CMPE 103

OBJECT-ORIENTED PROGRAMMING

Module 1

PYTHON'S STRING MANIPULATION

STRINGS

? Like many other popular programming languages, strings in Python are arrays of bytes representing Unicode characters. However, ***Python does not have a character data type, a single character is simply a string with a length of 1. Square brackets can be used to access elements of the string***.

HOW TO CHANGE OR DELETE A STRING?

- ***Strings are immutable***. This means that elements of a string cannot be changed once it has been assigned. We can simply reassign different strings to the same name.

```
>>> my_strin = ‘CPE’
>>> my_strin[5] = 'a'
```

- ***TypeError: 'str' object does not support item assignment.***

HOW TO CREATE A STRING IN PYTHON?

? ***How to create a string in Python?*** Strings can be ***created by enclosing characters inside a single quote or double quotes.*** Even triple quotes can be used in Python but generally used to represent multiline strings and docstrings.

REPRESENTATION OF STRING

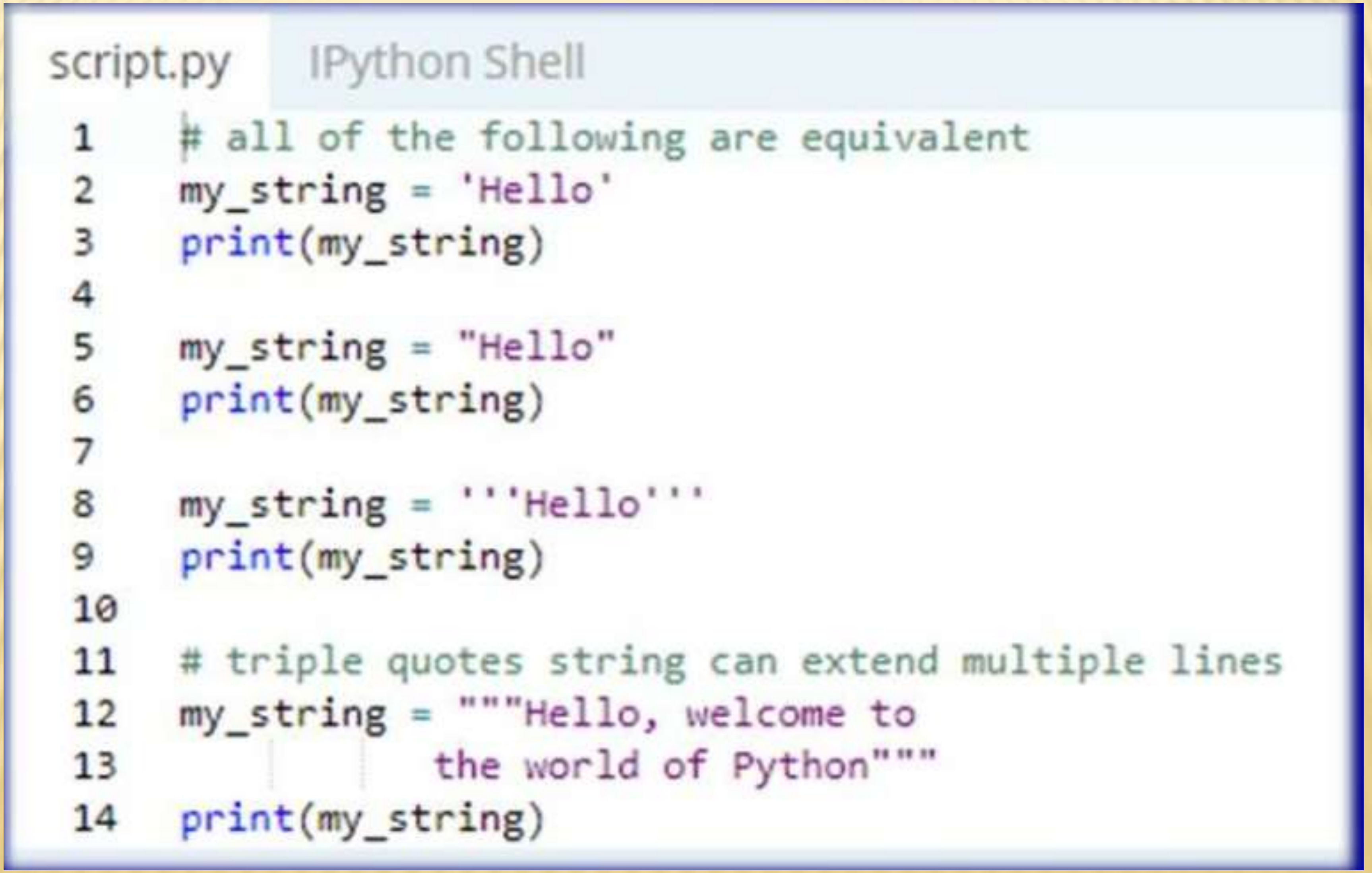
? >>> s = “Hello Python” This is how Python would index the string:

Backward Indexing

-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
H	e	l	l	o		P	y	t	h	o	n
0	1	2	3	4	5	6	7	8	9	10	11

Forward Indexing

PROGRAMMING EXAMPLE



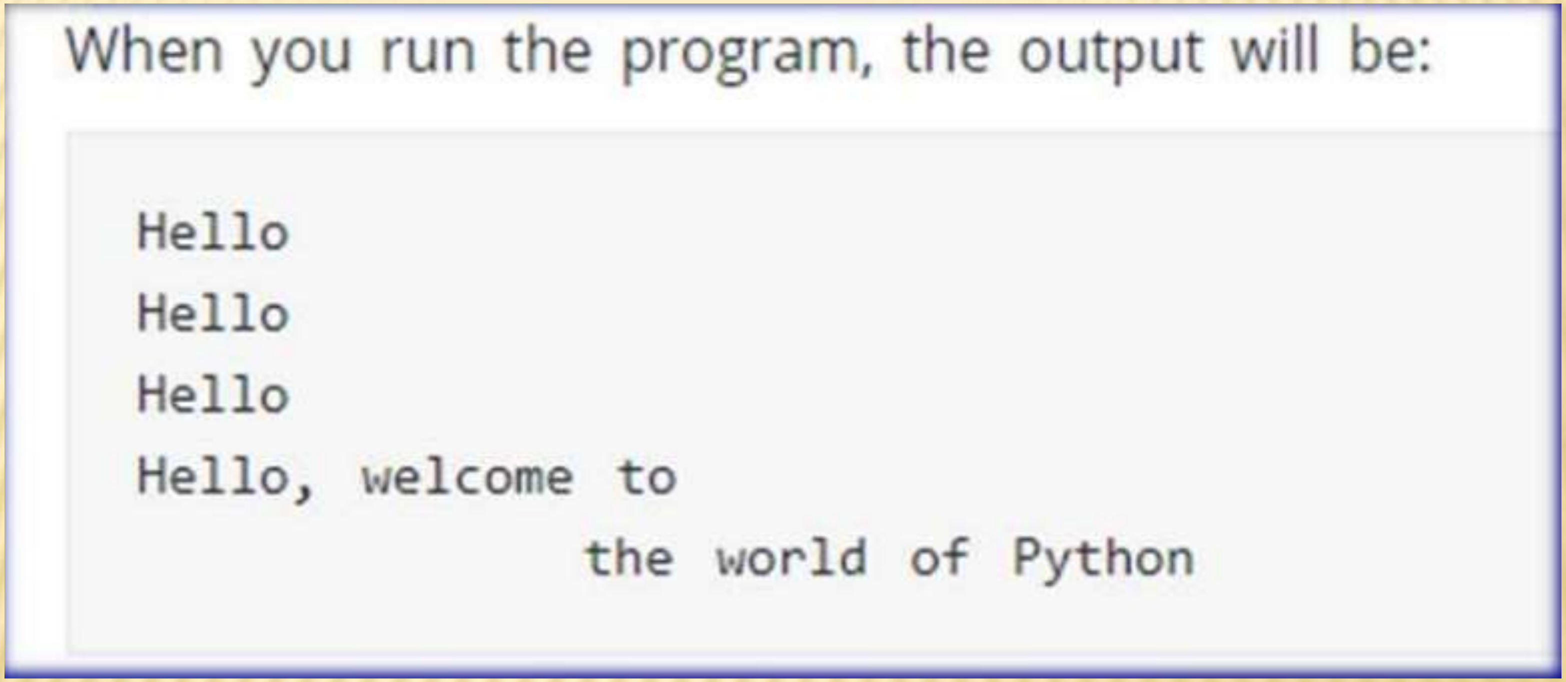
```python
# all of the following are equivalent
my_string = 'Hello'
print(my_string)

my_string = "Hello"
print(my_string)

my_string = '''Hello'''
print(my_string)

# triple quotes string can extend multiple lines
my_string = """Hello, welcome to
           the world of Python"""
print(my_string)
```

