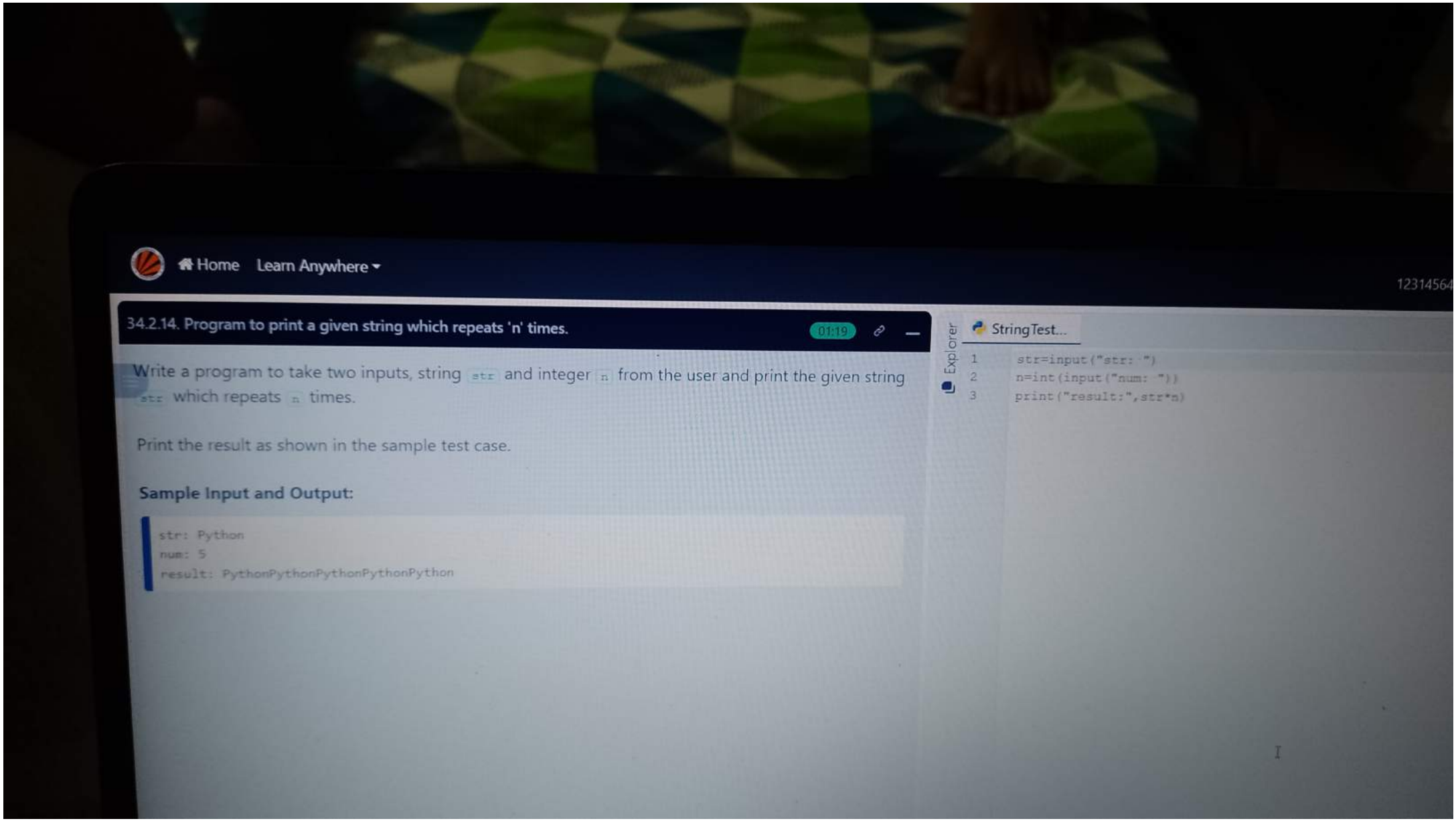
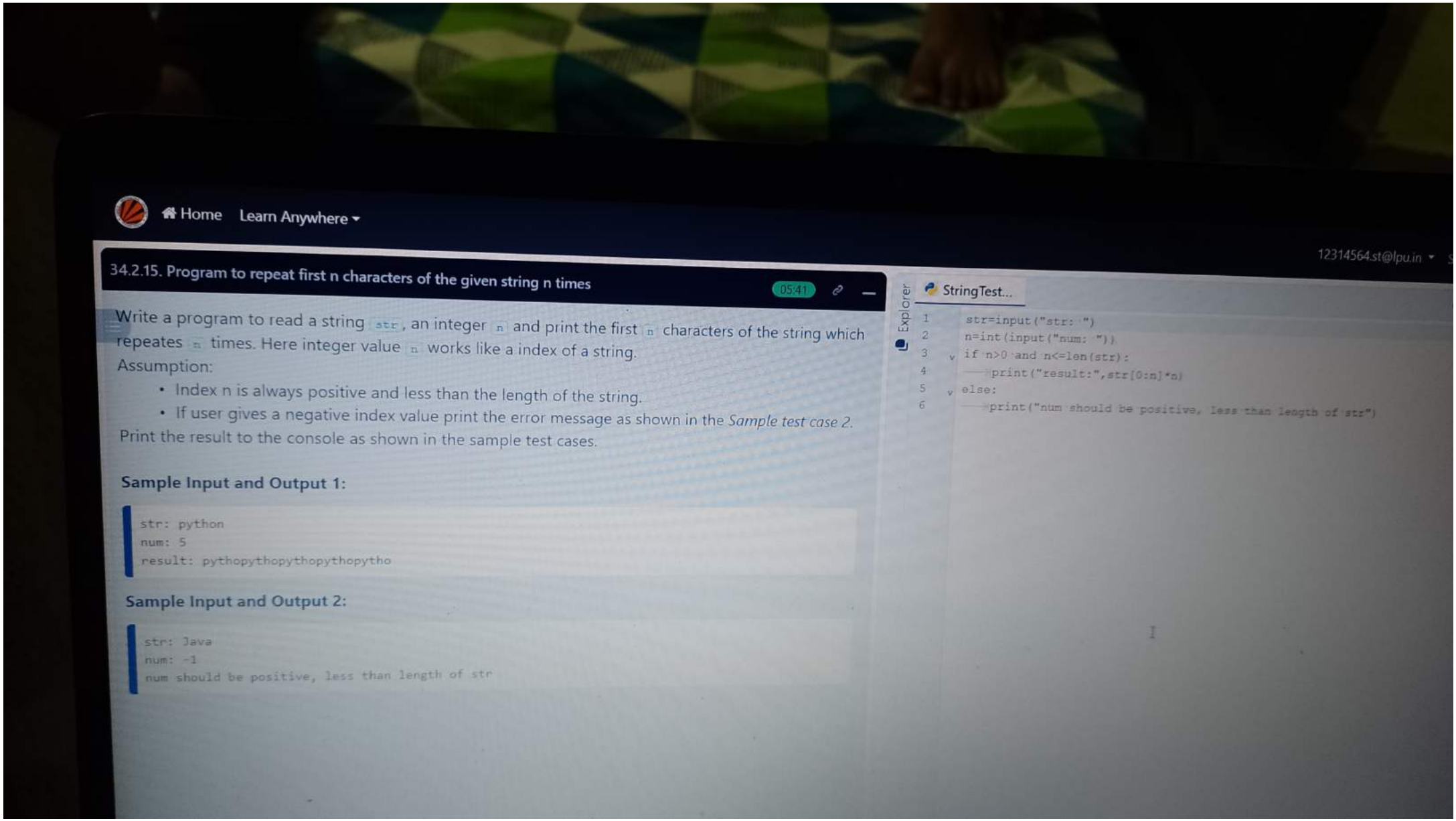
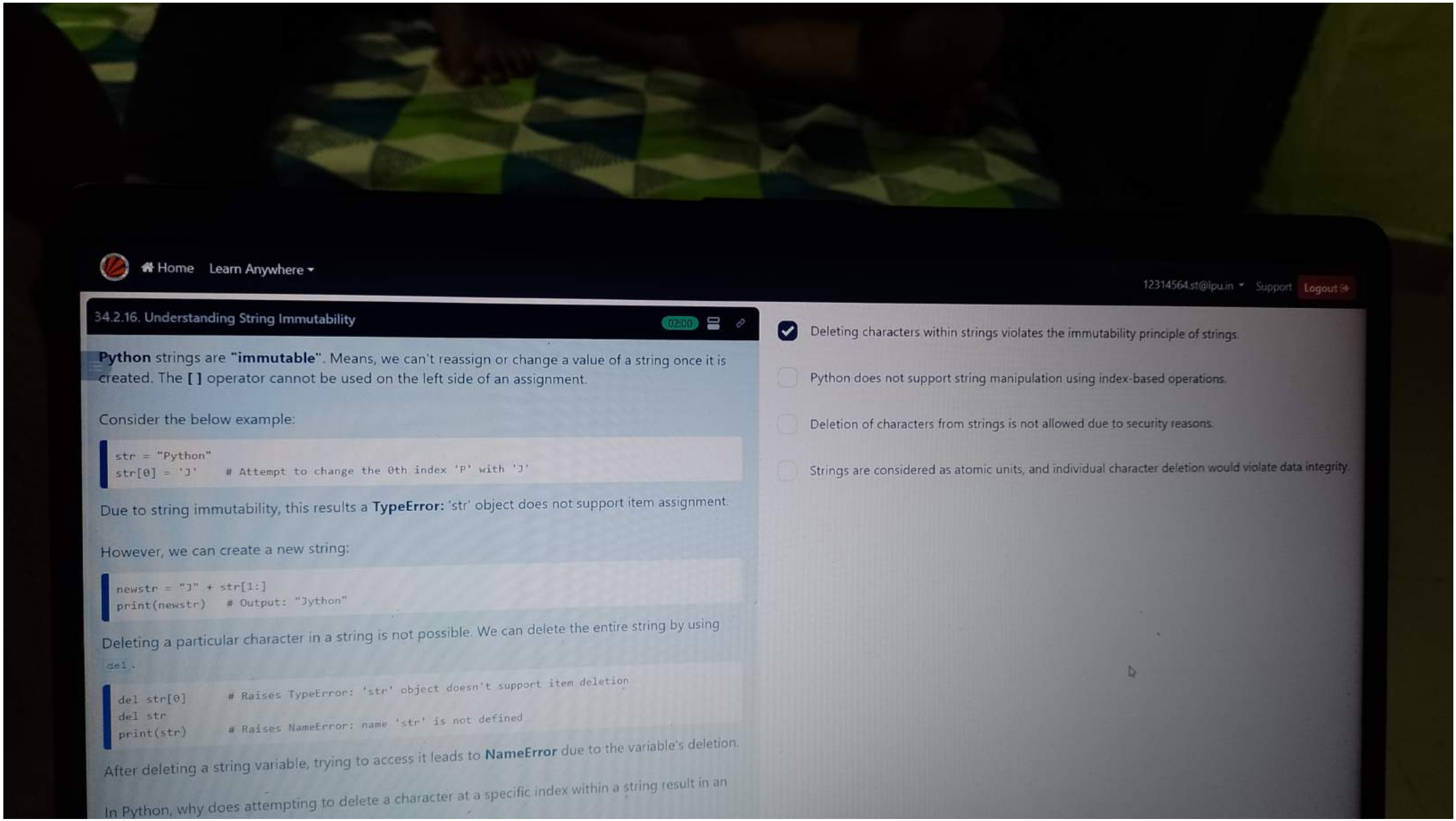

