

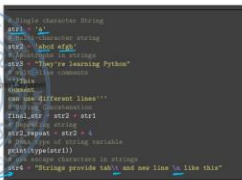
Advanced Business Decision Support Systems
Professor Deepu Philip
Department of Industrial Engineering and Management Engineering
Indian Institute of Technology, Kanpur
Professor Amandeep Singh
Imaging Laboratory
Dr. Prabal Pratap Singh
Indian Institute of Technology, Kanpur
Lecture 29
Data Types (Part 2 of 2)

Hello everyone, I welcome you all to the lecture series on Advanced Business Decision Support Systems and I am Prabal Pratap Singh from IIT Kanpur. We are discussing about the implementation aspects of decision support systems using Python. So, as you all know, till now we have covered the Elements and Classifications of programming languages and discussed what is Python programming language. Next, we also created our development environment using the Python executable file downloaded from its official site and we created our development environment on windows. So, today we will be discussing the data types in python. So, we already discussed few things about data types.

a = 40 40, 30, 30

Strings in Python

- * Strings are a kind of sequence data type
- * Collection of single/multiple character
- * Python identifies text inside a quote (single/double/triple) as a String.
- * Use triple quotes to create multi-line strings.
- * Using 'type' we can see that strings variables are objects of class str → Everything is an object in Python.
- * Use \n, \t escape sequences to insert new lines or tab character (white space) in a string.



```
#!/usr/bin/env python
str1 = 'A'
str2 = "Amandeep Singh"
str3 = 'Dr. Prabal Pratap Singh'
str4 = """Strings provide tab and new line \n like this"""

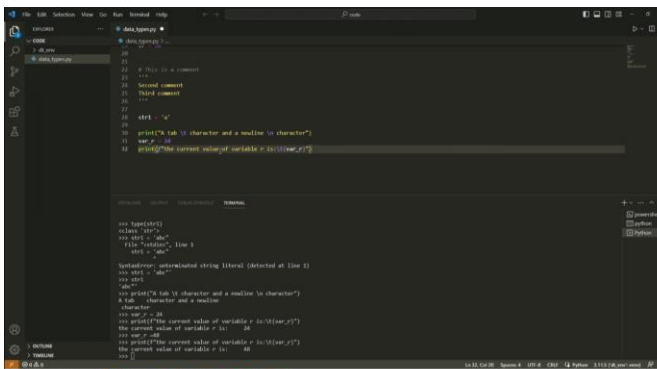
str1_upper = str1.upper()
str2_lower = str2.lower()
str3_title = str3.title()

str1_upper, str2_lower, str3_title

str1_upper == str2_lower
str2_lower == str3_title

print(type(str1))
print(type(str2))
str4 = "Strings provide tab and new line \n like this"
```