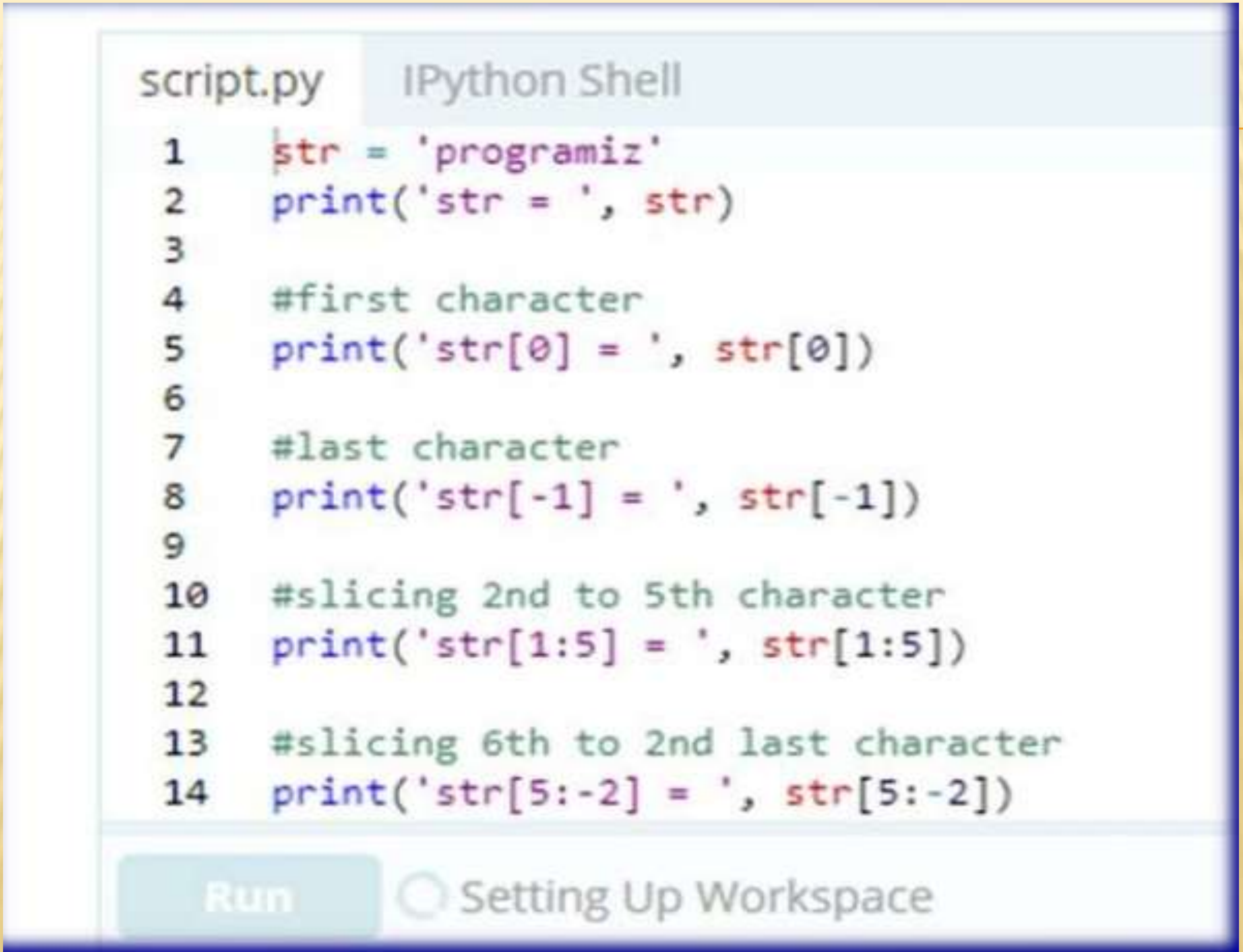
OUTPUT:


