581, 3308320, 6201621, 7282675, 3388483, 1182118, 5501975, 6595783,
4887636, 9798125, 2689295, 2748820, 3198550, 4742899, 5762166, 3344271, 9124969,
9728910, 635485, 3907835, 6791621, 6902476, 6557789, 4181115, 1562940, 4360852,
9080443, 3267111, 1310398, 2073996, 4380824, 4331562, 1423469, 3546321, 78886,
6911953, 6746990, 8961658, 9297439, 6949283, 4984886, 1127562, 758519, 2518761,
3224796, 4356840, 2206296, 7874972, 8837067, 9017226, 265179, 3198659, 4586749,
7209449, 4670014, 7981049, 8552407, 72625, 5334965, 8326482, 1529210, 2507331,
5855452, 5897219, 4291308, 6272937, 4926070, 2769095, 9866446, 5783210, 9176586,
7337458, 55108, 2745021, 841724, 4533772, 8791025, 8798537, 4715231, 4181009,
9713346, 3292202, 2385526, 696576, 184121, 8475814, 3912845, 6174489, 729658,
7349951, 9578212, 5947311, 3234322, 8951407, 8980122, 6467096, 6861430, 5969934,
820769, 8013255, 2815996, 1187180, 3446239, 8248447, 8183784, 1639803, 2816246,
9548929, 8289754, 6852065, 901610, 555748, 3327283, 8813251, 1548819, 686296,
1285065, 5896030, 9345125, 3193566, 2607305, 5795666, 2110617, 3839136, 7984578,
1704053, 9835172, 2179927, 6768597, 9543412, 2383179, 8534632, 87273, 8228070,
2786886, 685951, 3274106, 5858021, 1186146, 682524, 5831098, 7075688, 8968074,
8461349, 7363086, 2872714, 1290712, 8302981, 8412824, 7208222, 8455410, 531089,
9840569, 8104454, 8227218, 8611844, 2064476, 7018475, 156819, 291968, 95542, 5238846,
3604089, 3717617, 3205888, 524728, 2781858, 6399810, 8949441, 9987463, 7126889,
3383977, 1267931, 3893537, 3707992, 1347496, 378336, 6133543, 8686561, 7533347,
4808829, 3413725, 5061235, 3274348, 4281710, 1746468, 4551472, 4283330, 3040074,
1495499, 6562075, 6769498, 2225781, 8883390, 2981261, 2213712, 4239891, 5724483,
8008925, 706495, 6233082, 7838067, 1338837, 6029525, 393622, 9835632, 6586444,
3985117, 6478882, 1248461, 9426491, 1415708, 3858392, 9565181, 9567377, 8526234,
3765840, 9726823, 3341025, 2391617, 9486168, 7549851, 9651493, 1753217, 4958864,
707921, 3812722, 5698913, 841645, 226089, 4849928, 723556, 5289589, 763808, 7514400,
6100291, 1732110, 9685191, 9779156, 324222, 1700799, 6588273, 4747140, 8783584,
5611294, 9939837, 3768780, 7666849, 7782755, 8556126, 2283551, 2253181, 5710716,
1244558, 4259255, 583925, 6956189, 2164624, 1444997, 8092444, 9264183, 8299937,