Dr. Prabal Pratap Singh Data Types in Python 8 / 12



```
1 # This is a comment
2
3 # Second comment
4 # Third comment
5
6 str1 = 'A'
7
8 print("A tab \t character and a newline \n character")
9 str_p = "Amandeep Singh"
10 print("The current value of variable 's' is:", str_p)
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

So, moving to our next data type that is Strings, which is the most common type of data type. The very first special thing about strings is that, strings are a kind of sequence data types. So, what are the Sequence data types? So, we until now, we have seen that, if we are defining a variable let us say, a is equal to 40. So, the memory storing one particular value in a particular location, but what happens, if we need to store multiple values in a sequential order like 40, then 20, then 30.

So, let us say that how many kind of fruits are available in a particular warehouse. So, let us say, in the first warehouse, there are 40 fruits, in the second, there are only 20 kinds, in the third warehouse, there is 30. So, this can be stored in a particular memory location and this will become a Sequential data type. So, strings also are classified as sequential data type instead of using numbers, they are using characters to store in sequences. So, these are nothing but collection of single or multiple characters.

So, here we can see on the right, that a single character string can look like this. Here, we are only using a particular character, this can be a, b, c or anything and we are storing it in a variable string 1 (str1). So, if again, I am repeating that if we will write here it as 1 str, then this will not be a valid variable, but str1 is a valid variable name. The next thing is Multi character string which can be written as here. So, this complete str2 is a single variable that is holding abcd space efgh right and to define a particular character as a string in python, we will use single quotes or double quotes or triple quotes.

So, single quote and double quotes are interchangeable, if you are using single quotes to start a string, then you need to end the single quote to end the string. If you mix and match these two quotes like if you start with double quote and end with single quote, then this will be a syntax error. So, let me write this as well, Python identifies text inside a quote, quote can be single, double, triple as a string. We can use triple quotes to create Multi line strings. So, we can see, this in our environment like, to create a multi line string.

So, we have seen that, a single line comment is like this is a comment. Now, to create multiple line comments, we can use second comment or you can write third comment. Otherwise, what you can do is, you start with a triple quote and remove these. So, this is a valid multi line string and the interpreter will not throw an error here. Again, let us define a simple string `str1 = 'a'`.

So, if you run this, it stores the interpreter, now stores str1 as a string and you can check the type of this variable using the type function `str1` and it shows that it is using a class string. Now, if you write `string str1 = 'abc'`. So, this is a syntax error here unterminated string literal. So, the interpreter is saying that, you started it with a single quote, but you have not yet terminated it. So, if you just change the other thing with a single quote.

Now, this is a valid string. So, you can see, the value of `str1`, what holds is `abc` with a double quote. So, this whole `abc` and double quote is the value of this variable `str1`. Now, I have already shown you that using `type` function, we can see that the variables are that string variables, are objects of class `string`, which confirms that everything is an object in python. The last thing about strings is that, we can use special characters like `\n` or `\t`, these are the escape sequences to insert new lines or tab characters that is, white spaces in a string.

So, here we can see, this string `4`, it is using this tab character `\t` and `\n`. So, what happens is, `\t` will be a tab character. So, after this tab, there will be a blank space and then `a` and `d` will start and here after line space, then a new line will start. So, we can check this, how the interpreter will print the statement. See, when I ran the above statement using shift plus enter, the interpreter says that this `\t` is missing from this output.

Why? Because `\t` was interpreted as a tab character here. So, there is a large gap between tab and character. Similarly, new line and character should be on the same line, but after new line, we have inputted a `\n` character. So, that is why, the interpreter moved to the next line. So, this is how you can use, you can customize your output using print statements.

Also, I have shown you in the presentation that, we can use f strings, these are f strings to use a dynamic variable into a string. So, that we can dynamically provide an output to the screen when the programmer wants. So, let us say, the current value of variable `r` is `24` and I will use tab character `\t` and open and close curly braces and then write the variable name here where `_r`. So, to use where `_r`, we first need to define it. So, let us say, `var _r` has a value of `24`.


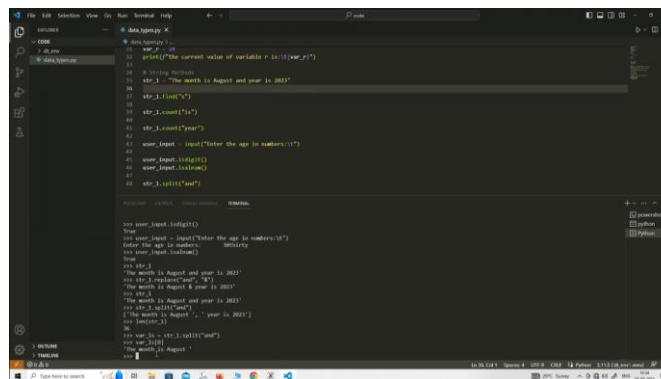
So, we first run `var _r` is `24`. Now, we write this. So, what happens in this print statement is after variable `r` is a colon `\t` will provide a tab character and then we should get the value of `var_r`, the value stored in `var r` as an output to the string. So, let us print this. So, here it is showing that the current value of variable `r` is `24`.

Now, if I change the value of `var r` in my interpreter as `48` and then again run this command. So, now it will print current value of variable `r` is `48`. So, where this thing will be useful is when, if you want to check the value of a particular variable throughout its computation during the progression of your code and you want to check, whether the value of variable is changing as per your computations or not, you can create these kind of print statements. Although, you should use logging statements for these, but that will be an advanced stage. So, before that, we can use these kind of print statements to check whether your computation is going well or not.

So, let us move further. So, until now, we have defined strings and saw how to create different kinds of print statements using strings, but since, string is an object and a class, as we will see in the advanced lectures of this python programming that class has data and methods inside it. So, every object has data and associated methods with it. So, since, string is an object, so string will also have this kind of behavior. So, strings will have predefined methods here.

Strings Methods

- * Use find to search inside a string.
- * Use count to count the occurrence of a particular string.
- * Use capitalize to make the text in capital case.
- * Use title to create the text in title case. *world of blender*
- * Check whether the string contains alphabets, alphanumeric, or only digits using isalpha(), isalnum(), isdigit()
- * Use replace to find and replace the provided substrings.
- * Use split to divide a string based on a particular character/string.
- * Use strip to remove the whitespace from the string.

```

# Python demonstrating various string methods
str1 = "The month is August and year is 2023."
str1.find("s")
str1.find("year")

# Count
str1.count("s")
str1.count("year")

# Capitalize
str1.capitalize()
str1.upper()

# Title Case
str1.title()

# Check whether the string contains alphabets, alphanumeric, or only digits using isalpha(), isalnum(), isdigit()
str1.isalpha()
str1.isalnum()
str1.isdigit()