When you run the program, the output will be:

```
Hello
Hello
Hello
Hello, welcome to
                   the world of Python
```

HOW TO ACCESS CHARACTERS IN STRING

- We can ***access individual characters using indexing and a range of characters using slicing. Index starts from 0.*** Trying to access a character out of index range will raise an ***IndexError.*** **The index must be an integer. We can't use float or other types, this will result into TypeError.**
- ***Python allows negative indexing for its sequences. The index of -1 refers to the last item,*** -2 to the second last item and so on. We can access a range of items in a string by using the slicing operator (colon).



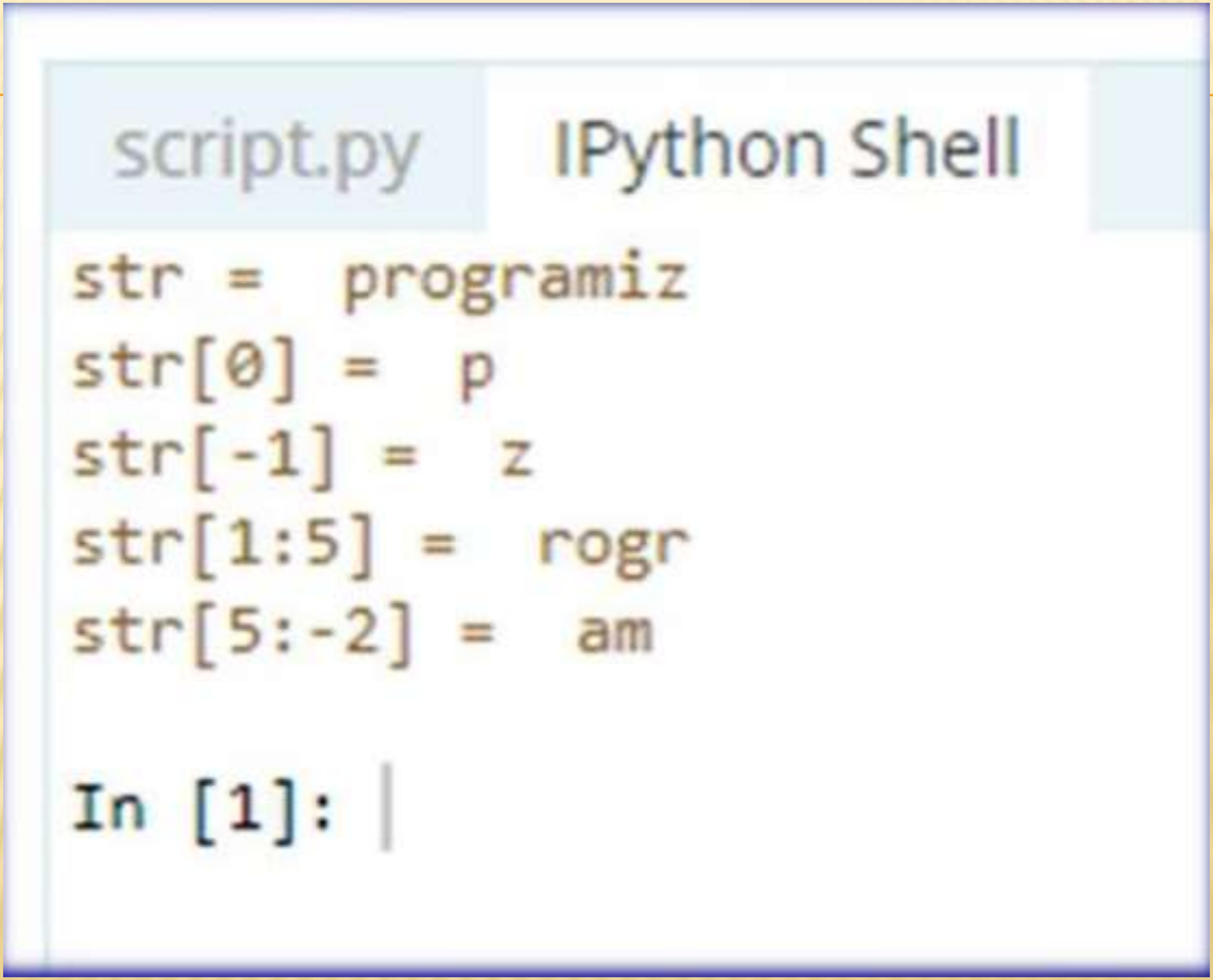
```python
str = 'programiz'
print('str = ', str)

#first character
print('str[0] = ', str[0])

#last character
print('str[-1] = ', str[-1])

#slicing 2nd to 5th character
print('str[1:5] = ', str[1:5])

#slicing 6th to 2nd last character
print('str[5:-2] = ', str[5:-2])
```