Home
Learn Anywhere
12314564
34.2.14. Program to print a given string which repeats 'n' times.
01:19
Write a program to take two inputs, string str and integer n from the user and print the given string str which repeats n times.
Print the result as shown in the sample test case.
Sample Input and Output:
str: Python
num: 5
result: PythonPythonPythonPythonPython
Explorer
StringTest...
1 str=input("str: ")
2 n=int(input("num: "))
3 print("result:",str*n)


Home
Learn Anywhere
12314564.st@lpu.in
34.2.15. Program to repeat first n characters of the given string n times
05:41
Write a program to read a string str, an integer n and print the first n characters of the string which repeates n times. Here integer value n works like a index of a string.
Assumption:
• Index n is always positive and less than the length of the string.
• If user gives a negative index value print the error message as shown in the Sample test case 2.
Print the result to the console as shown in the sample test cases.
Sample Input and Output 1:
str: python
num: 5
result: pythopythopythopythopytho
Sample Input and Output 2:
str: Java
num: -1
num should be positive, less than length of str
Explorer
StringTest...
1 str=input("str: ")
2 n=int(input("num: "))
3 if n>0 and n<=len(str):
4 print("result:",str[0:n]*n)
5 else:
6 print("num should be positive, less than length of str")


Home
Learn Anywhere
12314564.st@lpu.in
Support
Logout
34.2.16. Understanding String Immutability
02:00
Python strings are "immutable". Means, we can't reassign or change a value of a string once it is created. The [ ] operator cannot be used on the left side of an assignment.
Consider the below example:
str = "Python"
str[0] = 'J' # Attempt to change the 0th index 'P' with 'J'
Due to string immutability, this results a TypeError: 'str' object does not support item assignment.
However, we can create a new string:
newstr = "J" + str[1:]
print(newstr) # Output: "Jython"
Deleting a particular character in a string is not possible. We can delete the entire string by using del .
del str[0] # Raises TypeError: 'str' object doesn't support item deletion
del str
print(str) # Raises NameError: name 'str' is not defined
After deleting a string variable, trying to access it leads to NameError due to the variable's deletion.
In Python, why does attempting to delete a character at a specific index within a string result in an
Deleting characters within strings violates the immutability principle of strings.
Python does not support string manipulation using index-based operations.
Deletion of characters from strings is not allowed due to security reasons.
Strings are considered as atomic units, and individual character deletion would violate data integrity.

Home Learn Anywhere

12314564.st@lpu.in Support Logout

35.1.1. Functions of String Data Type

04:03

Python provides the following built-in **string methods** (operations that can be performed with string objects). **Syntax** to execute these methods is:

```
stringobject.methodname()
```

Given an input string `a = "hello python"`, now understand the working principles of the following methods:

1) **capitalize()** - Capitalizes first letter of a string.

```
print(a.capitalize())    # Result: Hello python
```

2) **upper()** - Converts the string to uppercase.

```
print(a.upper())         # Result: 'HELLO PYTHON'
```

3) **lower()** - Converts the string to uppercase.

```
print(a.lower())         # Result: 'hello python'
```

4) **title()** - Converts the string to title case, i.e., first characters of all the words of string are capitalized.

```
print(a.title())         # Result: 'Hello Python'
```

lowercase into uppercase

- [x] In Python, we can convert strings of Uppercase letters into lower case and lower case to upper case using swapcase() method.
- [ ] print('python is simple'.title()) - Outputs: Python is Simple
- [ ] a = "python is my favourite" a[3] = 'l' , replaces 'l' with 'h'
- [ ] a = "python" + 3.7 - Outputs: python3.7
- [x] print('abcpyxyzpython', 3, 10) - Outputs: abcpyxyzpython 3 10.

Home Learn Anywhere

12314564.st@lpu.in Support Logout

35.1.2. Methods Of Strings - count(), replace(), join()

14:36

Consider an input string `a = "hello happy birthday happy birthday life is happy"`, let's understand the working principles of the following methods:

8) **count(substring)** - Returns the count of occurrences of the **substring** in a string. If the **substring** does not exist, it returns **zero**.

```
print(a.count('happy'))      # Result: 3 (happy word occurred 3 times)
```

9) **replace(old, new)** - Replace all occurrences of **old** substring with **new** substring. If the **old** substring does not exist, no modifications will be done.

```
print(a.replace('happy' ,'joyful'))   # Result: 'hello joyful birthday joyful birthday life is joyf
```

10) **join(iterable)** - Concatenate the elements of the **iterable** using the string as the separator. ("L1" is iterable in the below example.)

```
b = '.'
L1 = ["www", "codetantra", "com"]
print(b.join(L1))     # Concatenates each item in the list with '.' Result: 'www.codetantra.com'
```

Write a program to demonstrate string methods using a user input. Follow the instructions provided in the comment lines.

Explorer

StringEx1.py

```
1   str1 = input("str: ")
2   # Convert the string to uppercase
3   print(str1.upper())
4   # Convert the string to title case
5   print(str1.title())
6   # Split the string into a list of words
7   print(str1.split())
8   width = int(input("width: "))
9   fill_char = input("fillchar: ")
10  # Center align the string with given width and fill character
11  print(str1.center(width,fill_char))
12  # Convert the string to lowercase
13  print(str1.lower())
14  str2 = input("Enter a joining character: ")
15  # Join the characters of the string with the given character
16  l=list(str1)
17  print(str2.join(l))
18  replace_old = input("old substring: ")
19  replace_new = input("new substring: ")
20  # Replace occurrences of old substring with new substring
21  print(str1.replace(replace_old,replace_new))
22
23
24
25
26
27
```

Debugger

Plots

35.1.3. Methods Of Strings - isupper(), islower(), isalpha(), isalnum()

03:57

11) **isupper()** - Checks if all characters in the string are uppercase or not. If yes returns **True**, otherwise **False**.

12) **islower()** - Checks if all characters in the string are lowercase or not. If yes returns **True**, otherwise **False**.

13) **isalpha()** - Checks if the string contains alphabetic characters only or not. If yes returns **True**, otherwise **False**.

- Space is not considered as alphabet character, it will fall in the category of special characters.

14) **isalnum()** - Checks if the string contains alphanumeric characters only or not. If yes returns **True**, otherwise **False**.