# Replace
str1.replace("and", "&")

# Split
str1.split("and")
# Strip
str1.strip()

# User input
user_input = input("Enter the app in numbers: ")
user_input.isdigit()
user_input.isalpha()
user_input.isalnum()
user_input.strip()

```

So, some of the most used methods in strings are, we can use find to search inside a string. So, let us see, this as an example here. We are defining a new string which is month is August and year is 2023. Now, we can either find a single character s or we can find a complete word like year. So, the output of this first statement will be the location of the first occurrence of s in this string and similarly, for the complete word, if the word exists in this, then the interpreter will give you the output of the first occurrence of the location of the word year.

So, we can see this year, we will create another section by writing a new comment string methods and save the file and now create year, 1 is August and year is 2023. Now, we will find the occurrence of s character. So, it is showing 11. So, what this 11 hold is? As I have told you in earlier lecture that, the indexing starts with 0 in python and now, we know that string is a sequence data type. So, the indexing will start from 0.

So, the t in the memory location, it is not stored in the first location, it is stored in the 0th location. So, counting from here, the t is at the 0th location, h is at the first location. So, 0, 1, 2 the space is also holding a position. So, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11. So, s is at the 11th position in the memory.

So, this is what this find method is showing that the first occurrence of s is at 11th. Now, similarly we can find the location of year, if it is available in the string. So, it is showing 24. So, until s, it was 11 then, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24. So, wherever the word year starts, it will give you the location of the first character of that word.

So, it is giving the output as 24. So, let us save the script the next useful method is count. We can use count the occurrence of a particular string. So, here we can see that, in our string, there are two is. So, we can use a count method to find the number of occurrences of the word is and it should give the output as 2 for year, it should give the output as 1.

So, let us see what is. So, it is showing 2 and similarly, for the word year, str is 41. count year, it is only 1. So, you can use any word, any character using this count method. Next, useful method is Capitalize, you can use capitalize to make the text in capital case. So, these kind of methods are useful when we are trying to change the cases of our long text. So, this looks trivial for short strings, but let us say, you are changing the whole text of a particular e-book.

So, you can use this to change a huge amount of text in that complete e-book and see, whether if you need to capitalize all the sentences, then just use this single function and it will do the job. Similarly, let us say, if you have a complete paragraph and you want to change it into upper case that is, in all caps. So, you can use this method upper. So, what it will do is, it will change the complete text into upper case, next thing is, you can use Title method to create the text in title case. So, title case is let us say, we have a string world of wonder.

So, this is not in title case right now. So, title case will be, the first character of every word should be in capital letters. So, this function will do it. The next useful thing is, we can also check, whether the string contains alphabets, alphanumeric text or only digits using functions like isalpha(), isalnum ()or isdigit(). Now, this is very crucial when we are taking the input from users.

So, this we can see with the example. I have also written this example here. So, you can see that we can use this input function that is predefined in python to get some input from the user and user can do anything while providing the input. It can provide you just the number, just the text or the mixture of both and it does not matter what you want from the user, it can provide anything. So, you want to first check, whether user has

given the correct input or not. So, to do that, we can use these kinds of methods in strings.

So, let us see, how this works. So, we can create a new variable `user_input` and we will use the predefined input method `input()` and this input method, we can use just like that without putting any string inside it or we can create a string. So, that user will see the string while giving the input. So, let us say, that we want a digit input from the user. So, we can ask for enter the age in numbers `input()`. Now, the user should give an input in numbers.

But what if the user is not giving it in numbers? So, we can first run this statement. So, as soon as we run this statement, the interpreter has seen that there is a string provided inside the input with a `\t` character. So, it has created a tab character here and is asking for a input after the tab character. So, I can write `input()`, which is not a correct input, but user can do that. So, we can see, what `user_input` variable holds `user_input`.

So, it holds a string, but we need a number. So, when you are creating a long form of code. So, you have not the capability to check all these things by going to the interpreter and writing all this stuff. So, what happens is, you can use the prebuilt functions like, `user_input.isdigit()`. Now, these are all the methods that are available.

So, we can use `isdigit()` and run this. So, it is showing false here. So, Python is saying that, this input does not contain numbers, it contains something else. So, let us run this line again and give it a correct input as 30. Sorry, here 30 and now run this check method again and now it is showing true.

