String functions :-

chr() → Convert an integers into character

ord() → " character to an integers

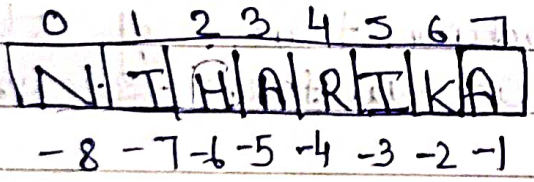
len() → Return the length of a String

str() → Return a string representation of an object.

→ This gives ASCII value of the character given.

S = 'Ram a string..'

len(s) O/P : 13



→ String

Have some predefined functions

- upper()
- lower()
- capitalize()
- title()
- replace()

Replace ()

```
S = "we are here to learn Python";
Print = ("Printing the string:")
Print (S)

Print ("Replacing Python to string")
S = S.replace ('Python', 'String')
```

'Python' → which is to change; 'String' → new changed string

Count ():- Counting the occurance of the given characters

```
Print ("counting the characters:")
Print (S.Count('a'))
```

7/3/2024

endswith () → It is used to return 'True' if we pass the character is same as in the string and 'False' if passed character is not available at the end of the given string.

Startswith () → It is uses to return 'True' if the passed character is present in the starting of the string and 'false' if the passing character is not present in the starting of the string.

Ex:- S = " we are here to learn Python from 9PCS"

```
print ("Printing the string:")
Print (S)
```

```
Print ("Ending with :")
print (S.endswith ('S'))

Print ("Starts With :"),
print (S.startswith ('W'))
```

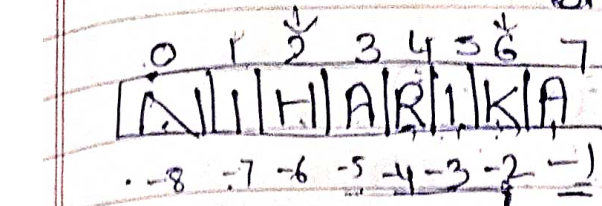
O/P:

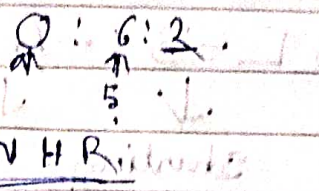
```
Printing the String:
We are here to learn Python from IPCS
Ending with
True
Starts with
True
```

Boolean String functions: Functions that return the o/p in the form of 'True' and 'False'

1. isalpha ()
2. isalnum ()
3. isspace ()
4. isdigit ()
5. isupper ()
6. islower ()

String Slicing :- It is uses to slice the string that is given.






```
S = "NIHARIKA"
Print ("Priting the String:")
Print (S2)
Print ("Print first character:")
Print (S2[0])
```

O/P:

N

```
Print ("Print characters from 0 to 3")
Print (S2[0:3])
```

O/P:

NIH

```
Print ("Print character from 3 to 6.")
Print (S2[3:6])
```

O/P:

ART

```
Print ("Print last character")
Print (S2[-1])
```

O/P:

A

```
Print ("Print last character to -6:")
Print (S2[-1:-6:-1])
```

-1 → from last

-6 → destination value + 1

-1 → increment of index value

o/p:

AKIRA

print ("Printing from -3 to -6"):

print (S2[-3:-6:-1])

↓ Starting index value; ↓ Ending index value; → increment of index (i.e.) Right to left (-1)

o/p:-

ARI

Print ("Skipping two characters from last")

Print (S2[-1:-6:-2])

o/p

KR

Print ("Skipping two characters")

Print (S2[2:6:2])

→ value is +ve because increment from left to right

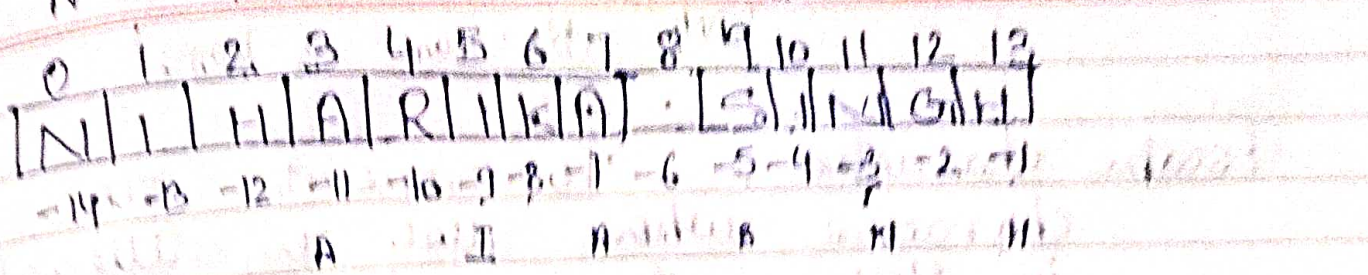
Print ("Printing Reverse")

Print (S2[::-1])

o/p

By default value is -1 → by default value is 1st index value

KIRAHIN



Print ("Print from start to 10")

Print (S[: 11])

↳ By default start index value is 0

Print ("Print from 10 to last")

Print (S[10 :])

↓ By default the last notation range is taken as the last index value).

⇒ f string method: → By using this type conversion is not needed if their is integer value of string.

Ex:-

Var = 123

Print (f 'a dog says {var}')

↓ Syntax

{} ↳ Placeholder

↓ Here 123 is integer value but we using as characters without their type conversion.

O/P:

a dog says 123

'My pet's name is Dogs'

Escape characters :— To insert the characters whose use is illegal in string, to maintain their use we have to use escape character.

Ex:—

```
txt = "We are the so-called \"Vikings\" from the north."
```

Escape character

O/P:

We are the so-called "Vikings" from the north.

Escape character	Result
\'	Single Quote
\\	Backslash
\n	Newline
\r	Carriage Return
\t	Tab
\b	Backspace
\f	Form Feed
\ooo	Octal Value
\xhh	Hex value