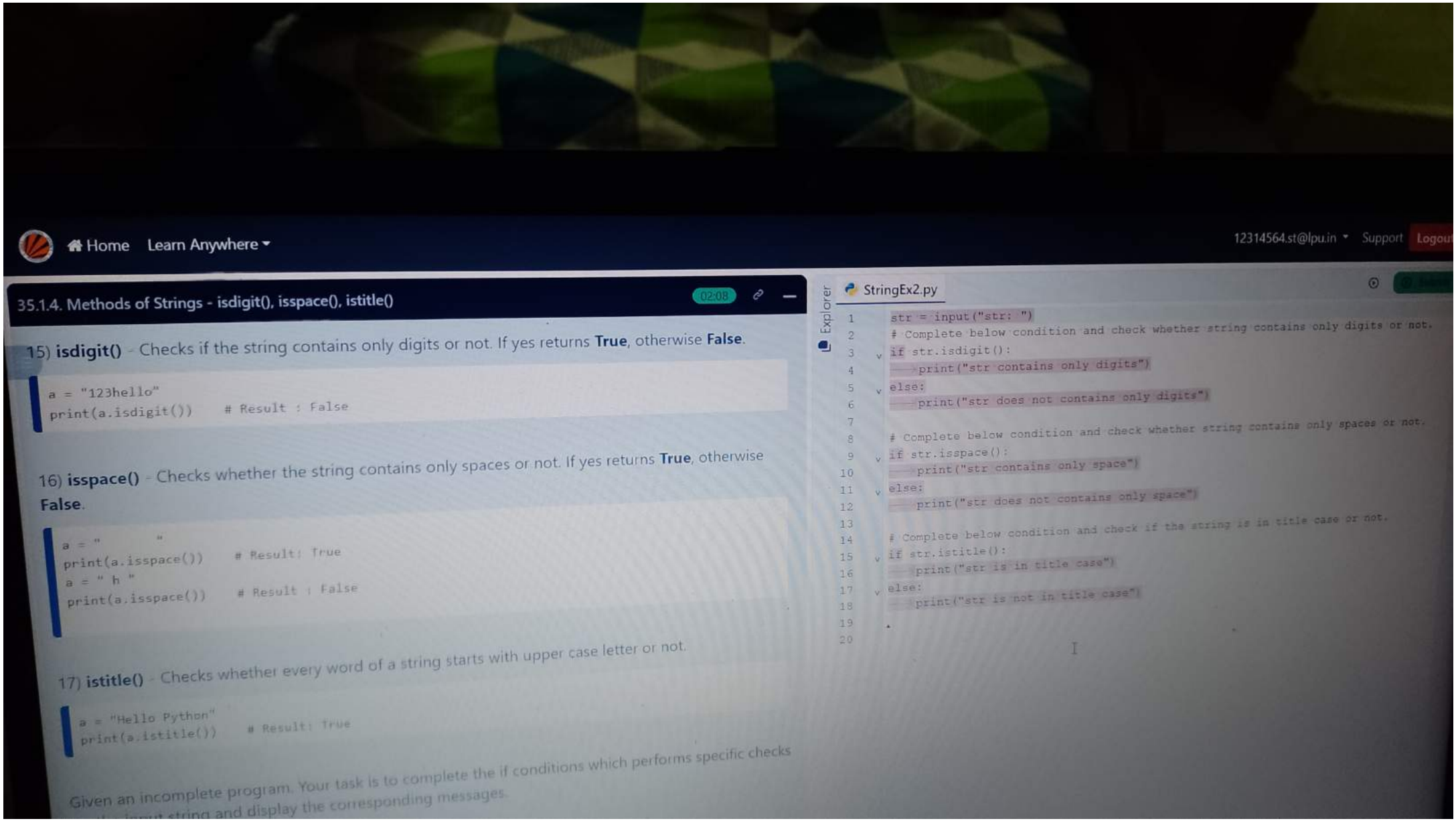
- Characters those are not alphanumeric are: (space) ! # % & ? etc.
- Numerals (0-9), alphabets (A-Z, a-z) will fall into the category of alphanumeric characters.

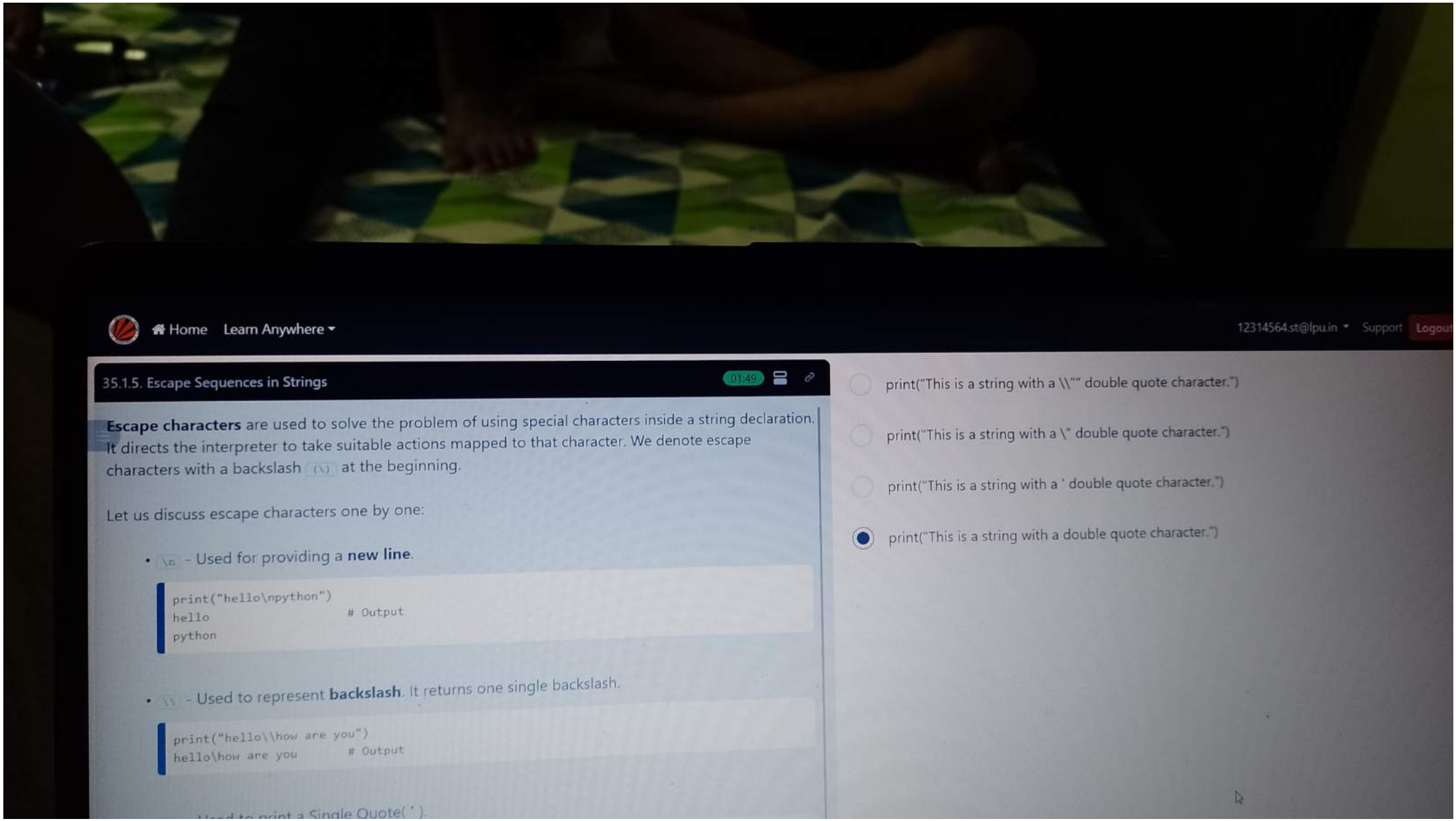
Now, let's understand these methods with small example. Assume there is an string `a = "HELLOWORLD123"` . Observe the output:

```
print(a.isupper())     # Result: True
print(a.islower())     # Result: False
print(a.isalpha())     # Result: False
print(a.isalnum())     # Result: True
```

Select all the correct statements among the provided options.

- [ ] str = "PYTHOn" print(str.isupper()) returns True.
- [x] print("Hello123".isalpha()) returns False.
- [x] str = '123', the below code is correct to convert string into int. print(int(str)) # 123.
- [x] str = "hello world", print(str.isalnum()) returns False.


