



SNS COLLEGE OF ENGINEERING
Kurumbapalayam (Po), Coimbatore – 641 107



An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF CSE (IoT & CYBER SECURITY INCLUDING BLOCKCHAIN TECHNOLOGY)



19IT103 – COMPUTATIONAL THINKING AND PYTHON PROGRAMMING

- ❖ A readable, dynamic, pleasant, flexible, fast and powerful language

Recap

- Strings Read/Convert
- Indexing strings using []
- Looping through strings with **for** and **while**
- Concatenating strings with **+**
- Strings are immutable

Agenda

- Slicing Strings
- String Functions and Methods
- Sting Module
- Lists as arrays

Slicing Strings - What is it?



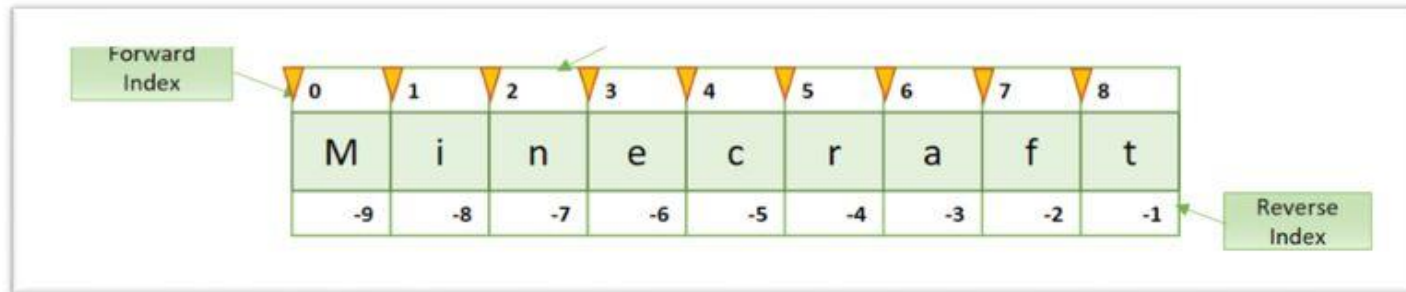
- ❖ Slicing means taking part of a string as in cutting it up
- ❖ It is often useful to take part of a string or analyse it or use it in another part of the code.
- ❖ Example:
 - you have been asked to write a program to check if a string contains a valid email address.
 - One of the checks will mean that you need to slice the string taking all the characters from the left hand part of the string up to the @ character



Slicing Strings -How to do it



- ❖ Strings have indices just like lists:
 - ❑ Forward index starts at 0 from the LHS
 - ❑ Reverse index starts at -1 from the RHS



- ❖ To specify a slice you need to
 - ❑ give the START position from the left hand side
 - ❑ give the END position from the left hand side
 - ❑ Separate the two values with a colon

- ❖ Example:

```
newString = myString [1:3]
```

Slicing Strings - Some Rules



- ❖ If you don't specify a **START** it defaults to the beginning:

```
newString = myString [:3]
```

This means start at position 0 end at position 3

- ❖ If you don't specify a **END** it defaults to the end:

```
newString = myString [3:]
```

This means start at position until the end

When SLICING, think of the index as pointing to the character to the left of the arrow

0	1	2	3	4	5	6	7	8
M	i	n	e	c	r	a	f	t
-9	-8	-7	-6	-5	-4	-3	-2	-1

```
newString = myString [1:3]
```