485530, 429772, 9449089, 7812193, 1865956, 9746820, 3407600, 4468480, 3073139,
2424888, 9131407, 7325516, 617052, 4682413, 3281912, 1627898, 6315997, 3632895,
3232379, 7606265, 3588312, 8800225, 7785200, 986466, 2998651, 2083964, 9311779,
536663, 6194398, 5350882, 991591, 4338674, 7672456, 9977028, 3197481, 4518558,
7623163, 4097622, 8719264, 5437854, 3679930, 3182778, 508362, 7547713, 326793,
6736197, 5765236, 3540471, 9399184, 9122807, 6971272, 2178135, 7807134, 3990824,
6548160, 1815469, 6978349, 1890467, 7402090, 8593483, 6148510, 3881710, 3892578,
6788482, 8585650, 5595140, 3950839, 2485109, 318718, 1147288, 1397291, 2604708,
394793, 6028369, 5010475, 6118346, 4352774, 1722425, 1916126, 9396261, 881347,
769393, 5914418, 9401172, 2171030, 5273916, 2857087, 5835235, 3285944, 7691012]
```

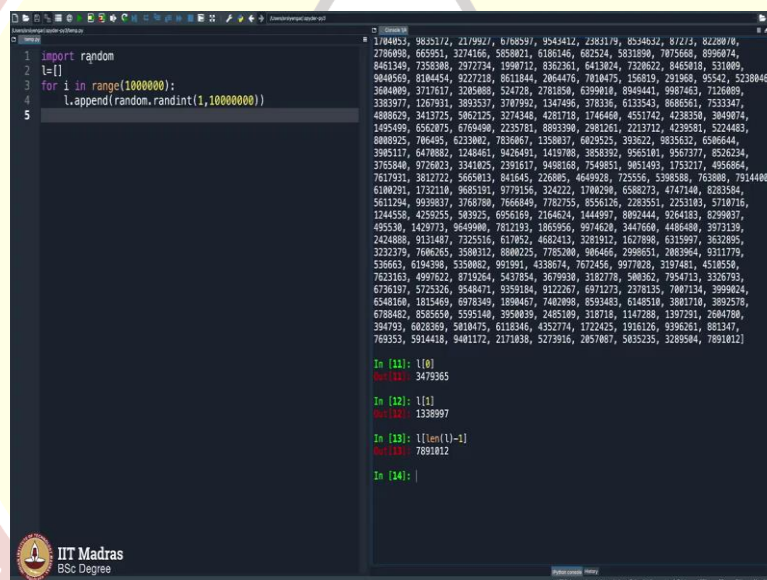
So, I repeat once again. So, what do I have import random and then I say I create let us say a list, I am sorry bear with me. So, I create a list l, what exactly did I do? I created a list l and then I populated the list with some 10000 numbers or maybe let us say 100 numbers, just that

it becomes easy, does not scroll much and it is easy to see, easy on the eyes rather. Say enter and you get 100 elements here from 1 to 1000000.

So far so good. So, then this is a big list, but I want a very big list. So, what do I do? I start with `l` equals so much and then firstly of course, import random, do not forget we are trying to create a big list. For `i` in range a big list I want let us say 1000000 numbers in it. This is 1000000 numbers. I append random randint in the range 10000000. From 1 to 10000000, I am picking 1000000 numbers.

So, now the list `l` will be ready for execute this, I am executing it. I am expecting it will take some time, but it was really fast, very good. So, if I say print of `l`, it will give me once executed, as I said, the variables will be there; you can execute and now you see `l` here.

(Refer Slide Time: 3:47)



```
1 import random
2 l=[]
3 for i in range(1000000):
4     l.append(random.randint(1,10000000))
5
```