Home
Learn Anywhere
12314564.st@lpu.in
Support
Logout
35.1.4. Methods of Strings - isdigit(), isspace(), istitle()
02:08
15) isdigit() - Checks if the string contains only digits or not. If yes returns True, otherwise False.
a = "123hello"
print(a.isdigit()) # Result : False
16) isspace() - Checks whether the string contains only spaces or not. If yes returns True, otherwise False.
a = " "
print(a.isspace()) # Result: True
a = " h "
print(a.isspace()) # Result : False
17) istitle() - Checks whether every word of a string starts with upper case letter or not.
a = "Hello Python"
print(a.istitle()) # Result: True
Given an incomplete program. Your task is to complete the if conditions which performs specific checks
string and display the corresponding messages
Explorer
StringEx2.py
str = input("str: ")
# Complete below condition and check whether string contains only digits or not.
if str.isdigit():
    print("str contains only digits")
else:
    print("str does not contains only digits")
# Complete below condition and check whether string contains only spaces or not.
if str.isspace():
    print("str contains only space")
else:
    print("str does not contains only space")
# Complete below condition and check if the string is in title case or not.
if str.istitle():
    print("str is in title case")
else:
    print("str is not in title case")


Home
Learn Anywhere
12314564.st@lpu.in
Support
Logout
35.1.5. Escape Sequences in Strings
01:49
Escape characters are used to solve the problem of using special characters inside a string declaration. It directs the interpreter to take suitable actions mapped to that character. We denote escape characters with a backslash (\) at the beginning.
Let us discuss escape characters one by one:
• \n - Used for providing a new line.
print("hello\npython")
hello # Output
python
• \\ - Used to represent backslash. It returns one single backslash.
print("hello\\how are you")
hello\how are you # Output
print("This is a string with a \\"" double quote character.")
print("This is a string with a \" double quote character.")
print("This is a string with a ' double quote character.")
print("This is a string with a double quote character.")

35.1.6. Escape Sequences

02:35

Escape sequences can sometimes be printed as they are, without being treated as special characters. To achieve this, you can use the `repr()` function or the `r` / `R` prefix.

Example using `repr()` :

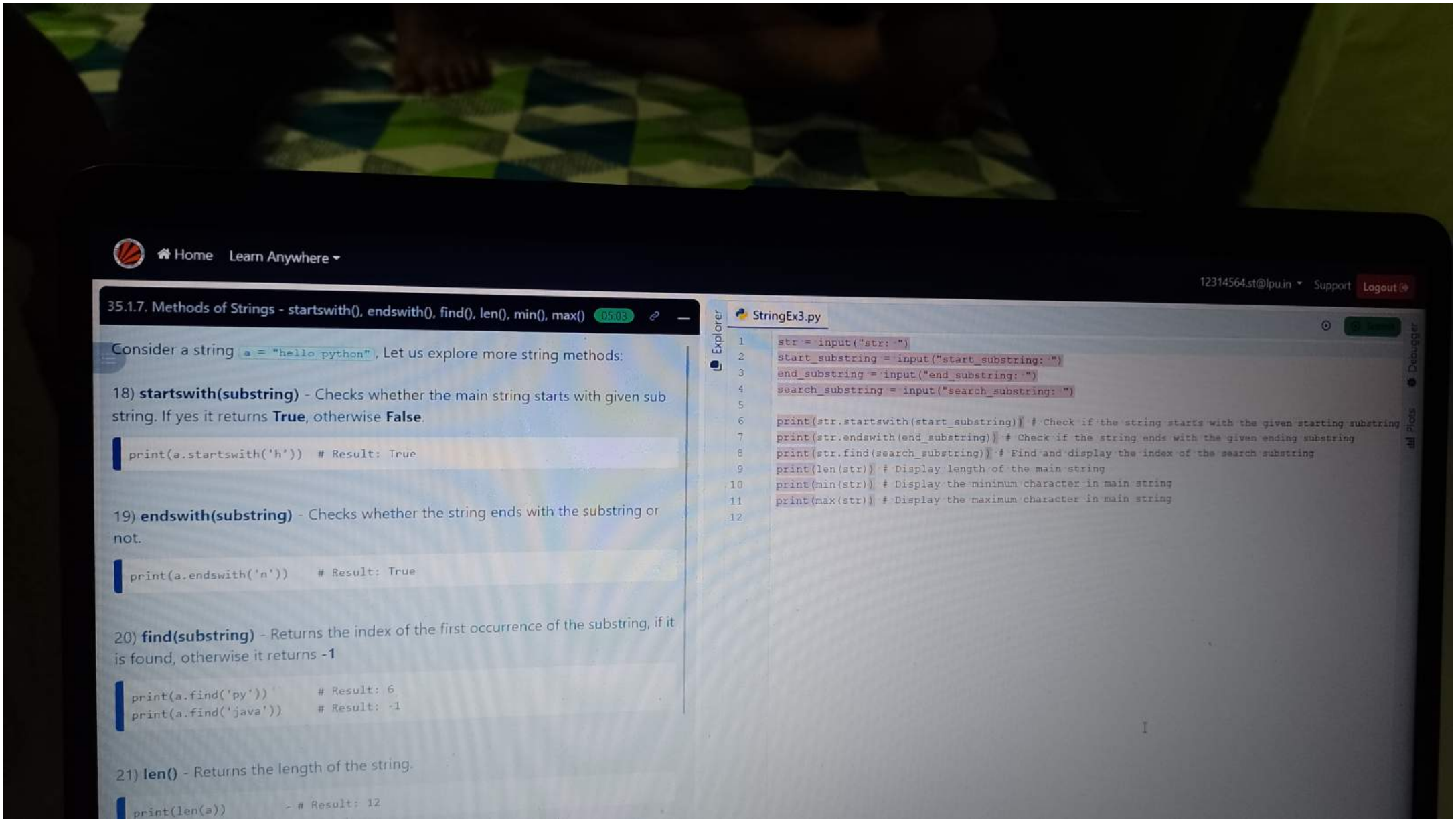
```
str = "Hello\tPython\nPython is very interesting"
print(str)  # will print result as follows
Hello       Python
Python is very interesting

print(repr(str)) # will print result as follows
'Hello\tPython\nPython is very interesting'
```

Example using `'r'` and `'R'` :

```
print(r"Hello\tPython\nPython is very interesting")  # wil
Hello\tPython\nPython is very interesting

print(R"Hello\tPython\nPython is very interesting") # will
Hello\tPython\nPython is very interesting
```

Which of the following statements about escape sequences in

- Escape sequences are always interpreted as special characters, even when using the **repr()** function or the **r / R** prefix.
- Escape sequences are only interpreted as special characters when using the **print()** function.
- **Escape sequences can be printed as they are, without being interpreted as special characters, by using the repr() function or the r / R prefix.**
- The repr() function and the r / R prefix have the same effect on escape sequences.


Home
Learn Anywhere
12314564.st@lpu.in
Support
Logout
35.1.7. Methods of Strings - startswith(), endswith(), find(), len(), min(), max()
05:03
Consider a string a = "hello python", Let us explore more string methods:
18) startswith(substring) - Checks whether the main string starts with given sub string. If yes it returns True, otherwise False.
print(a.startswith('h')) # Result: True
19) endswith(substring) - Checks whether the string ends with the substring or not.
print(a.endswith('n')) # Result: True
20) find(substring) - Returns the index of the first occurrence of the substring, if it is found, otherwise it returns -1
print(a.find('py')) # Result: 6
print(a.find('java')) # Result: -1
21) len() - Returns the length of the string.
print(len(a)) - # Result: 12
Explorer
StringEx3.py
str = input("str: ")
start_substring = input("start_substring: ")
end_substring = input("end_substring: ")
search_substring = input("search_substring: ")
print(str.startswith(start_substring)) # Check if the string starts with the given starting substring
print(str.endswith(end_substring)) # Check if the string ends with the given ending substring
print(str.find(search_substring)) # Find and display the index of the search substring
print(len(str)) # Display length of the main string
print(min(str)) # Display the minimum character in main string
print(max(str)) # Display the maximum character in main string
Debugger
Plots

Home Learn Anywhere

12314564.st@lpu.in Support Logout

35.1.8. Program to enclose the longer string with-in the shorter string and display the result.

08:33

Write a program to take two strings of different lengths as input from the user and enclose the longer string with-in the shorter string.

Print the result as shown in the sample test cases.

Sample Input and Output 1:

```
str1: Big Data
str2: Hadoop
HadoopBig DataHadoop
```

Here, **len(Big Data)** is **8** and **len(Hadoop)** is **6** so **Big Data** is enclosed with **Hadoop**.

Sample Input and Output 2:

```
str1: Django
str2: Django
strings are same length
```

Explorer

StringTest1...

```
1  str1 = input("str1: ")
2  str2 = input("str2: ")
3
4  # Calculate the lengths of the input strings
5  l1 =len(str1)
6  l2 =len(str2)
7
8  # Print strings are equal if the both lengths are equal
9
10 if len(str1)==len(str2):
11     print("strings are same length")
12 # If the length of str1 is shorter than the length of str2, print the concatenation
13 elif len(str1)<len(str2):
14     print(str1+str2+str1)
15
16 # If the length of str1 is longer than the length of str2, print the concatenation
17 else:
18     print(str2+str1+str2)
19
20
```

Debugger

Plots

Home Learn Anywhere

12314564.st@lpu.in Support

35.1.9. Write a program to find whether a given string starts with 'Python' or not.

22:33

Write a program to take string as input from the user and check whether the given input begins with **Python** or not. If yes, print the input string as it is, otherwise add **Python** to input string.

