**Strings in Python:**

String is a collection of Unicode Characters.

**Examples:**

‘PDPU’

“PDPU”

“””Pandit Deendayal Energy University”””

‘’’Pandit Deendayal Energy University’’’

NMQuote1 = '\'When in doubt, disclose.\' – Narayan Murthy, Infosys.'

NMQuote2 = 'When in doubt, disclose.'

**Multiple Strings:**

* All but the last line ends with \.
* Enclosed text within “”” or ‘’’.
* st= ‘Text1’,

‘Text2’ -- error

**Accessing String Elements:**

Index value starts at 0. So last character’s index will be len()-1.

e.g. PDPU is stored inside as under:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 🡨 Positive Index Values |
| P | D | P | U |  |
| -4 | -3 | -2 | -1 | 🡨Negative Index values |

a = “PDPU”

b = a[0] # answer: P

c = a[3] # answer: U

d = a[-0] # answer: P , -0 or 0 are same.

e = a[-1] # answer: U

f = a[-4] # answer: P

**String Slicing- returns a substring of a string:**

**s= “PDEU College”**

* s[start:end] – Extract from start to end-1 with an increment of 1.
* s[start:] - Extract from start to end with an increment of 1.
* s[:end] - Extract from start to end – 1 with an increment of 1.
* s[-start:] – Extract from –start (included) to end with an increment of 1.
* s[:-end] – Extract from beginning to –end -1 with an increment of 1.
* s[start:end:step] - Extract from start to end-1 with an increment of 2.

**String Property:**

* Python strings are immutable – they can’t be changed.
  + s = “hi”
  + s[0] = ‘b’ 🡨 Error as we try to mutate the string.
  + s = ‘bye’ 🡨 s is a variable, it can change.
* **Type conversion**
  + int (‘0’)
  + float(“3.5”)
  + int(“-90”)
  + int (‘a’) – error

**String Operators:**

* Concatenation: Strings can be concatenated using +
  + str3 = str1 + str2
  + Ex: str3= ‘PDEU’+ ‘College’
  + Str3= ‘PDEU’ + 5 -- error
  + Str3= ‘PDEU’ + ‘5’-- no error
* Repetition Operator \*:
  + Str1= “PDEU” \* 3 -- PDEUPDEUPDEU
  + Str1= “Hello” \* 0 -- empty string
  + Str1= “Hello” \* -1 -- empty string
* Membership operator **in**
  + Finding substring using **in** from a string

1. print ( 'abc' in 'xyzabcdef') 🡨 True
2. if ('abc' in 'xyzabcdef') 🡨 condition is true

**Built-in functions**

* + uniname = “PDPU”
  + len(uniname) 🡨 returns 4
  + min(uniname) 🡨 returns ‘D’
  + max(uniname) 🡨 returns ‘U’
  + type(uniname) 🡨 <class ‘str’>
  + id(uniname) 🡨 returns memory address

**String Methods**

* + uniname.upper() 🡨 “PDPU” -- returns a new string
  + uniname.lower() 🡨 “pdpu” -- returns a new string
  + uniname.capitalize() 🡨 “Pdpu” -- returns a new string
  + uniname.title() 🡨 “Pdpu” # converts 1st character of each word to uppercase. -- returns a new string
  + uniname.swapcase() 🡨 “pdpu” # Swaps case of each character in the string -- returns a new string
  + uniname.isalpha() 🡨 True -- Only letters (a-z, A-Z)
  + uniname.isdigit() 🡨 False -- string of only digits
  + uniname.alnum() 🡨 True
  + uniname.islower() 🡨 False
  + uniname.isupper() 🡨 True
  + uniname.startswith(‘P’) 🡨 True
  + uniname.endswith(‘a’) 🡨 False
  + uniname.count(‘P’) 🡨 2 -- returns int
  + uniname.find(‘O’) 🡨 -1
  + uniname.find(‘U’) 🡨 3
  + uniname.find(‘U’,1,2) 🡨 -1
    - find(...)
    - S.find(sub[, start[, end]]) -> int
    - Return the lowest index in S where substring sub is found,
    - such that sub is contained within S[start:end].
    - Optional arguments start and end are interpreted as in slice notation.
    - Returns -1 on failure.
  + uniname.replace(‘U’,”Univ”) 🡨 “PDPUniv” -- returns a new string
  + uniname.lstrip(), uniname.rstrip(), uniname.strip() 🡨 removes whitespace from left, right or from left and right respectively. -- returns a new string
  + split():
    - sname, fname, surname = input(“Enter your full name :”).split()
    - print(sname,fname, surname)
    - sname, fname, surname = input(“Enter your full name with comma seperation:”).split(‘,’)

--splits the string on comma

* + - p,r,n = input("Enter Principal, rate and no. of years").split()
    - p = int(p)
    - r = float(r)
    - n = float(n)
    - print("Interest = ", p \* r \* n / 100)
  + "-".join("PDEU") 🡨 “P-D-E-U” -- returns a new string
* **Built-in functions for conversion from string to numbers and vice-versa:**
  + str() 🡪 converts an int, float or complex to string
  + int()🡪 converts a numeric string to int
  + float() 🡪 converts a numeric string to float
  + complex() 🡪 converts a numeric string to complex
* chr() returns a string representing its Unicode value. E.g. chr(65) – ‘A’
* ord() returns Unicode value of a character. E.g. ord(‘a’) – 97
* **String Comparison**
  + Comparison is done in lexicographical order, character by character. Result would be either True or False.
  + Use ==, < , > , <=, >= and != operators.

Some Examples:

def printstr(str1):

# print the string character by character in one line.

for ch in str1:

print(ch, end=' ')

print()

# print the string character by character in different lines.

i = 0

while i < len(str1):

print(str1[i])

i = i + 1

str1 = “pandit deendayal energy university”

printstr(str1)

Practice Work:

1. Count how many vowels are there in a string. Accept the string from the user.
2. Write your own functions (without using built-in functions) to convert all characters of a string into lower case/upper case/toggle case.
3. Accept two strings. Check whether one string is there in another string.
4. Write a function that removes one string from another string, if present. E.g. Onestring = “abcdef”, removestring = “cd”. The finalstring should contain “abef”.