The content of newString would be:
in

Slicing Strings - Quickstart



0	1	2	3	4	5	6	7	8
M	i	n	e	c	r	a	f	t
-9	-8	-7	-6	-5	-4	-3	-2	-1

❖ First Character

```
firstChar = myString [0]
```

M

❖ Last Character:

```
firstChar = myString [-1]
```

t

❖ All characters but the first:

```
slice = myString[1:]
```

inecraft

❖ All characters but the last:

```
slice = myString[:-1]
```

Minecraf

String functions and methods

<code>len()</code>	<code>min()</code>	<code>max()</code>	<code>isalnum()</code>	<code>isalpha()</code>
<code>isdigit()</code>	<code>islower()</code>	<code>isupper()</code>	<code>isspace()</code>	<code>isidentifier()</code>
<code>endswith()</code>	<code>startswith()</code>	<code>find()</code>	<code>count()</code>	<code>capitalize()</code>
<code>title()</code>	<code>lower()</code>	<code>upper()</code>	<code>swapcase()</code>	<code>replace()</code>
<code>center()</code>	<code>ljust()</code>	<code>rjust()</code>	<code>center()</code>	<code>rstrip()</code>
<code>rstrip()</code>	<code>strip()</code>			

i) Converting string functions

<code>capitalize()</code>	Only First character capitalized
<code>lower()</code>	All character converted to lowercase
<code>upper()</code>	All character converted to uppercase
<code>title()</code>	First character capitalized in each word
<code>swapcase()</code>	Lower case letters are converted to Uppercase and Uppercase letters are converted to Lowercase
<code>replace(old,new)</code>	Replaces old string with nre string

Program:

```
str=input("Enter any string:")
print("String Capitalized:", str.capitalize())
print("String lower case:", str.lower())
print("String upper case:", str.upper())
print("String title case:", str.title())
print("String swap case:", str.swapcase())
print("String replace case:", str.replace("python", "python programming"))
```

Output:

```
Enter any string: Welcome to python
String Capitalized: Welcome to python
String lower case: welcome to python
String upper case: WELCOME TO PYTHON
String title case: Welcome To Python
String swap case: wELCOME TO PYTHON
String replace case: Welcome to python programming
```

ii) Formatting String functions

<code>center(width)</code>	Returns a string centered in a field of given width
<code>ljust(width)</code>	Returns a string left justified in a field of given width
<code>rjust(width)</code>	Returns a string right justified in a field of given width
<code>format(items)</code>	Formats a string

Program:

```
a=input("Enter any string:")  
print("Center alignment:", a.center(20))  
print("Left alignment:", a.ljust(20))  
print("Right alignment:", a.rjust(20))
```

Output:

```
Enter any string:welcome  
Center alignment:      welcome  
Left alignment: welcome  
Right alignment:      welcome
```


iii) Removing whitespace characters

<code>lstrip()</code>	Returns a string with leading whitespace characters removed
<code>rstrip()</code>	Returns a string with trailing whitespace characters removed
<code>strip()</code>	Returns a string with leading and trailing whitespace characters removed

```
ENV['BUNDLE_GEMFILE'] ||= File.expand_path('../Gemfile', __FILE__)

require 'bundler/setup' # Set up gems listed in the Gemfile.

require 'something'

def something
end
```

Leading space should not be visible

Trailing space

Program

```
a=input("Enter any string:")  
print("Left space trim:",a.lstrip())  
print("Right space trim:",a.rstrip())  
print("Left and right trim:",a.strip())
```

Output:

```
Enter any string:      welcome  
Left space trim: welcome  
Right space trim:      welcome  
Left and right trim: welcome
```

iv) Testing String/Character

isalnum()	Returns true if all characters in string are alphanumeric and there is atleast one character
isalpha()	Returns true if all characters in string are alphabetic
isdigit()	Returns true if string contains only number character
islower()	Returns true if all characters in string are lowercase letters
isupper()	Returns true if all characters in string are uppercase letters
isspace()	Returns true if string contains only whitespace characters.