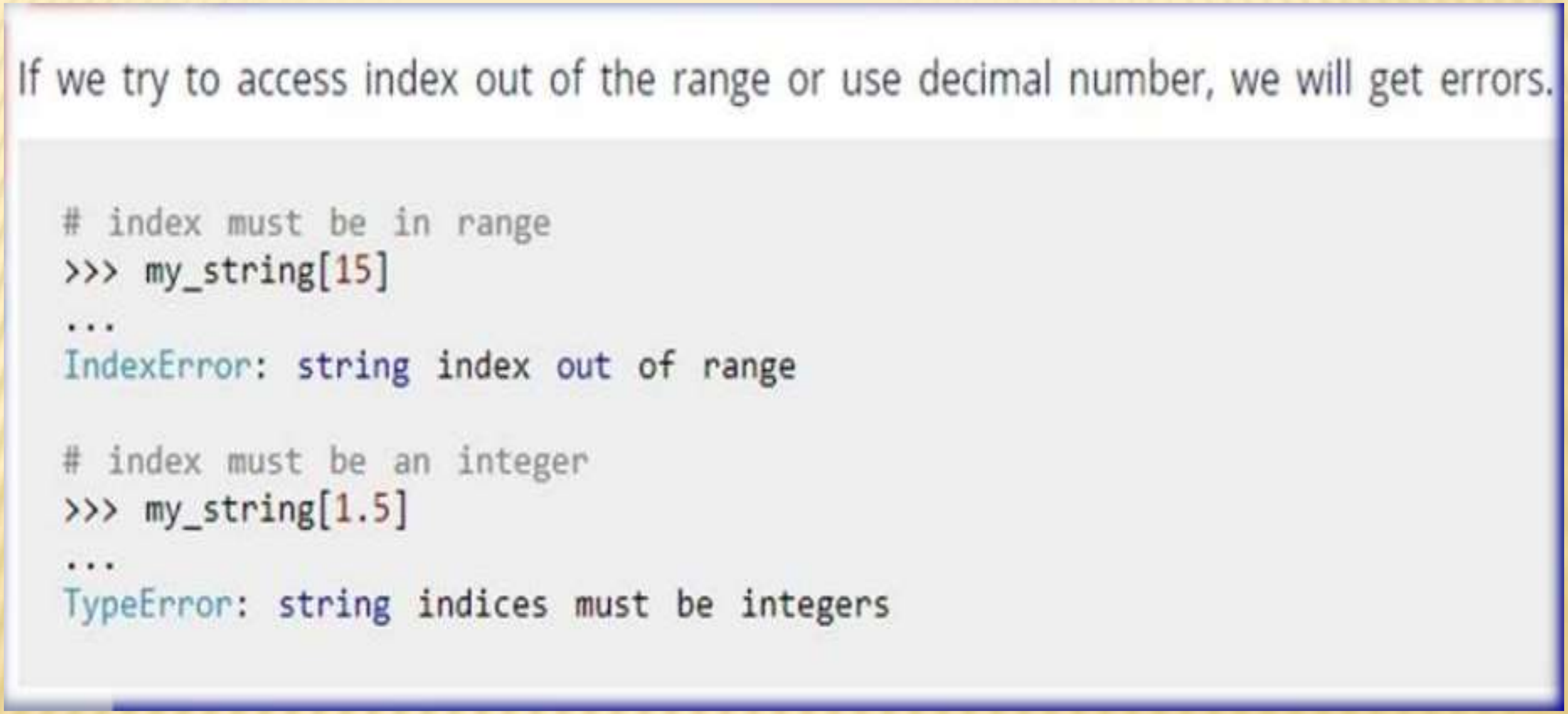
```
str =  programiz
str[0] =  p
str[-1] =  z
str[1:5] =  rogr
str[5:-2] =  am


In [1]:
```