Print the result as shown in the sample test cases.

Sample Input and Output 1:

```
str: Python
str is: Python
```

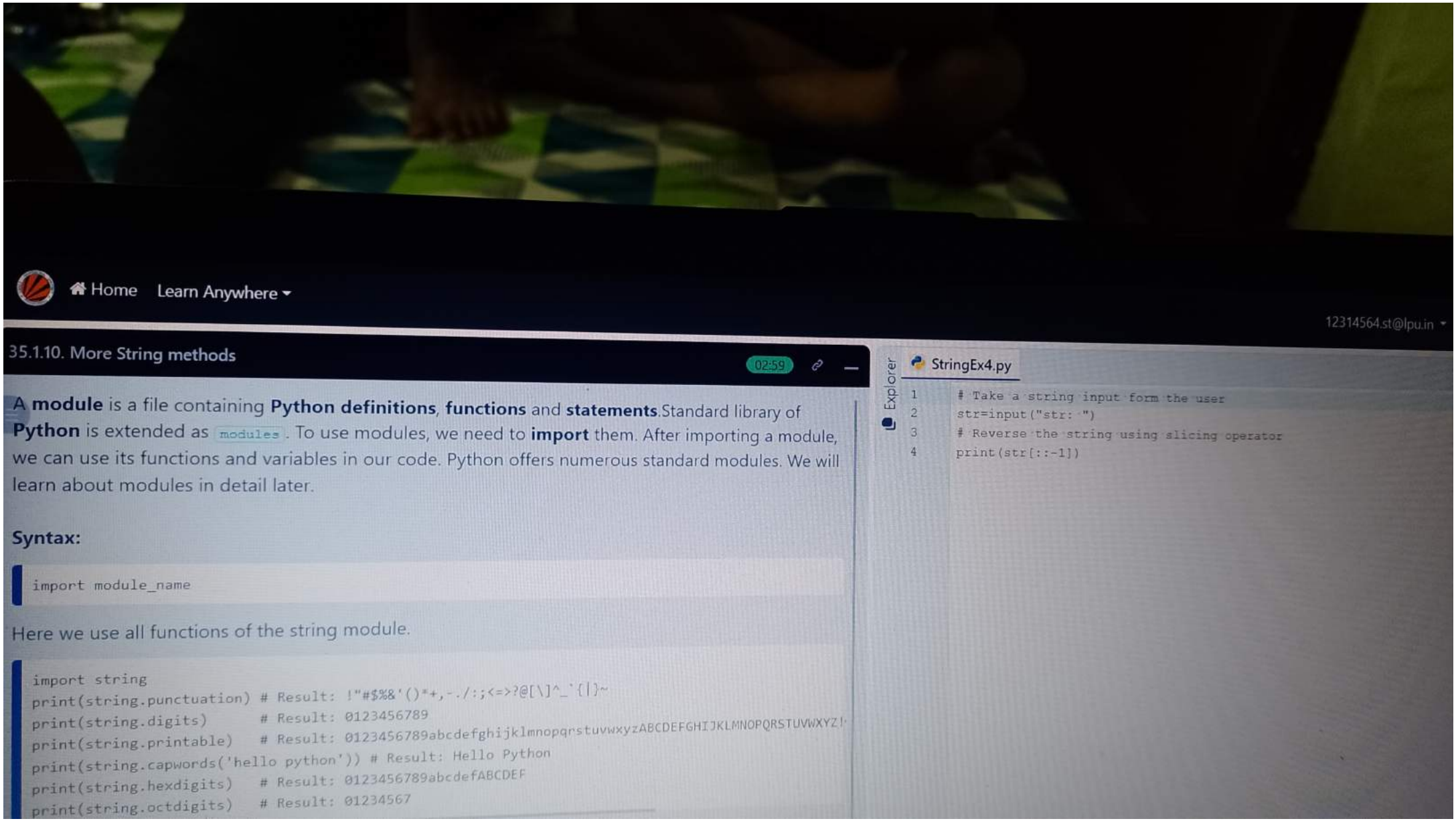
Sample Input and Output 2:

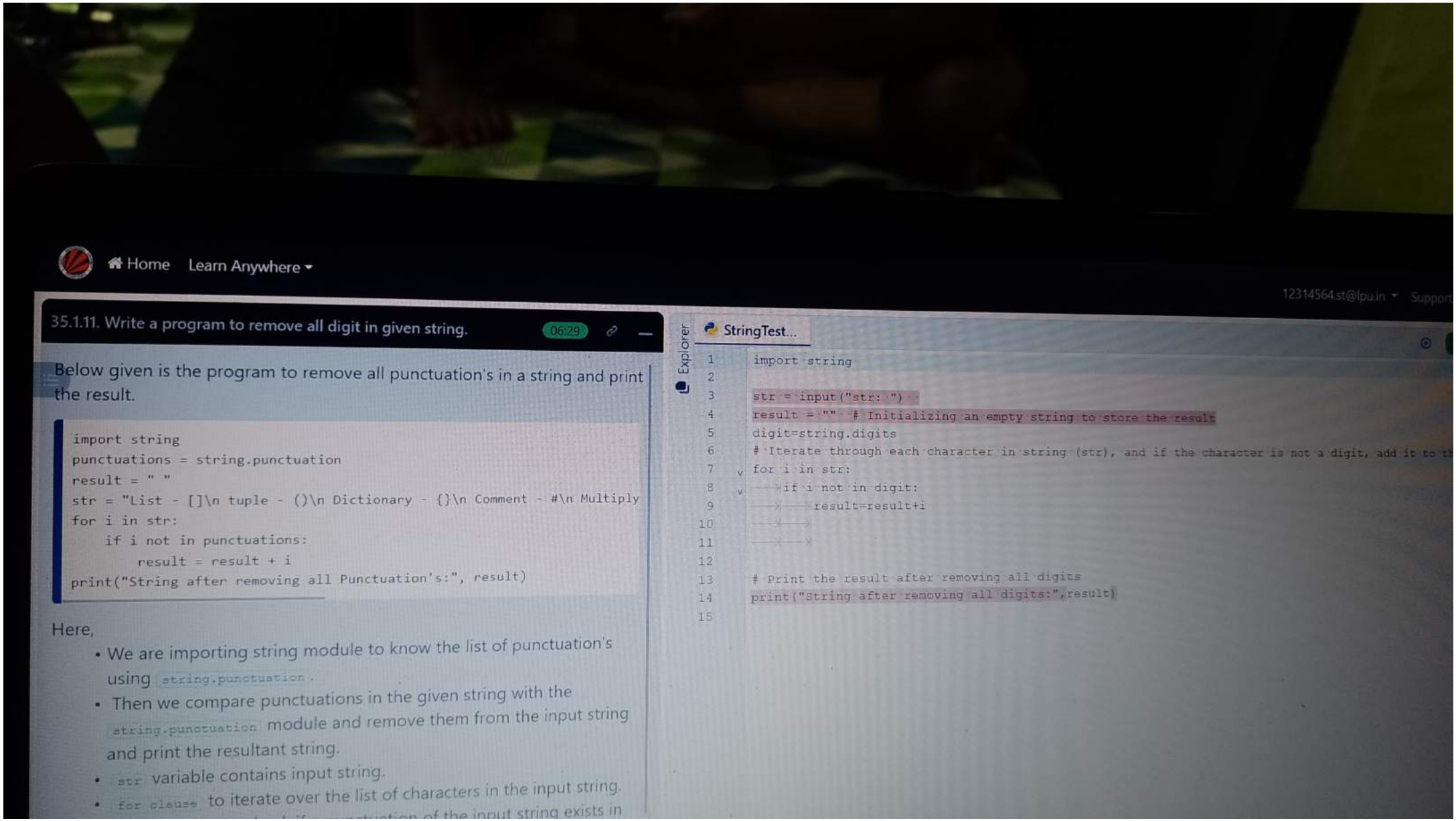
```
str: World
str after adding 'Python': Python World
```