So, this is how we can use these things. So, let us say, if the user has given something very different: 30, thirty both in numbers and in words. So, this is an alphanumeric text and we can also check this whether the string contains an alphanumeric text or not. So, we used here the method `isalnum()` and it is showing true. So, this is how these string methods are useful. So, we can also use `replace()` to find and replace the provided substring.

So, we all have done it at least once in our life that, we use excel or in Microsoft word that, we need to find something and then replace it in the whole document. So, Python can also do it with this `replace()` method. So, here I have shown you an example like in this string, there is `end` and I want to replace it with this character `&`. So, I will just write `str1.replace('end', '&')`. So, I will first provide the `end` and then the `&` operator and it will do the job, we can see this quickly.

So, let us first check whether our string contains this statement or not yes. Now, we can write `str1.replace('end', '&')` and replace it with `&` characters. So, you can see

here that, our string has been changed and end is replaced by the ampersand character, but keep in mind that, when we are using these methods. So, these are not overwriting the string. So, str1 is still the same in the memory, but the output of this operation is here, it has not yet overwritten the original string.

So, some methods, in some kind, some different data types can overwrite things, but this replace method is not that kind. So, we will also look those methods that can replace things in the original string or list or other kind of data types. The other method is, we can use split to divide a string based on a particular character or string in the available text. So, here I have shown you the example that let us say, this complete sentence contains two different sub sentences like month is August and year is 2023. So, what if we want to divide this complete string into two on the basis of this end.

So, we can use the split function and provide this end as a string, then it will create a new data type. We are using strings here, but it will create a new data type, which is list and it will store the first sub sentence and the second sub sentence in the list. So, we can see here and it will also be an introduction to a list str1 . split on the basis of string a and d. So, you can see here that, our complete string is has been split into the month is August and the other one is space year is 2023. So, why these two spaces? Because when we wrote our string so, the python used this end and it gave as a output, this complete string including this space character.

Similarly, this space character is included in the second sub string and this is the notation of a list with a square bracket and a comma d limits different quantities or different elements of a list. So, this list has two elements. Similarly, we have also this one useful function to check the length of string. So, what if we need to find the length of a particular paragraph or text or complete book. So, we can use this simple function len str _ 1 and it is showing that, it contains 36 characters.

Let us say, this is a container and it stores 40. Now, sequential data types has this kind of array like behavior, which can store multiple quantities sequentially. So, the first location stores 40, second may store 50, third store 7, 4, 3 like this. So, storage is ok, but how to retrieve or modify these things? So, what python developers did, they configured it in the memory using this deterministic order. So, what they say is that, any sequential data type will start its location using the zeroth index. So, the first location is 0, second is 1, third is 2, similarly, 3 and 4.

So, this sequential data type which contains only integers has the address of the first value at 0, then 1, 2, 3 and 4. So, these are like containers that can store multiple values in a deterministic order. So, whenever you want to retrieve the seventh value, you will say that the name of this container let us say, bd and you will ask the interpreter to go to this variable bd and search at the second location, what is the value. This second location is actually 3, but as per the python interpreter, it is 2 and then, retrieve that value.

So, interpreter will give you the output as 7 right. So, string is one out of many sequence types available in python and programmers often need to retrieve or modify particular parts of sequences which comes under this slicing feature. So, either a programmer wants a single value or it can also ask for a series of values from 1 and going until the third location. So, this is getting a slice from this sequential data type and the interpreter will provide these three values 57 and 4. So, we will learn here how to extract or retrieve this kind of series of values from a sequential data type, what is the correct syntax for that. So, let me again mention this that, a sequence container assigns integer index to each unit in the sequence starting from 0.

Now, let us see the syntax of the slicing. So, let us say that, we have a variable with the name var _ name then, we will use the square brackets, we will open the square bracket then, we will write the start index, which is nothing, but this integer value starting from 0 until the length of the sequence, you can use any one of it then, you will use a colon operator. After that, you will write the end index. After that, one more optional parameter is to provide the step. So, what happens is, this step can skip some values while counting in the sequential data type.

So, let us say, we provide a step of 2 starting from 0. So, what happens is that, interpreter will give first value is 40, then it will skip two values and give the second value is 4 and since, there are no more values, so it will stop here. So, this kind of slicing with step of 2 will only provide two values if we start from 0. So, we can see this in our Visual Studio code by creating a new section. So, we have our string available str _ 1 and we can use the slicing here as well. So, let us say, str _ 1, open the bracket and write starting from first, we need to check until the seventh value as the end index.