SLICING STRINGS EXAMPLES

- For example:
- >>>“Program”[3:5] will result in: ‘gr ’
  >>>“Program”[3:6] will yield: ‘gra’
- >>>p = “Program”
  >>>p [:4] ‘Prog’
- >>>p = “Program”
  >>>p [4:] ‘ram’
- >>>p = “Program”
  >>>p [3:6] ‘gra’

STRINGS –INDEX ERROR



If we try to access index out of the range or use decimal number, we will get errors.

```
# index must be in range
>>> my_string[15]
...
IndexError: string index out of range

# index must be an integer
>>> my_string[1.5]
...
TypeError: string indices must be integers
```

MORE FUNCTIONALITY OF STRING

? **Finding Length of string**

? >>> len(“Computer Engineering”)

? **String Concatenation**

? >>>print(“CMPE” + “103”)

String Repeat

>>>print(“A” * 4) AAAA

Substring Tests

>>>”C” in “Computer” True

>>>”pr” in “computer” False

>>>”pr” not in “computer” True

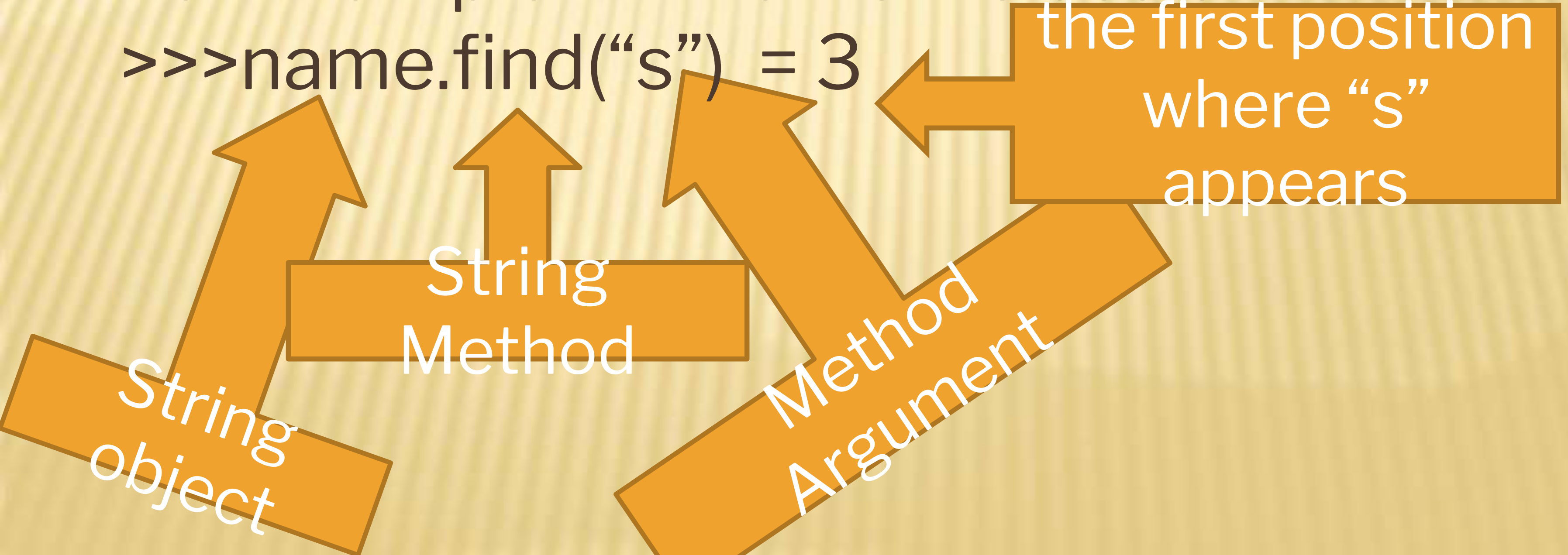
MORE FUNCTIONALITY OF STRING

```
? >>> name1="computer"
? >>> name2=name1[3:5]
? >>>name2
      pu
```