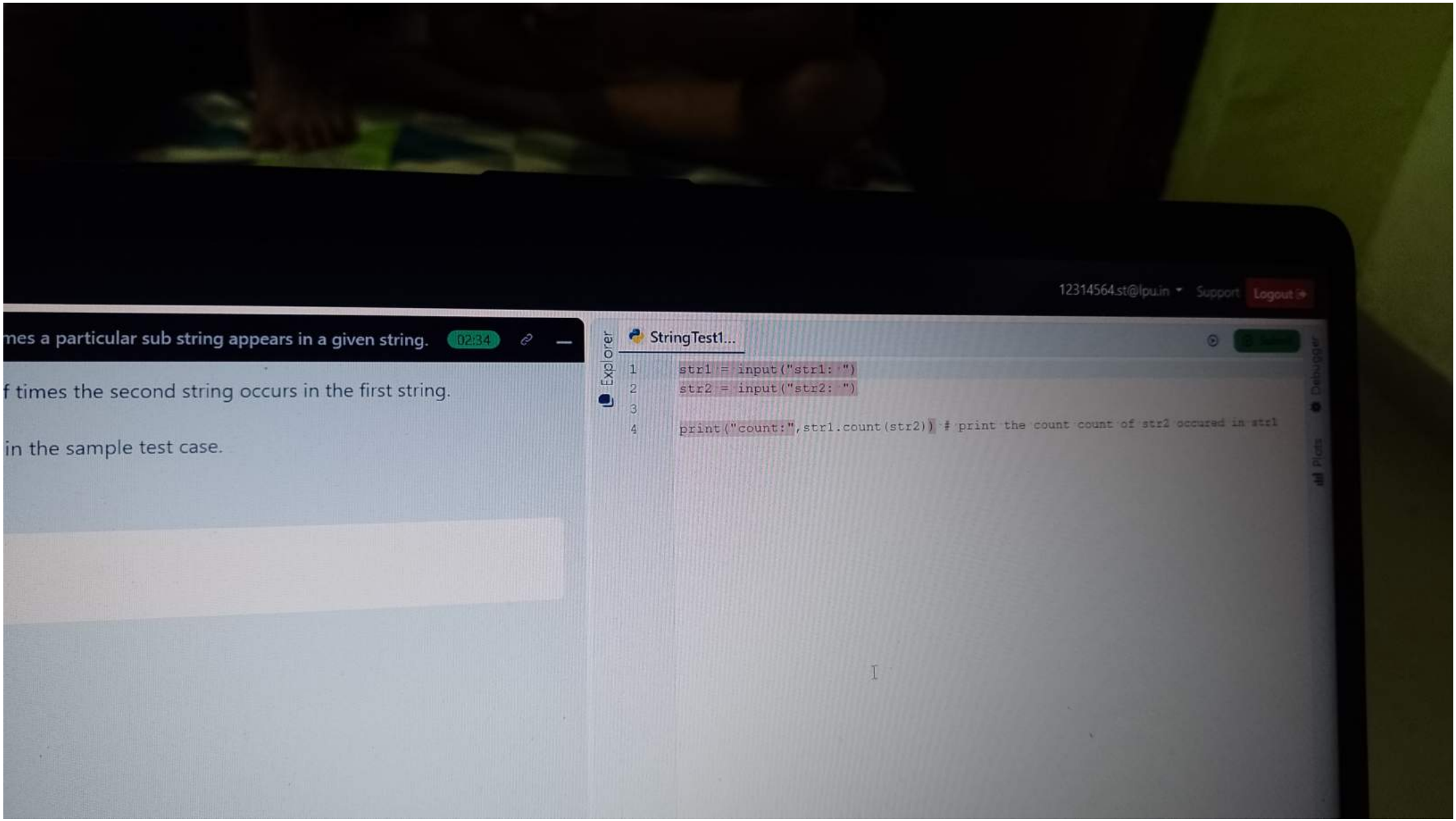
Explorer

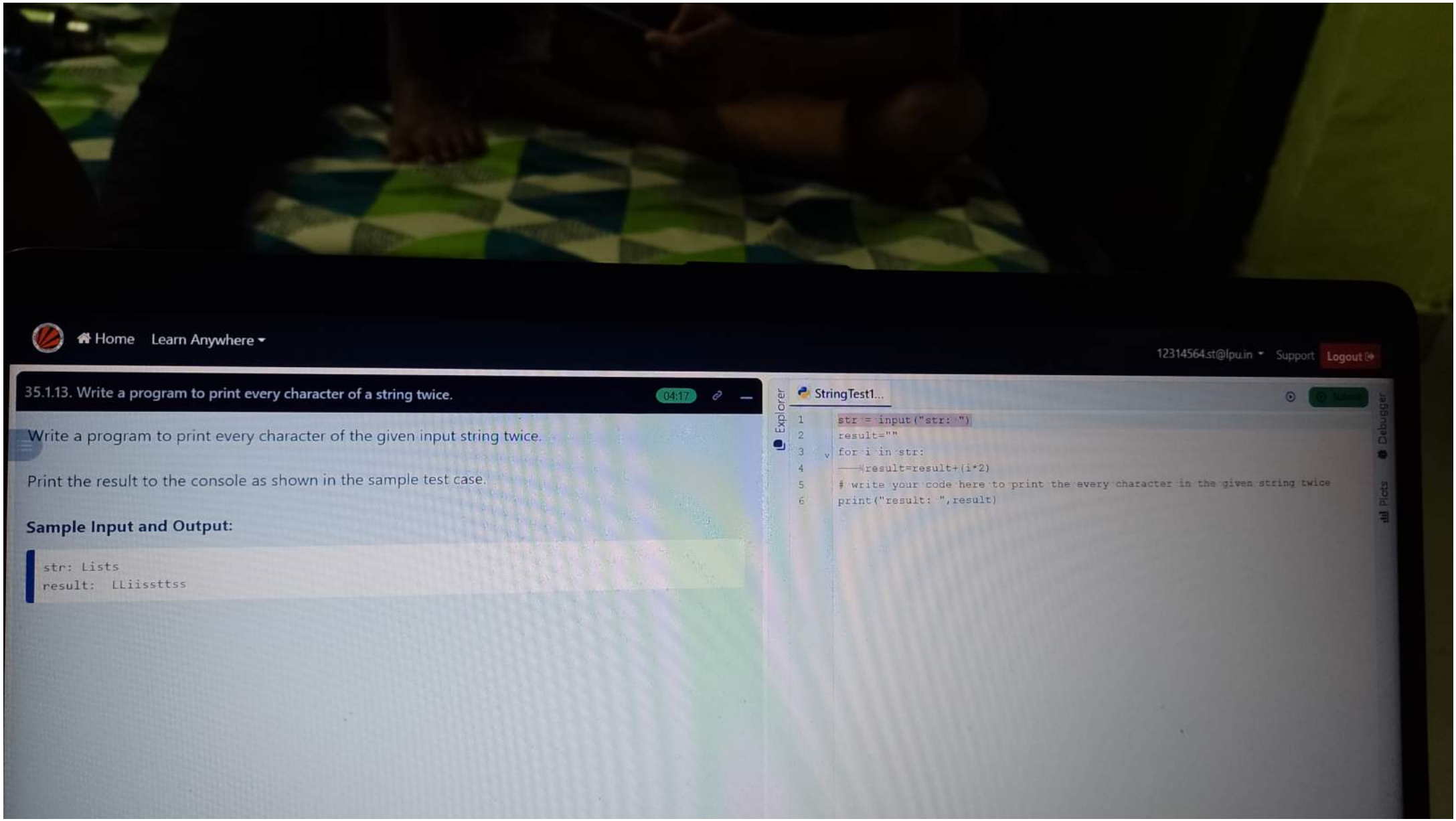
StringTest1...

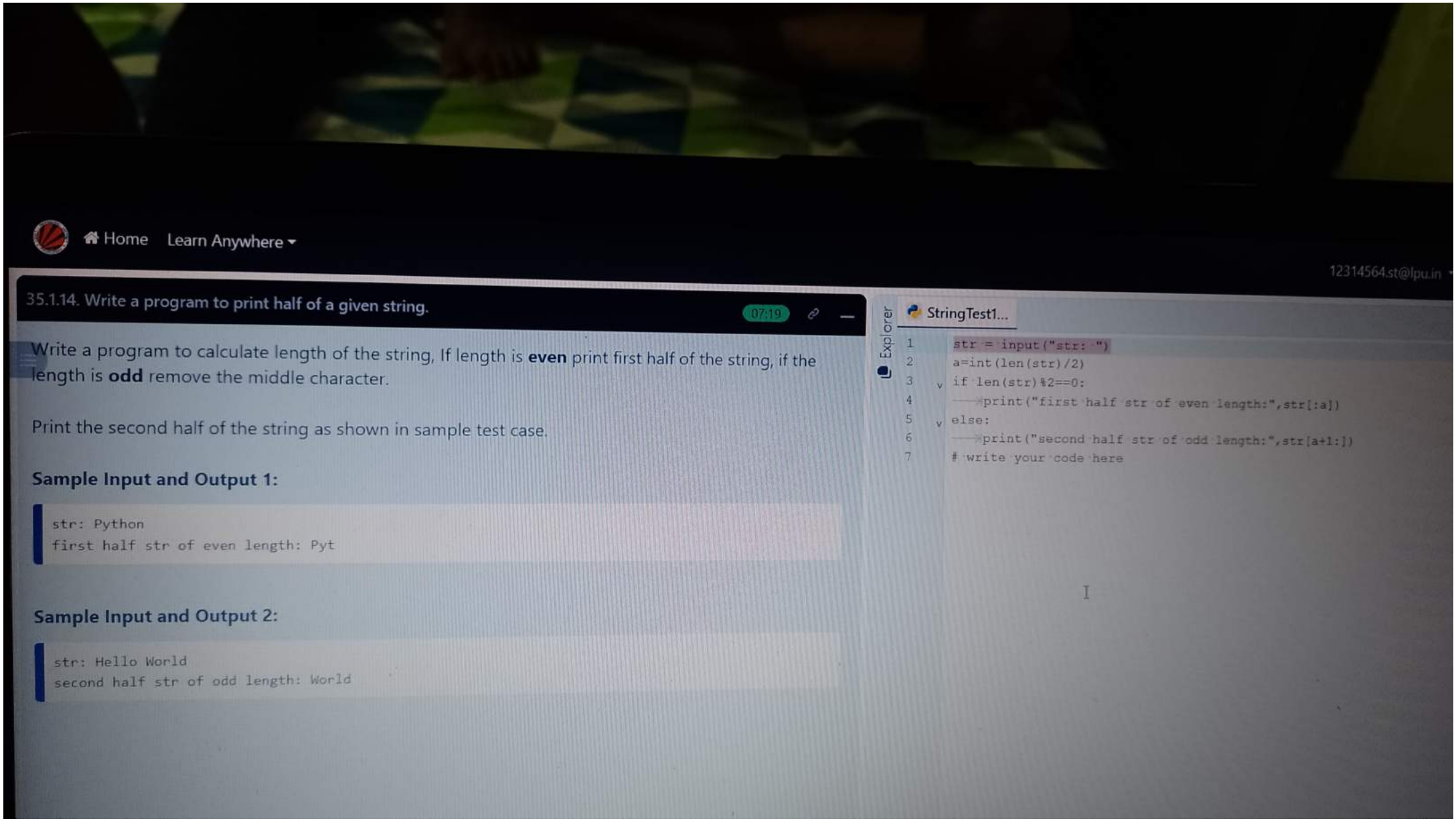
```
# Note: Python and python are different.

str = input("str: ")
# write yur code here
str=input()
if str.startswith("Python"):
    print("str is:",str)
else:
    result="Python "+str
    print("str after adding \'Python\':",result)
```