Out[10]: 1704053, 9845172, 2179927, 5708397, 9543412, 2481179, 8534632, 872713, 8220070, 2706090, 665951, 3274166, 5850021, 6186146, 602524, 5831890, 7075668, 8996074, 8461349, 7358306, 2972734, 1990712, 8362361, 6413824, 7206622, 8465018, 531009, 9040569, 8104454, 9227218, 8611844, 2064476, 7010475, 156819, 291968, 95542, 5230046, 3040009, 3717627, 2208080, 524728, 2781850, 6299010, 8904041, 0987463, 7126089, 3383977, 1267931, 3093537, 3707992, 1347496, 378336, 6133543, 8686561, 7533347, 4080629, 3413725, 5062125, 3274348, 4281718, 1746460, 4551742, 4238350, 3049074, 1495499, 6562075, 6769498, 2235781, 8893390, 2981261, 2213712, 4239581, 5224483, 0000025, 706495, 8233002, 7830607, 1358037, 0229251, 935622, 9835332, 6506644, 3005117, 6470802, 1248461, 9426491, 1419786, 3858392, 9565101, 9567377, 8526234, 3765840, 9726023, 3341825, 2391617, 9498168, 7549851, 9051493, 1733217, 4956864, 7617931, 3812722, 5665813, 041645, 226085, 4649928, 725556, 5398588, 763080, 7914400, 6100291, 1732110, 9085191, 9779156, 324222, 1700290, 6508273, 0747140, 828356, 5511294, 9930327, 3768700, 7666049, 7702725, 8550126, 2203951, 2253105, 5710716, 1244558, 4259255, 503925, 6956189, 2164624, 1444997, 8002444, 9264183, 8799037, 495530, 1429773, 9649980, 7812193, 1065956, 9974620, 3447660, 4486408, 397139, 2424888, 9131487, 7325516, 617052, 4682413, 3281912, 1627898, 6315997, 3632895, 3232379, 7607465, 3690312, 6000275, 7785200, 086466, 2960691, 2002964, 9211779, 536663, 6194398, 5350882, 991091, 4338674, 7672456, 9977020, 3197481, 4510550, 7623163, 4997622, 8719264, 5437854, 3679930, 3182778, 500362, 7954713, 3326793, 6736197, 5725326, 9548471, 9359184, 9122267, 6912729, 2378135, 7007134, 3999024, 6549100, 1815469, 6978349, 1894467, 7402908, 6593463, 6145810, 3001770, 3692578, 6788482, 8538550, 5593140, 3590059, 2485100, 510718, 1147260, 1397091, 2604700, 394793, 6028369, 5010475, 6118346, 4352774, 1722425, 1916126, 9396261, 801347, 769353, 5914418, 9401172, 2171038, 5273916, 2057087, 5035235, 328504, 7891012

In [11]: l[0]
Out[11]: 3479385

In [12]: l[1]
Out[12]: 1336997

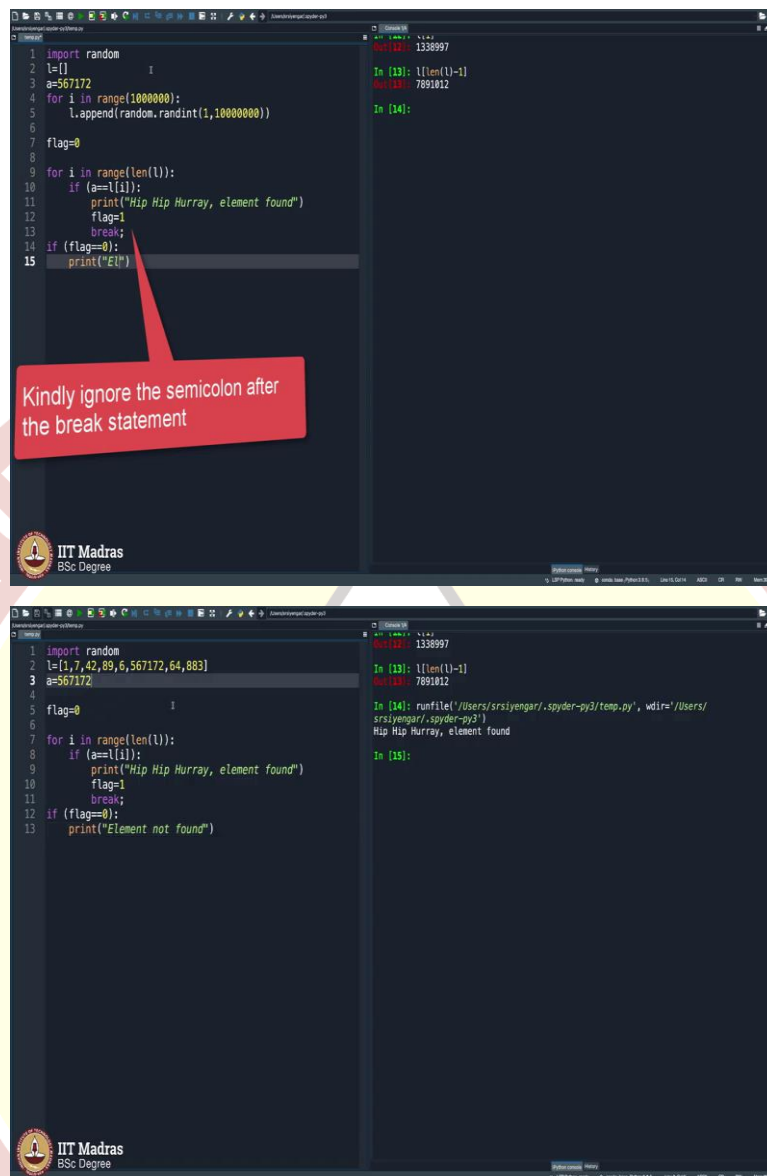
In [13]: l[len(l)-1]
Out[13]: 7891012

In [14]: |

As you know `l` of 0 will be the first element, `l` of 1 will be the first, second element. How do you see the last element? `len` of `l` minus 1. So, it does get stuck like this. This is very common whenever you handle a big list and your memory is full and even your Python compiler requires some memory, your internal memory also called the RAM. So, this is very usual. You should get used to it. I mean when you handle a huge data.