STRING METHODS

- String Methods In Python, a method is a function that is defined with respect to a particular object.
- Syntax: object.method(arguments)
- For Example: >>>name="Classic"

>>>name.find("s") = 3



1. CAPITALIZE() METHOD

- Capitalizes first letter of string
- >>>name="computer"
- >>>name.capitalize()

  'Computer'

2. LSTRIP() & 3. RSTRIP() METHODS

- lstrip() method is used to remove left padded spaces in a given string

  ```
  >>>name1=“ a “
  >>>name1.lstrip()
   ‘a ‘
  ```

- rstrip() method is used to remove right padded spaces in a given string

  ```
  >>>name1.rstrip()
  ‘ a’
  ```

  Removing left spaces Removing right spaces

4. STRIP() METHOD

? strip() method is used to remove left and right padded spaces in a given string

```
>>>name1=“ a “
>>>name1.strip()
    ‘a‘
```

Removing left and right spaces for a given string

5. LOWER() METHOD

- lower() method is used to convert given string in to lower case.

```
>>>name1=“ COMPUTER“
>>>name1.lower()
     computer
```

6. UPPER() METHOD

? upper() method is used to convert given string in to upper case.

```
>>>name1=" computer"
>>>name1.upper()
    COMPUTER
```

7. TITLE() METHOD

? title() method is used to convert given string in to title case. Every first character of word of a given string is converted to title case. >>>name1=“ cpe python syllabus“ >>>name1.title()

Cpe Python Syllabus

8. SWAPCASE() METHOD

- swapcase() method is toggle the case. Meaining upper to lower and lower to upper case. >>>name1=“ Computer “ >>>name1.swapcase()

  cOMPUTER
- Every character case is changed

9. LJUST() METHOD

? ljust() method is used to add spaces to the left side of the given string

```
>>>name1="anand "
>>>name1.ljust(15)
    'anand          '
```

Left side padded with spaces

Note: string length is 5 and 10 spaces added to the left side of string

10. RJUST() METHOD

- rjust() method is used to add spaces to the left side of the given string

```
>>>name1="anand "
>>>name1.rjust(15)
      '          anand'
```

Left side padded with spaces

Note: string length is 5 and 10 spaces added to the left side of string

11. CENTER(WIDTH, FILLCHAR) METHOD

- The method center() returns centered in a string of length width. Padding is done using the specified fillchar. Default filler is a space. Centered string
- >>>name="Anand"
- >>>name.center(36,"a")
- aaaaaaaaaaaaaaaaAnandaaaaaaaaaaaaaaaaa
- >>>name.center(20,"*")
- *******Anand********

12. ZFILL() METHOD

- zfill() method is used to fill the zero to a given string
- >>>name1=“ 123“
- >>>name1.zfill(5)

  ‘00123 ’

  Filling Zeros

13. FIND() METHOD

- find() method is used to find a perticular character or string in a given string.
- >>> name1=“Internet"
- >>> name1.find("e")
- 3
- e is present at 3rd location (first appearance) in a given string

14. COUNT() METHOD

? count() method is used to the number of times character or string appears in a given string. >>>name1=“Internet “

>>>name1.count(“n”)

2

2 times n appears in a given string

15. STARTSWITH() METHOD

- startswith() method is used check string start with particular string or not

```
>>>name1=“Delhi“
>>>name1.startswith(“a”)
    False
```

- Given string not starting with “a”

16. ENDSWITH() METHOD

? endswith() method is used check string ends with particular string or not

```
>>>name1="Dairy"
>>>name1.endswith("ry")
True
```

Given string ends with "en"

17. ISDIGIT() METHOD

- isdigit() method is used check string is digit (number) or not and returns Boolean value true or false.

```
>>>name2="123"
>>>name2.isdigit()
     True
>>name1="123keyboard"
>>name1.isdigit()
     False
```