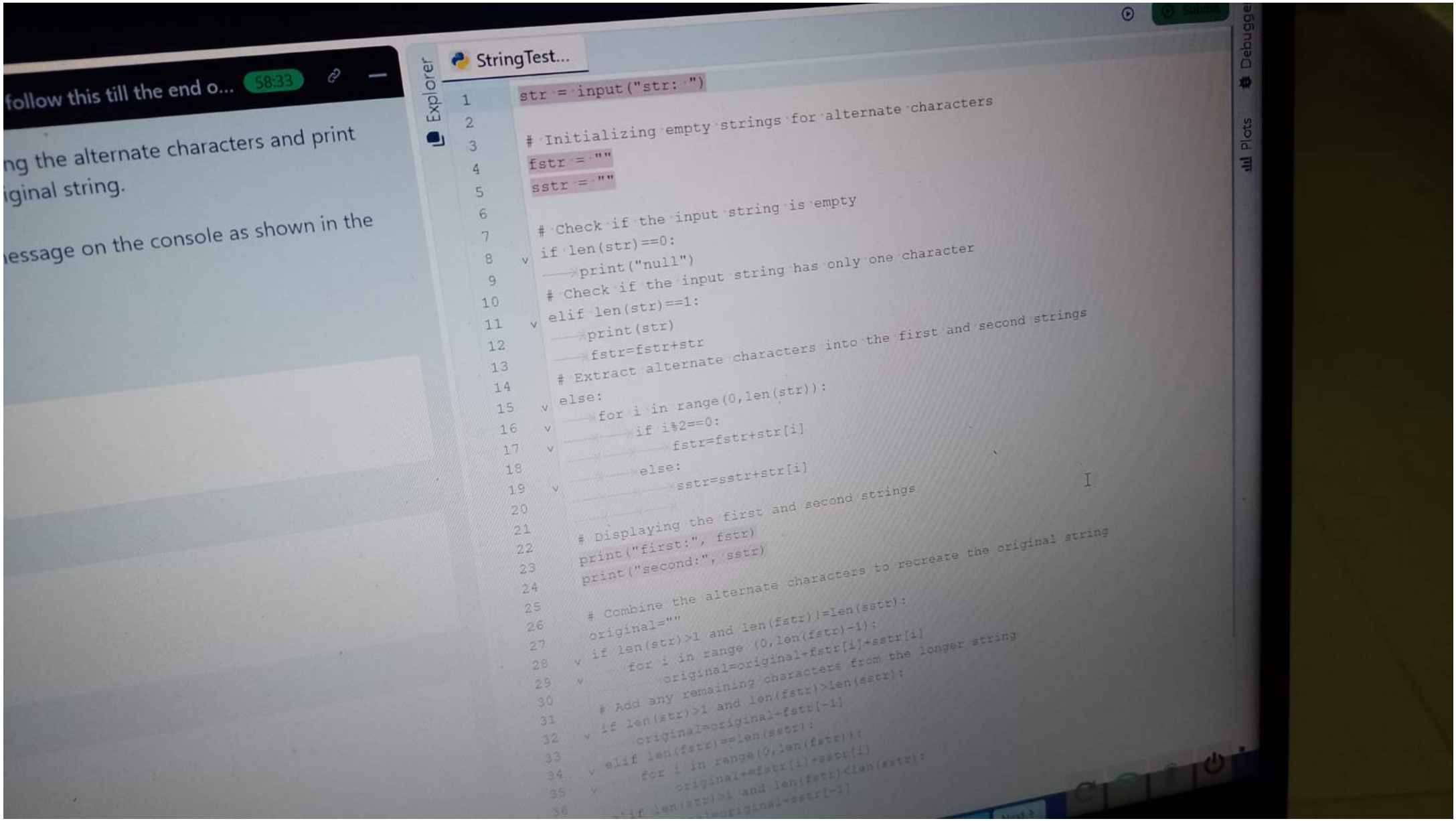

Home
Learn Anywhere
12314564.st@lpu.in
35.1.10. More String methods
02:59
A **module** is a file containing **Python definitions**, **functions** and **statements**.Standard library of **Python** is extended as `modules`. To use modules, we need to **import** them. After importing a module, we can use its functions and variables in our code. Python offers numerous standard modules. We will learn about modules in detail later.
Syntax:
import module_name
Here we use all functions of the string module.
import string
print(string.punctuation) # Result: !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
print(string.digits) # Result: 0123456789
print(string.printable) # Result: 0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!
print(string.capwords('hello python')) # Result: Hello Python
print(string.hexdigits) # Result: 0123456789abcdefABCDEF
print(string.octdigits) # Result: 01234567
Explorer
StringEx4.py
1 # Take a string input form the user
2 str=input("str: ")
3 # Reverse the string using slicing operator
4 print(str[::-1])