types in strings which makes it general purpose and we can create an empty list using the list method or we can initialize a new list using square brackets.

So, this we can see here that, we are generating a new list by using just this list method and after that, we can also initialize a different list by using these square brackets and this string contains a number, a string, a variable itself which we have defined earlier and it can also do inline computations like this. So, 2 into 7 will be stored as 14 when the interpreter will run this code. So, nesting of list is also possible by storing a list inside a list. So, the example for this is here, the nest list variable will contain an empty list, which we defined here then, the first list which we defined here and these two numbers 45 and 22.

So, this list contains number data type, an empty list and a list as well. This list contains strings. So, this is called nesting of list and this variable holds different kinds of data types in itself and the numbering starts here as 0, 1, 2, 3. Now, since this is again a list. So, the next numbering starts from at the third, we start again with a square bracket and the numbering starts from 0, 1 for this abc, then 2 for the string, 2 variable and 3 for 2 into 7, 14.

So, this is how it works. Now, we can see the example of this quickly. So, let us first define empty list equals list. So, we can run this one by one. So, now, our interpreter has ran all the three commands and we have these three kind of list in our memory. So, we can print this like, first list and we can assess the different kinds of elements using the square notation. As we have discussed earlier, like first since, there are three numbers, but the indexing starts from 0.

So, the first value is available at the 0th location. So, 0, 1, 2. So, to get the string value from the list, we will write 1 here. I have not used the correct name of the list variable first list. So, it is giving the output as abc. Now, let us see what next list contains.

So, this contains an empty list, then two number data types and the other list. So, to assess abc again from this nested list, we will write next list square bracket and we will first give the locations available in this next list. So, that is 0th location for this empty list 1, 2 and at the third location, the other list is available.

So, we will go to the third location first, then we will close the square brackets and then we will again start a new square bracket to assess the location of abc in the second list and we have already seen that, this is available at 0 and 1 first location. So, this also gives abc. So, to assess nested list, we use these kind of multiple square notations to assess a particular element from the next list.

So, next important thing is list is a new table data type that is, it can add delete, modify the contents of a list after initializing it. So, this list sequence data type is a new table

data type, it can modify anything in the list, but the string data type which we have seen earlier strings are immutable. So, we can look at an example of this here. So, let us say that, the mission name is Chandrayan and we want to change the last n to the capital N.

So, the first thing which we should know is that, we can also in the last slide, I have shown you that we can initialize Sequence data type and then, start the indexing from 0, but what if a programmer wants to access the last location of the data contained in a sequence data type?

So, to do that, Python programmers define that, we can also access the last location by using the negative sequencing. So, negative sequencing does not start with 0 from the right, it starts from minus 1, minus 2, minus 3, minus 4, minus 5. So, we can see here that, to access this last n, we can use `mission_name` and square notation with the minus n and to overwrite that value, we can use this capital N as a string. So, let us see this and we will provide a new section here with immutable t and let us initialize our variable mission name of course, Chandrayan, since it is a sequence data type. So, we can access the last location by using minus indexing and it will provide you the last character as n.

Now, to overwrite this character in string, you can say that, assign this n character as a capital N, but the interpreter should throw an error here like this, type error string object does not support item assignment.

Why? Because strings are immutable, but let us say, we want to change this value in nest list 22 to 44. So, we can do this. Let us first check what nest list contains.

So, it contains the same things. Since, we have not modified anything. So, nest list and assess this 22 location as 0, 1. So, the value 22 is available at the first location at the index 1. So, to overwrite this, we just need to assign it as 44.

Now, there are no errors. So, let us again check nest list. Now, it has been overwritten with 44. So, this way, you can mutate a list, but cannot mutate a string. You can also mutate this same location with a string and it will work as well. So, now, the same thing is towards the string. So, this is how a string can handle different kinds of data types and that is different kinds of data types and that is why, we usually call it a general purpose data type.

List Methods

- * Use `append` method to add an element at the end of the list.
- * Use `insert` method to add an element at a specific location
- * Use `pop` method to remove the element from the end of the list.
- * Use `extend` to add any items to a list.
- * Use `sort` to the list in a particular order.
 - ↳ reverse
 - ↳ key