So, the last number is 7891012 and that is precisely what you see here. I would like to check if there is a number let us say I will type that here, please do not mind the lag here because I am generating big numbers, it can cry like this a times.

(Refer Slide Time: 3:41)



```
1 import random
2 l=[]
3 a=567172
4 for i in range(1000000):
5     l.append(random.randint(1,1000000))
6
7 flag=0
8
9 for i in range(len(l)):
10     if (a==l[i]):
11         print("Hip Hip Hurray, element found")
12         flag=1
13         break;
14 if (flag==0):
15     print("Element not found")
```

Kindly ignore the semicolon after the break statement

```
1 import random
2 l=[1,7,42,89,6,567172,64,883]
3 a=567172
4
5 flag=0
6
7 for i in range(len(l)):
8     if (a==l[i]):
9         print("Hip Hip Hurray, element found")
10         flag=1
11         break;
12 if (flag==0):
13     print("Element not found")
```

Output:

```
In [12]: 1338997
Out[12]: 1338997
In [13]: l[len(l)-1]
Out[13]: 7891012
In [14]:
Out[14]:
In [14]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
Hip Hip Hurray, element found
In [19]:
```

So, what is happening here is I have scrolled up and scrolled down and it is happening slowly whatever I did. So, what I do is I say element a equals 567172 and I would like to check whether this is there in l or not. So, how do I go about it? For i in range length of l so which means it runs, i runs from 0 to upto the length of l minus 1, the whole of the array. If a is equal to l of i print 'Hip Hip Hurray, element found'. Otherwise it should print 'element not found'.

So, as I have been saying what you will do here is you will keep a flag variable that is the most complicated thing in programming. So, as we have discussed already I will not go in detail about this, you already know what this does, it will be sort of self-explanatory. So, once

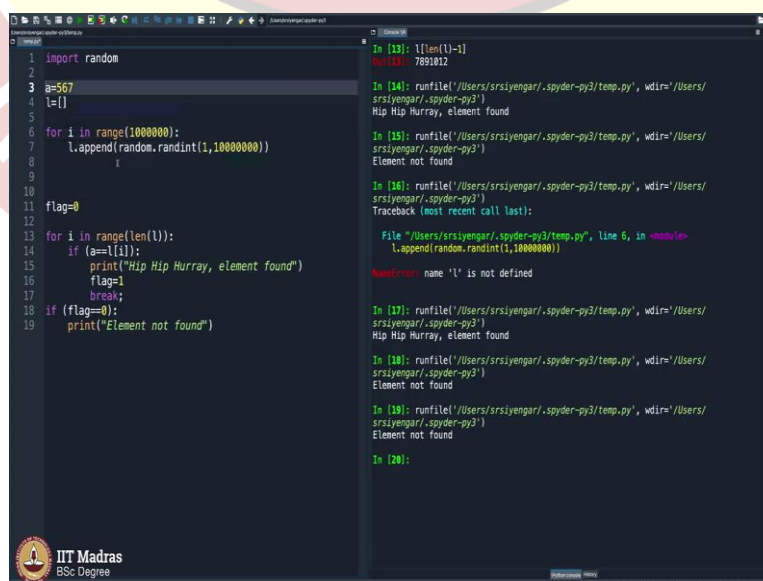
you find the element, you break from the for loop. And then if flag is equal to 0, then you say 'element not found'.

So, what is happening here? I think a good way to illustrate this is to actually not use the for loop, let us say I will comment this, please get used to commenting. So, I will type in a small array here, for small list here; 1, 7, 42, 89, 6, 567172, 64, 883. Your element is definitely here. So, what happens now, when you say a equals 567172, I will just remove this for loop just so that, so I am not trying to confuse you people, all I am trying to do is I am trying to keep the lists smaller.

Once you make it bigger, it does not matter, the program still runs. But it will be easy on your minds if I keep the lists smaller, I realised it as I was programming so I am doing it right now. Let it not confuse you, the program is pretty simple and straight forward. All I am doing is I initialise flag with 0 and I say I declare a variable a to be the number and check whether this number is present in the list or not.

So, I run i through the entire l and in case this value a is equal to any value in l, I print 'Hip Hip Hurray' I create flag equals 1 and then I break which means flag is no more 0, which means this will not get executed in case you have seen the element. This will only get executed if flag never became 1, so if flag equals 0, then it says print 'element not found'. This programs might appear little complicated to you to begin with. But with time you will get used to it. I just executed, it says 'Hip Hip Hurray, element found'.