Program

```
a=input("Enter any string:")  
print("Alphanumeric:",a.isalnum())  
print("Alphabetic:",a.isalpha())  
print("Digits:",a.isdigit())  
print("Lowecase:",a.islower())  
print("Upper:",a.isupper())
```

Output:

```
Enter any string:python  
Alphanumeric: True  
Alphabetic: True  
Digits: False  
Lowecase: True  
Upper: False
```

v) Searching for substring

Endswith()	Returns true if the strings ends with the substring
Startswith()	Returns true if the strings starts with the substring
Find()	Returns the lowest index or -1 if substring not found
Count()	Returns the number of occurrences of substring

Program

```
a=input("Enter any string:")  
print("Is string ends with thon:", a.endswith("thon"))  
print("Is string starts with good:", a.startswith("good"))  
print("Find:", a.find("ython"))  
print("Count:", a.count("o"))
```

Output:

```
Enter any string : welcome to python  
Is string ends with thon: True  
Is string starts with good: False  
Find: 12  
Count: 3
```

String Modules

- String module contains a number of functions to process standard Python strings
- **Mostly used string modules:**

```
string.upper()  
string.upper()  
string.split()  
string.join()  
string.replace()  
string.find()  
string.count()
```

Python
Programming

Example

```
import string
text="Monty Python Flying Circus"
print("Upper:", string.upper(text))
print("Lower:", string.lower(text))
print("Split:", string.split(text))
print("Join:", string.join(string.split(text),"+"))
print("Replace:", string.replace(text,"Python", "Java"))
print("Find:", string.find(text,"Python"))
print("Count", string.count(text,"n"))
```



Output

Upper: "MONTY PYTHON FLYING CIRCUS"

Lower: "monty python flying circus"

Split: ['Monty', 'Python', 'Flying', 'Circus']

Join : Monty+Python+Flying+Circus

Replace: Monty Java Flying Circus

Find: 7

Count: 3

Lists as arrays

- Both lists and arrays are used to store data in Python.
- Also these data structures allow indexing, slicing, and iterating.
- List is a built-in data structure whereas the *array* data structure belongs to the "must-import" category.
- ***NumPy* package** or the ***array* module** has the array functions.

Lists as arrays

Features of Lists	Features of Arrays
List items are enclosed in square brackets	Array items are enclosed in square brackets
Lists are ordered	Arrays are ordered
Lists are mutable	Arrays are mutable
List elements do not need to be unique	Array elements do not need to be unique
Elements can be of different data types	Depends on the kind of array used
Lists need not to be declared	Arrays need to be declared
Lists are not efficient storage structure for large amount of data	Arrays can store data very compactly
Lists cannot directly handle math operations	Arrays are great for numerical operations

*Array module
supports unique
types*

Lists as arrays

*List can be
converted into
arrays*

Program.

```
import array as arr
array_1 = arr.array("i", [3, 6, 9, 12])
print(array_1)
print(type(array_1))
```

*Unicode character
(b, B, u, h, H, i, l, l, L, q, Q, f
or d)*

Output:

```
array('i', [3, 6, 9, 12])
<class 'array.array'>
```

Lists as arrays

Example:

```
import numpy as np
array_2 = np.array(["numbers", 3, 6, 9, 12])
print (array_2)
print(type(array_2))
```

*numpy supports
different types*

Output:

```
['numbers' '3' '6' '9' '12']
<class 'numpy.ndarray'>
```

Lists as arrays

Example:

```
import numpy as np
array = np.array([3, 6, 9, 12])
division = array/3
print(division)
print (type(division))
```

*arrays can handle
math operations
directly*

Output:

```
[1. 2. 3. 4.]
<class 'numpy.ndarray'>
```

Summary

- Slicing means taking part of a string.
- To separate the starting and ending index of the slice ":" should be used.
- String functions and methods are categorized into converting, formatting, removing white space, testing and searching functions.
- String module has additional methods to support string operations.
- List is a built-in data structure whereas the *array* data structure belongs to the "must-import" category.
- List can easily be converted into arrays.

A yellow speech bubble with a pointed tail at the bottom right, set against a blue background. The words "THANK YOU" are cut out of the bubble in a bold, sans-serif font, revealing the blue background behind them.

THANK YOU