```
socialapps = ["Facebook", "i", "Instagram", "WhatsApp", "Google"]
socialapps.append("Threads")
socialapps.insert(2, "Nextdoor")
Dr. Prabal Prasad, Dept. of Computer Science, IIT Bombay
socialapps.pop()
socialapps.extend(["Zoom", "Meet", "Teams"])
socialapps.sort(reverse=True)
socialapps.sort(key=len)
```

```

1 # Creating an Python
2
3 # Creating list
4 first_list = [1,2,3]
5
6 # Inserting element
7
8 # Insertion
9
10 # Insertion
11
12 # Insertion
13
14 # Insertion
15
16 # Insertion
17
18 # Insertion
19
20 # Insertion
21
22 # Insertion
23
24 # Insertion
25
26 # Insertion
27
28 # Insertion
29
30 # Insertion
31
32 # Insertion
33
34 # Insertion
35
36 # Insertion
37
38 # Insertion
39
40 # Insertion
41
42 # Insertion
43
44 # Insertion
45
46 # Insertion
47
48 # Insertion
49
50 # Insertion
51
52 # Insertion
53
54 # Insertion
55
56 # Insertion
57
58 # Insertion
59
60 # Insertion
61
62 # Insertion
63
64 # Insertion
65
66 # Insertion
67
68 # Insertion
69
70 # Insertion
71
72 # Insertion
73
74 # Insertion
75
76 # Insertion
77
78 # Insertion
79
80 # Insertion
81
82 # Insertion
83
84 # Insertion
85
86 # Insertion
87
88 # Insertion
89
90 # Insertion
91
92 # Insertion
93
94 # Insertion
95
96 # Insertion
97
98 # Insertion
99
100 # Insertion
101
102 # Insertion
103
104 # Insertion
105
106 # Insertion
107
108 # Insertion
109
110 # Insertion
111
112 # Insertion
113
114 # Insertion
115
116 # Insertion
117
118 # Insertion
119
120 # Insertion
121
122 # Insertion
123
124 # Insertion
125
126 # Insertion
127
128 # Insertion
129
130 # Insertion
131
132 # Insertion
133
134 # Insertion
135
136 # Insertion
137
138 # Insertion
139
140 # Insertion
141
142 # Insertion
143
144 # Insertion
145
146 # Insertion
147
148 # Insertion
149
150 # Insertion
151
152 # Insertion
153
154 # Insertion
155
156 # Insertion
157
158 # Insertion
159
160 # Insertion
161
162 # Insertion
163
164 # Insertion
165
166 # Insertion
167
168 # Insertion
169
170 # Insertion
171
172 # Insertion
173
174 # Insertion
175
176 # Insertion
177
178 # Insertion
179
180 # Insertion
181
182 # Insertion
183
184 # Insertion
185
186 # Insertion
187
188 # Insertion
189
190 # Insertion
191
192 # Insertion
193
194 # Insertion
195
196 # Insertion
197
198 # Insertion
199
200 # Insertion
201
202 # Insertion
203
204 # Insertion
205
206 # Insertion
207
208 # Insertion
209
210 # Insertion
211
212 # Insertion
213
214 # Insertion
215
216 # Insertion
217
218 # Insertion
219
220 # Insertion
221
222 # Insertion
223
224 # Insertion
225
226 # Insertion
227
228 # Insertion
229
230 # Insertion
231
232 # Insertion
233
234 # Insertion
235
236 # Insertion
237
238 # Insertion
239
240 # Insertion
241
242 # Insertion
243
244 # Insertion
245
246 # Insertion
247
248 # Insertion
249
250 # Insertion
251
252 # Insertion
253
254 # Insertion
255
256 # Insertion
257
258 # Insertion
259
260 # Insertion
261
262 # Insertion
263
264 # Insertion
265
266 # Insertion
267
268 # Insertion
269
270 # Insertion
271
272 # Insertion
273
274 # Insertion
275
276 # Insertion
277
278 # Insertion
279
280 # Insertion
281
282 # Insertion
283
284 # Insertion
285
286 # Insertion
287
288 # Insertion
289
290 # Insertion
291
292 # Insertion
293
294 # Insertion
295
296 # Insertion
297
298 # Insertion
299
300 # Insertion
301
302 # Insertion
303
304 # Insertion
305
306 # Insertion
307
308 # Insertion
309
310 # Insertion
311
312 # Insertion
313
314 # Insertion
315
316 # Insertion
317
318 # Insertion
319
320 # Insertion
321
322 # Insertion
323
324 # Insertion
325
326 # Insertion
327
328 # Insertion
329
330 # Insertion
331
332 # Insertion
333
334 # Insertion
335
336 # Insertion
337
338 # Insertion
339
340 # Insertion
341
342 # Insertion
343
344 # Insertion
345
346 # Insertion
347
348 # Insertion
349
350 # Insertion
351
352 # Insertion
353
354 # Insertion
355
356 # Insertion
357
358 # Insertion
359
360 # Insertion
361
362 # Insertion
363
364 # Insertion
365
366 # Insertion
367
368 # Insertion
369
370 # Insertion
371
372 # Insertion
373
374 # Insertion
375
376 # Insertion
377
378 # Insertion
379
380 # Insertion
381
382 # Insertion
383
384 # Insertion
385
386 # Insertion
387
388 # Insertion
389
390 # Insertion
391
392 # Insertion
393
394 # Insertion
395
396 # Insertion
397
398 # Insertion
399
400 # Insertion
401
402 # Insertion
403
404 # Insertion
405
406 # Insertion
407
408 # Insertion
409
410 # Insertion
411
412 # Insertion
413
414 # Insertion
415
416 # Insertion
417
418 # Insertion
419
420 # Insertion
421
422 # Insertion
423
424 # Insertion
425
426 # Insertion
427
428 # Insertion
429
430 # Insertion
431
432 # Insertion
433
434 # Insertion
435
436 # Insertion
437
438 # Insertion
439
440 # Insertion
441
442 # Insertion
443
444 # Insertion
445
446 # Insertion
447
448 # Insertion
449
450 # Insertion
451
452 # Insertion
453
454 # Insertion
455
456 # Insertion
457
458 # Insertion
459
460 # Insertion
461
462 # Insertion
463
464 # Insertion
465
466 # Insertion
467
468 # Insertion
469
470 # Insertion
471
472 # Insertion
473
474 # Insertion
475
476 # Insertion
477
478 # Insertion
479
480 # Insertion
481
482 # Insertion
483
484 # Insertion
485
486 # Insertion
487
488 # Insertion
489
490 # Insertion
491
492 # Insertion
493
494 # Insertion
495
496 # Insertion
497
498 # Insertion
499
500 # Insertion
501
502 # Insertion
503
504 # Insertion
505
506 # Insertion
507
508 # Insertion
509
510 # Insertion
511
512 # Insertion
513
514 # Insertion
515
516 # Insertion
517
518 # Insertion
519
520 # Insertion
521
522 # Insertion
523
524 # Insertion
525
526 # Insertion
527
528 # Insertion