(Refer Slide Time: 8:30)



```
1 import random
2
3 a=567
4 l=[]
5
6 for i in range(1000000):
7     l.append(random.randint(1,1000000))
8     i
9
10
11 flag=0
12
13 for i in range(len(l)):
14     if (a==l[i]):
15         print("Hip Hip Hurray, element found")
16         flag=1
17         break;
18 if (flag==0):
19     print("Element not found")
```

In [13]: l[len(l)-1]
Out[13]: 7891012

In [14]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
Hip Hip Hurray, element found

In [15]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
Element not found

In [16]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
Traceback (most recent call last):
File "/Users/srsiyengar/.spyder-py3/temp.py", line 6, in <module>
l.append(random.randint(1,1000000))
NameError: name 'l' is not defined

In [17]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
Hip Hip Hurray, element found

In [18]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
Element not found

In [19]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
Element not found

In [20]:

IIT Madras
BSc Degree

Just in case, I were to type in some other element, let us say 567 is not here in this list. It should say element not found. It indeed says element not found. Now what we will do is I will include the for loop, you probably are wondering, what is the big deal in seeing whether 567 is there in this list or not. It is obviously not there. It is visible for my naked eye, why do you write a program for it? So, that is when the complexity question pictures in. What if this list is not as simple as it appears here?

Let me remove it. Let a be 567 only and I am appending several several how much is this as I was telling you some 10000000, from 1 to 10000000 I am appending a random number a million times, a random number as big as let us say 10000000, 1 to 10000000. A million times I append. Just stare at this for loop for a minute. And in that I I am going to search for 567 if it is there or not. I am executing it.

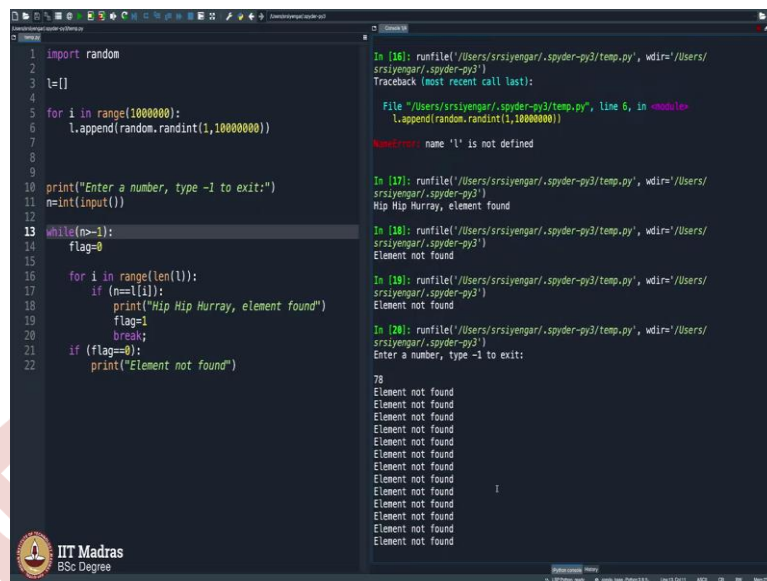
Boom, l is not defined that is because I made an error here of not initializing l. So, simply I am appending to the list l, the list l should exist, unless it exists I cannot append it as simple as that. This kind of mistakes even you will be doing, we are not going to edit these mistakes because these mistakes do happen, you must feel confident that everyone makes mistakes and such mistakes when you make, you know how to fix it.

So, I run this and I get 'Hip Hip Hurray, element found' which means a equals 567 this particular list generated. This particular for loop random number when I was adding, 567 got added. So, let me execute it once more. Now element not found. 567 did not get included in this list. So, you must be wondering why would we want to write a program like this. You are generating random numbers and you are finding a number there, will this be the beginning of the world or the end of the world, why are we even writing a program like this?