Given string not number so false

18. ISNUMERIC() METHOD

? isnumeric() is similar to isdigit() method and is used check string is digit (number) or not and returns Boolean value true or false.

```
>>>name2="123"
>>>name2.isnumeric()
True
>>name1="123keyboard"
>>name1.isnumeric()
False
```

Given string not number so false

19. ISDECIMAL() METHOD

? isnumeric(),isdigit() and isdecimal() methods are used to check string is digit (number) or not and returns Boolean value true or false. >>>name2=“123”

>>>name2.decimal()

True

>>name1=“123keyboard“

>>name1.isnumeric()

False

Given string not number so false

20. ISALPHA() METHOD

- isalpha() method is used check string is digit or not and returns Boolean value true or false.

-
  ```
  >>>name2="123"
  >>>name2.isalpha()
  False  (Given string does not contain string  )
  >>name1="123computer"
  >>name1.isalpha()
  False   (Given string is not a string it contains digits)
  >>>name3="Keyboard"
  >>>Name3.isalpha()
        True   (It's a string )
  ```

21. ISALNUM() METHOD

- isalnum() method is used check string is alpha numeric string or not.

```
>>>name2="123"
>>>name2.isalnum()
True   (True Given string is alpha numeric)
>>>name1="123computer"
>>>name1.isalnum()
True   (True Given string is alpha numeric )
>>>name3="Praveen"
>>>name3.isalnum()
True (Given string is alpha numeric )
```

22. ISLOWER() METHOD

? islower() method is used check string contains all lower case letters or not, it returns true or false result.

```
>>>name2=“Anand”
>>>name2.islower()
False   (Given string is not lower case string)
>>>name1=“anand“
>>>name1.islower()
True (Given string is lower case string )
```

23. ISUPPER() METHOD

? isupper() method is used check string contains all letters upper case or not, it returns true or false result.

```
>>>name2=“Anand”
>>>name2.isupper()
False  (Given string is not upper case string)
>>>name1=“ANAND”
>>>name1.isupper()
True (Given string is upper case string )
```

24. ISSPACE() METHOD

- isspace() method is used check string contains space only or not.
- >>>name2=“ ”
- >>>name2.isspace()
- True (Given string contains space only)
- >>>name1=“Anandalaya Anand “
  >>>name1.isspace()
- False (Given string not containing space only)

25. FIND() METHODS

? find() method is used to find a particular string (substring) in a given string.

```
>>>name="Classic"
>>>name.find("s")
3
```

the first position where "s" appears in the string.

26. STR() METHOD

- str() method is used convert non string data into string type.
- >>>str(576)

  ‘576’ (576 is number converted to string)

27 LEN() METHOD

? len() method is used get a length of string.

```
>>>len(“Naveen”)
6 ( Gives the string length)
```

28 MAX() METHOD

? max() method is used get a max alphabet of string.

```
>>>max(“Praveen”)
v   ( Gives max character )
```

29 MIN() METHOD

? min() method is used get a max alphabet of string.

  >>>min(“Anand”)

  A (Gives min character A because it has ASCII Value 65)

30 SPLIT() METHOD

? split() method is used split a string.

>>>name=“Anandalaya NDDB Campus Anand”

>>>name.split()

[“Anandalaya”,“NDDB”,“ Campus”,“Anand”]

Split in to several words or substrings

30 SPLIT() METHOD

? split() method is used split a string according to delimiter.

  >>>name=“Anandalaya NDDB Campus Anand”

  >>>name.split(“Ca”)

      [“Anandalaya NDDB”,“ mpus Anand”]

  Split in to several words or substrings acording to delimiter.

31. INDEX() METHOD

- Same as find(), but raises an exception if str not found.

```
>>> name="Sainik“
>>> name.index("a",3,5)
ValueError: substring not found
>>> name.index("a",1,5)
1  (Character found, returning the position )
```

32. ORD METHOD

? ord() method is used get a ASCII value for a character.

```
>>ord(“a”)
97
```

(97 is the ASCII value for character ‘a’)

33. CHR() METHOD

? chr() method is used get a character for an ASCII value.

```
>>chr(97)
‘a’
```

(‘a’ ASCII value is 97)