529
530 # Insertion
531
532 # Insertion
533
534 # Insertion
535
536 # Insertion
537
538 # Insertion
539
540 # Insertion
541
542 # Insertion
543
544 # Insertion
545
546 # Insertion
547
548 # Insertion
549
550 # Insertion
551
552 # Insertion
553
554 # Insertion
555
556 # Insertion
557
558 # Insertion
559
560 # Insertion
561
562 # Insertion
563
564 # Insertion
565
566 # Insertion
567
568 # Insertion
569
570 # Insertion
571
572 # Insertion
573
574 # Insertion
575
576 # Insertion
577
578 # Insertion
579
580 # Insertion
581
582 # Insertion
583
584 # Insertion
585
586 # Insertion
587
588 # Insertion
589
590 # Insertion
591
592 # Insertion
593
594 # Insertion
595
596 # Insertion
597
598 # Insertion
599
600 # Insertion
601
602 # Insertion
603
604 # Insertion
605
606 # Insertion
607
608 # Insertion
609
610 # Insertion
611
612 # Insertion
613
614 # Insertion
615
616 # Insertion
617
618 # Insertion
619
620 # Insertion
621
622 # Insertion
623
624 # Insertion
625
626 # Insertion
627
628 # Insertion
629
630 # Insertion
631
632 # Insertion
633
634 # Insertion
635
636 # Insertion
637
638 # Insertion
639
640 # Insertion
641
642 # Insertion
643
644 # Insertion
645
646 # Insertion
647
648 # Insertion
649
650 # Insertion
651
652 # Insertion
653
654 # Insertion
655
656 # Insertion
657
658 # Insertion
659
660 # Insertion
661
662 # Insertion
663
664 # Insertion
665
666 # Insertion
667
668 # Insertion
669
670 # Insertion
671
672 # Insertion
673
674 # Insertion
675
676 # Insertion
677
678 # Insertion
679
680 # Insertion
681
682 # Insertion
683
684 # Insertion
685
686 # Insertion
687
688 # Insertion
689
690 # Insertion
691
692 # Insertion
693
694 # Insertion
695
696 # Insertion
697
698 # Insertion
699
700 # Insertion
701
702 # Insertion
703
704 # Insertion
705
706 # Insertion
707
708 # Insertion
709
710 # Insertion
711
712 # Insertion
713
714 # Insertion
715
716 # Insertion
717
718 # Insertion
719
720 # Insertion
721
722 # Insertion
723
724 # Insertion
725
726 # Insertion
727
728 # Insertion
729
730 # Insertion
731
732 # Insertion
733
734 # Insertion
735
736 # Insertion
737
738 # Insertion
739
740 # Insertion
741
742 # Insertion
743
744 # Insertion
745
746 # Insertion
747
748 # Insertion
749
750 # Insertion
751
752 # Insertion
753
754 # Insertion
755
756 # Insertion
757
758 # Insertion
759
760 # Insertion
761
762 # Insertion
763
764 # Insertion
765
766 # Insertion
767
768 # Insertion
769
770 # Insertion
771
772 # Insertion
773
774 # Insertion
775
776 # Insertion
777
778 # Insertion
779
780 # Insertion
781
782 # Insertion
783
784 # Insertion
785
786 # Insertion
787
788 # Insertion
789
790 # Insertion
791
792 # Insertion
793
794 # Insertion
795
796 # Insertion
797
798 # Insertion
799
800 # Insertion
801
802 # Insertion
803
804 # Insertion
805
806 # Insertion
807
808 # Insertion
809
810 # Insertion
811
812 # Insertion
813
814 # Insertion
815
816 # Insertion
817
818 # Insertion
819
820 # Insertion
821
822 # Insertion
823
824 # Insertion
825
826 # Insertion
827
828 # Insertion
829
830 # Insertion
831
832 # Insertion
833
834 # Insertion
835
836 # Insertion
837
838 # Insertion
839
840 # Insertion
841
842 # Insertion
843
844 # Insertion
845
846 # Insertion
847
848 # Insertion
849
850 # Insertion
851
852 # Insertion
853
854 # Insertion
855
856 # Insertion
857
858 # Insertion
859
860 # Insertion
861
862 # Insertion
863
864 # Insertion
865
866 # Insertion
867
868 # Insertion
869
870 # Insertion
871
872 # Insertion
873
874 # Insertion
875
876 # Insertion
877
878 # Insertion
879
880 # Insertion
881
882 # Insertion
883
884 # Insertion
885
886 # Insertion
887
888 # Insertion
889
890 # Insertion
891
892 # Insertion
893
894 # Insertion
895
896 # Insertion
897
898 # Insertion
899
900 # Insertion
901
902 # Insertion
903
904 # Insertion
905
906 # Insertion
907
908 # Insertion
909
910 # Insertion
911
912 # Insertion
913
914 # Insertion
915
916 # Insertion
917
918 # Insertion
919
920 # Insertion
921
922 # Insertion
923
924 # Insertion
925
926 # Insertion
927
928 # Insertion
929
930 # Insertion
931
932 # Insertion
933
934 # Insertion
935
936 # Insertion
937
938 # Insertion
939
940 # Insertion
941
942 # Insertion
943
944 # Insertion
945
946 # Insertion
947
948 # Insertion
949
950 # Insertion
951
952 # Insertion
953
954 # Insertion
955
956 # Insertion
957
958 # Insertion
959
960 # Insertion
961
962 # Insertion
963
964 # Insertion
965
966 # Insertion
967
968 # Insertion
969
970 # Insertion
971
972 # Insertion
973
974 # Insertion
975
976 # Insertion
977
978 # Insertion
979
980 # Insertion
981
982 # Insertion
983
984 # Insertion
985
986 # Insertion
987
988 # Insertion
989
990 # Insertion
991
992 # Insertion
993
994 # Insertion
995
996 # Insertion
997
998 # Insertion
999
1000 # Insertion

```

Now, similar to the other sequence data types like, string, list also contains its own methods. So, we can use append method to add an element at the end of the list. We can also use insert method to add an element at a specified location. So, you may ask both of these methods are adding something to a list that a user has defined. So, what is the difference? Now, the difference here is at a specified location or at the end of the list. So, you can see here that, we have defined a new list named as social apps and it contains these different kinds of social applications available.

So, if we use the append method, then this newly available threads, social application will be appended to the end of the list. However, if we insert at a particular location that is here, the programmer is providing at the index location 2, we should insert this string mastodon. So, the list will contain this at the location 2.