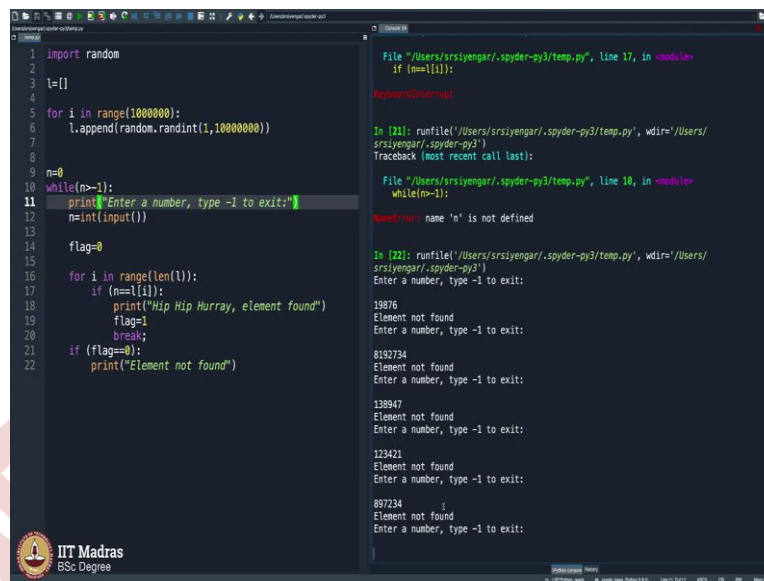
I am trying to tell you that you can do certain things really fast. Many of the mandarin things that we do in our life can be done really fast if we can write a piece of code. Of course, we stumble, we struggle in writing a simple piece of code like this, but then that is how programming is, but at the end of the day you automate things, you see execute it and a big lists gets created and it says element not found. So, now, let me modify this slightly and then try to ask you to type in for an element, it will check whether the element is present or not.

So, what should I do? I have a list l, now list l is generated, it is a big list. Now what I will do is I will print 'enter a number' and after that I will take the number that you enter, input. As you all know this should be integer. And then I will initialise flag equals 0 and check if n present or not. So, instead of a I will put n in fact, I would have star, I could have called it n a itself there, but never mind.

So, I will just go ahead and then execute it. It says enter a number, type minus 1 to exit. I say 78, it says element not found, element not found, element not found, it is not stopping that is because I made an error here, I made a mistake here.



(Refer Slide Time: 12:58)



```
1 import random
2
3 t=[]
4
5 for i in range(1000000):
6     t.append(random.randint(1,10000000))
7
8
9 n=0
10 while(n>-1):
11     print("Enter a number, type -1 to exit:")
12     n=int(input())
13
14     flag=0
15     for i in range(len(t)):
16         if (n==t[i]):
17             print("Hip Hip Hurray, element found")
18             flag=1
19             break;
20     if (flag==0):
21         print("Element not found")
22
```

File "/Users/srsiyengar/.spyder-py3/temp.py", line 17, in <module>
if (n==t[i]):
KeyboardInterrupt

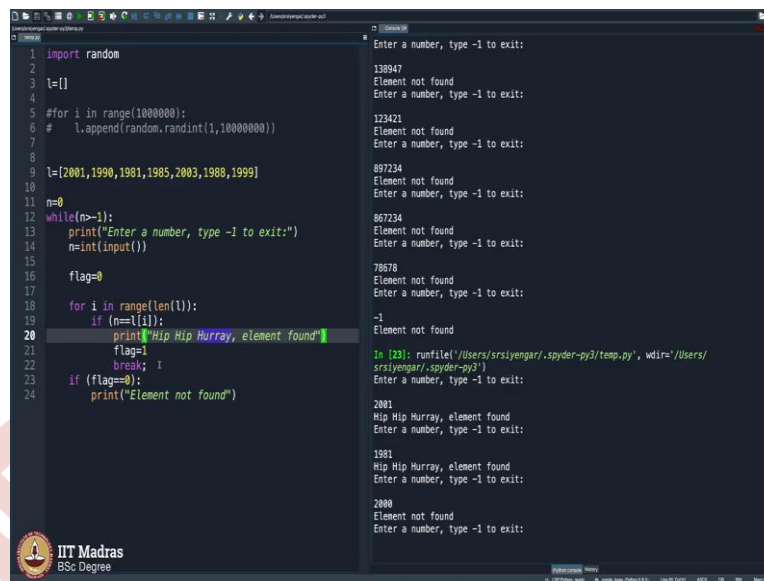
In [21]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
Traceback (most recent call last):
File "/Users/srsiyengar/.spyder-py3/temp.py", line 10, in <module>
while(n>-1):
NameError: name 'n' is not defined

In [22]: runfile('/Users/srsiyengar/.spyder-py3/temp.py', wdir='/Users/srsiyengar/.spyder-py3')
Enter a number, type -1 to exit:
19876
Element not found
Enter a number, type -1 to exit:
8197234
Element not found
Enter a number, type -1 to exit:
138947
Element not found
Enter a number, type -1 to exit:
123421
Element not found
Enter a number, type -1 to exit:
897234
1
Element not found
Enter a number, type -1 to exit:

Let me stop this. I want this to be interactive. I will put this inside, what do I mean by that? you will get to know. This enter a number type minus 1 to exit should keep happening. So, I will initialize n to be let us say 0, n is 0 I need time initialize n that is why it is throwing an error here, n is 0, please do not worry about my editor throwing errors, you know that is how we all code. That is when logic converges. This is how you should get comfortable coding.