So, we can quickly see. We will first initialize our variable by providing the values. So, now we can check the difference between append and insert method. So, social apps .append, we can append threads and it should go to the end of the list. So, let us see. Now, you can see that, threads is available at the end and as I have already told you that, in strings, there are some functions, there are some methods that overwrite the contents or there are some methods that do not overwrite the contents.

Similarly, in list also, this method will overwrite the content. This will not just provide you a new output. It has overwritten our original parent list. Now, use the insert method and see the Facebook is available at the zeroth location. The next is 1, 2. So, we need to provide this new string at the second location. So, we will first give the location as 2 and then, we will provide our string which is master and this will insert the this string before Instagram.

So, let us see again what happened. So, you can see that, at the second location 0, 1, 2, the mastodon string is available before Instagram. So, next important method available in list is, we can use pop method to remove the element from the end of the list. So, by default, it will remove from the end, but if you want to remove a particular element, then we can also specify the location. Also, we can use extend to add any kind of iterable.

Iterable is nothing but sequence data types like list or we will see dictionaries as well in our further lectures.

We can also use `sort`; to sort the list in a particular order and this `sort` method has two arguments with it like `reverse` and `key`. `key`. So, what `reverse` does, it is a `bool` means, it is a binary argument it will be either true or false. So, if true, it will sort the complete list in a reverse order. Similarly, if we are using a `key`, then we can provide a function which we will discuss in our further lectures, how to write a function and how to return values from it and if it returns some values based on those outputs, the `sort` method can sort the complete list on the returned output from that function.

So, we can also see this on our code. So, we can use the `pop` method to first see, what it performs by default, `social_x.pop()`. If there is a method, then it should use an opening and closing parenthesis. Let us run. So, it printed the last value from the list and should have removed from the container, that is list.

So, let us see whether it has removed or not that is, whether it has overwritten the values or not. So, now you can see that, the interpreter does not contain threads in its original list, but what if we want to remove a particular value at a particular location using the `pop` method? So, we can do that as well, `social_apps.pop(2)` and provide the particular location like, two where we inserted `mastodon` earlier and every time we use `pop`, it will always provide the return output that has been removed from the list.

So, we can see here that, earlier it showed threads, now it is showing `mastodon` and we can check, whether it has removed or not. So, now you can see that there is no `mastodon` in the list. Similarly, we can extend this list by using the `extend` function, using the `extend` method `social_apps.extend(nested_list)` and earlier we have a nested list.

So, we can use that list here, let us run this and see what happened `social_apps`. So, now our original `social_apps` list has been extended with a new list, nested list which we used earlier and you can see after this, `google_plus_string`, we have the empty list which was available in nested list. Then, `def`, then 45 and other another list which was contained in the nested list. So, this way, we can extend a list with another list. The last thing which we discussed is to `sort`.

So, if we try to run this again `social_apps` original `social_apps` and then try to `sort` this with default input. So, now these are following a particular alphabetical sorting method in ascending order like, `f` (facebook) is the first, `google plus` is the next, `f`, `g`, `i`, `w`, `x`, but originally it was something different `f`, then `x`, then `instagram`, then `whatsapp`, then `google plus`. So, this way, what if we want to provide this sorting order in reverse. So, we can use `social_apps.sort(reverse=True)` and then, it should sort this in reverse order.

So, now you can see the last element earlier `x` has become first, then WhatsApp, then Instagram, google plus and Facebook. So, these are some of the major data types that python contains and we will discuss further data types like dictionaries, sets, tuples in the upcoming lectures, but before that to understand few very beautiful characteristics of the list and other data types, we need to first learn what are the control flow statements? what are the looping statements and other things in python.

So, that we can use features like, list, comprehensions and other things. So, in the next lecture, we will discuss these control statements first and then we will complete our data types like, dictionaries and other available data types in python. So, thank you very much. I hope you are practicing all these things in your development environment as well. If you find any difficulties, please reach out us to in forums and we will help you out. Thank you.