So, n equals 0 initialized and then while n is greater than minus 1, definitely yes, n is greater than minus 1 because n is 0, it enters here, it says enter a number, type minus 1 to exit. I execute, it says enter a number, type minus 1 to exit. I say 19876. So, it says element not found. Enter a number, type minus 1 to exit. I type some random number which is element not found. I can keep doing this, keep doing this, keep doing this.

(Refer Slide Time: 14:17)



```
1 import random
2 l=[]
3
4 #for i in range(1000000):
5 #    l.append(random.randint(1,10000000))
6
7
8 l=[2001,1990,1981,1985,2003,1988,1999]
9
10 n=0
11 while(n>=1):
12     print("Enter a number, type -1 to exit:")
13     n=int(input())
14
15     flag=0
16     for i in range(len(l)):
17         if (n==l[i]):
18             print("Hip Hip Hurray, element found")
19             flag=1
20             break;
21         if (flag==0):
22             print("Element not found")
```

Enter a number, type -1 to exit:
138947
Element not found
Enter a number, type -1 to exit:
123421
Element not found
Enter a number, type -1 to exit:
897234
Element not found
Enter a number, type -1 to exit:
897234
Element not found
Enter a number, type -1 to exit:
78578
Element not found
Enter a number, type -1 to exit:
-1
Element not found
In [29]: runfile('Users/srsiyengar/.spyder-py3/temp.py', wdir='Users/srsiyengar/.spyder-py3')
Enter a number, type -1 to exit:
2001
Hip Hip Hurray, element found
Enter a number, type -1 to exit:
1981
Hip Hip Hurray, element found
Enter a number, type -1 to exit:
2000
Element not found
Enter a number, type -1 to exit:

In fact none of these elements that I am trying to find is present. So, I will type minus 1 and I will come outside. So, a good illustration is actually to type a list of your choice maybe you can type a list of your, of all our day of births maybe there are 2001 borns here I am sure there will be some 1990 borns here, 1981, 1985, 2000, I am sure all of you are 18 plus, so the minimum here would be 2003 and so on. I will type some numbers here, 1988, 1999 and so on.

So, now when I execute this, I will comment this just so that you can see what are the, whether the program is running fine or not. In fact, if the number is, is you are typing a number that is present here, it should say present here, otherwise no. So, I say 2001, it says 'Hip Hip Hurray, number element found'. I say 1981, it says 'Hip Hip Hurray'. But then as you can see the year 2000 is not here. So, if I say 2000, it will throw an error. By error I mean it will say number not found and that is indeed true, the number is not here in the list.

So, using the for loop I created a big list, just so that you realize even the, it is like finding a needle in a big hay stack, but still your program goes ahead and then finds it. Some of you are wondering why am I writing such a complicated program. Some of you are wondering why are we talking so trivial things. So, please be patient, as it always I have been telling this, we are going to make things complicated for you slowly.

So, this week's discussion as I have been telling is about a few pieces of code, I mean some 8 to 10 codes, the programs that we will be revisiting again and again. There is a reason why I am introducing this searching for an element in a list as boring as it can get, but it will get

very interesting in fact, we will try to see how you can find a small world in your browser, you always type this control f, how exactly does that work?

You basically are starting with chapter 1 of that search. So, what is the moral here, I mean, what is the bottom line here? Bottom line is that no matter how big the list is, we can search for an element there in a very seemingly big list, very very big list with around 1 million entries, you can go ahead and search through the entire thing and stop when you find the element. So, you may want to retype this code many a times and then get familiarized with it. We will now go ahead with sorting in our next video.