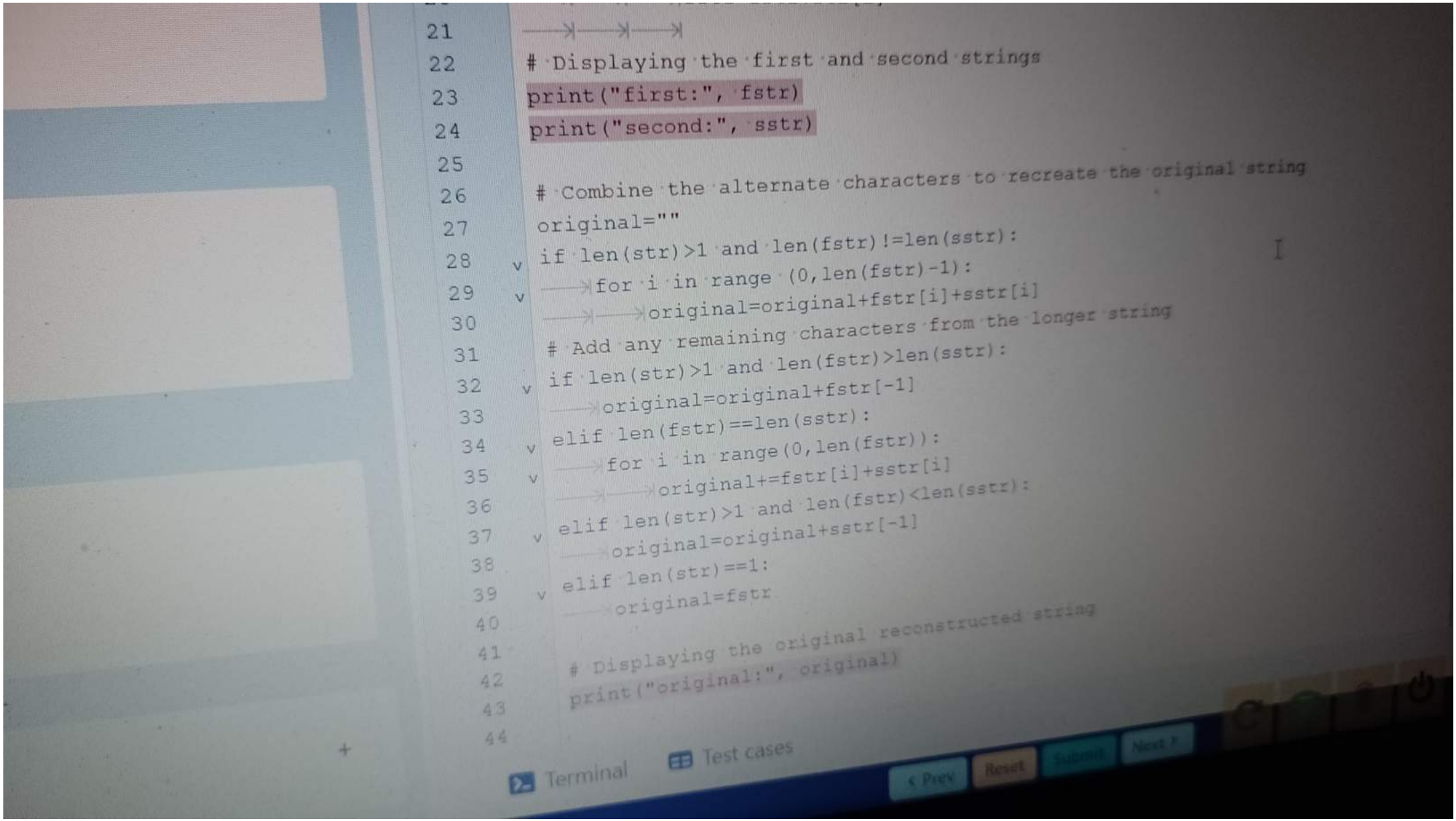

Home
Learn Anywhere
12314564.st@lpu.in
Support
35.1.11. Write a program to remove all digit in given string.
06:29
Below given is the program to remove all punctuation's in a string and print the result.
import string
punctuations = string.punctuation
result = " "
str = "List - []\n tuple - ()\n Dictionary - {}\n Comment - #\n Multiply
for i in str:
if i not in punctuations:
result = result + i
print("String after removing all Punctuation's:", result)
Here,
We are importing string module to know the list of punctuation's using string.punctuation .
Then we compare punctuations in the given string with the string.punctuation module and remove them from the input string and print the resultant string.
str variable contains input string.
for clause to iterate over the list of characters in the input string.
Explorer
StringTest...
import string
str = input("str: ")
result = "" # Initializing an empty string to store the result
digit=string.digits
# Iterate through each character in string (str), and if the character is not a digit, add it to th
for i in str:
if i not in digit:
result=result+i
# Print the result after removing all digits
print("String after removing all digits:",result)


12314564.st@lpu.in
Support
Logout
mes a particular sub string appears in a given string.
02:34
f times the second string occurs in the first string.
in the sample test case.
Explorer
StringTest1...
1 str1 = input("str1: ")
2 str2 = input("str2: ")
3
4 print("count:",str1.count(str2)) # print the count count of str2 occured in str1
Debugger
Plots


Home
Learn Anywhere
12314564.st@lpu.in
Support
Logout
35.1.13. Write a program to print every character of a string twice.
04:17
Write a program to print every character of the given input string twice.
Print the result to the console as shown in the sample test case.
Sample Input and Output:
str: Lists
result: LLiissttss
Explorer
StringTest1...
1 str = input("str: ")
2 result=""
3 for i in str:
4 result=result+(i*2)
5 # write your code here to print the every character in the given string twice
6 print("result: ",result)
Debugger
Picks


Home
Learn Anywhere
12314564.st@lpu.in
35.1.14. Write a program to print half of a given string.
07:19
Write a program to calculate length of the string, If length is **even** print first half of the string, if the length is **odd** remove the middle character.
Print the second half of the string as shown in sample test case.
Sample Input and Output 1:
str: Python
first half str of even length: Pyt
Sample Input and Output 2:
str: Hello World
second half str of odd length: World
Explorer
StringTest1...
1 str = input("str: ")
2 a=int(len(str)/2)
3 if len(str)%2==0:
4 print("first half str of even length:",str[:a])
5 else:
6 print("second half str of odd length:",str[a+1:])
7 # write your code here

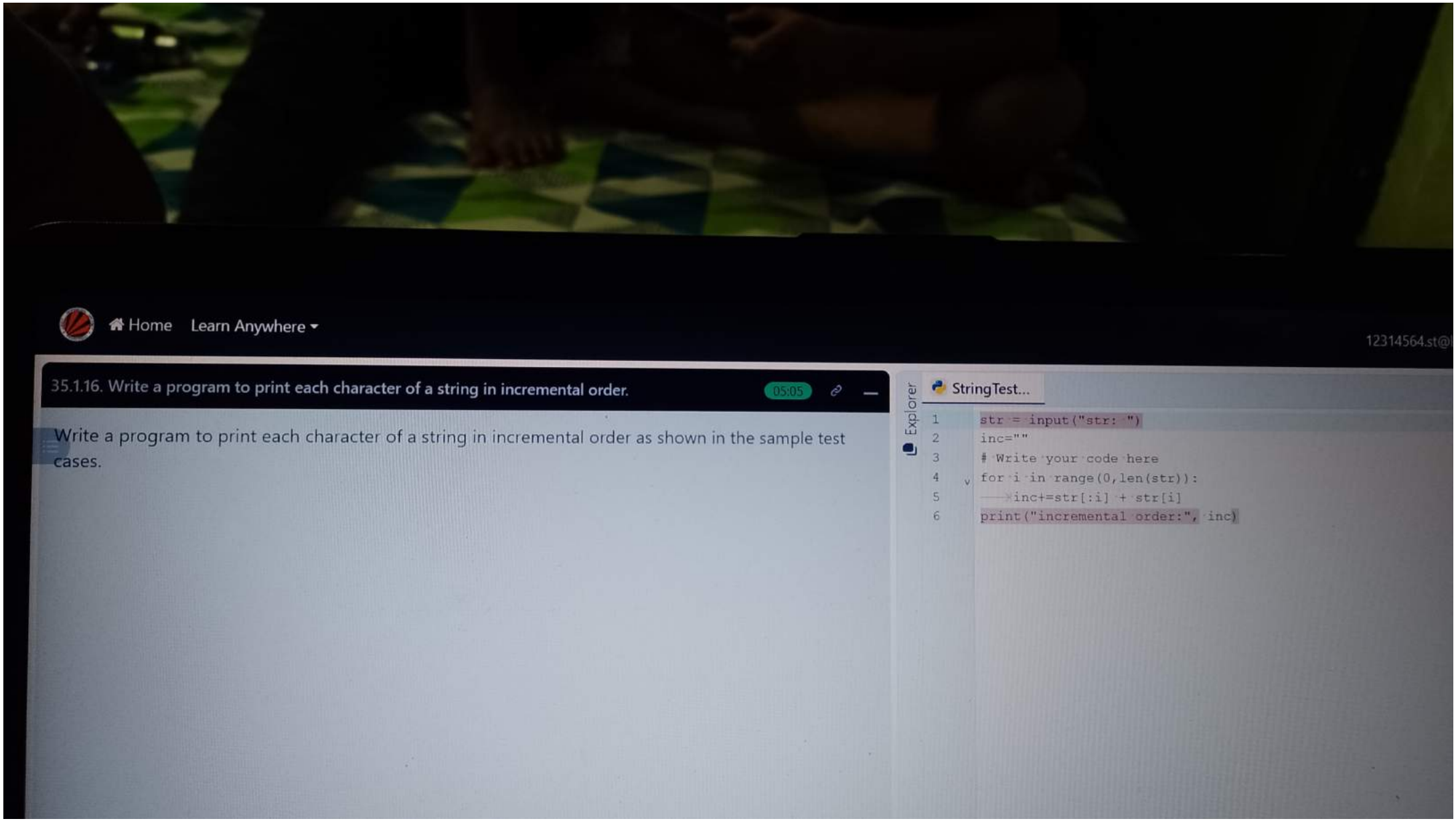

follow this till the end o...
58:33
ng the alternate characters and print
iginal string.
essage on the console as shown in the
Explorer
StringTest...
Debugger
Plots
str = input("str: ")
# Initializing empty strings for alternate characters
fstr = ""
sstr = ""
# Check if the input string is empty
if len(str)==0:
print("null")
# Check if the input string has only one character
elif len(str)==1:
print(str)
fstr=fstr+str
# Extract alternate characters into the first and second strings
else:
for i in range(0,len(str)):
if i%2==0:
fstr=fstr+str[i]
else:
sstr=sstr+str[i]
# Displaying the first and second strings
print("first:", fstr)
print("second:", sstr)
# Combine the alternate characters to recreate the original string
original=""
if len(str)>1 and len(fstr)!=len(sstr):
for i in range (0,len(fstr)-1):
original=original+fstr[i]+sstr[i]
# Add any remaining characters from the longer string
if len(str)>1 and len(fstr)>len(sstr):
original=original+fstr[-1]
elif len(fstr)==len(sstr):
for i in range(0,len(fstr)):
original+=fstr[i]+sstr[i]
if len(str)>1 and len(fstr)<len(sstr):
original+=sstr[-1]

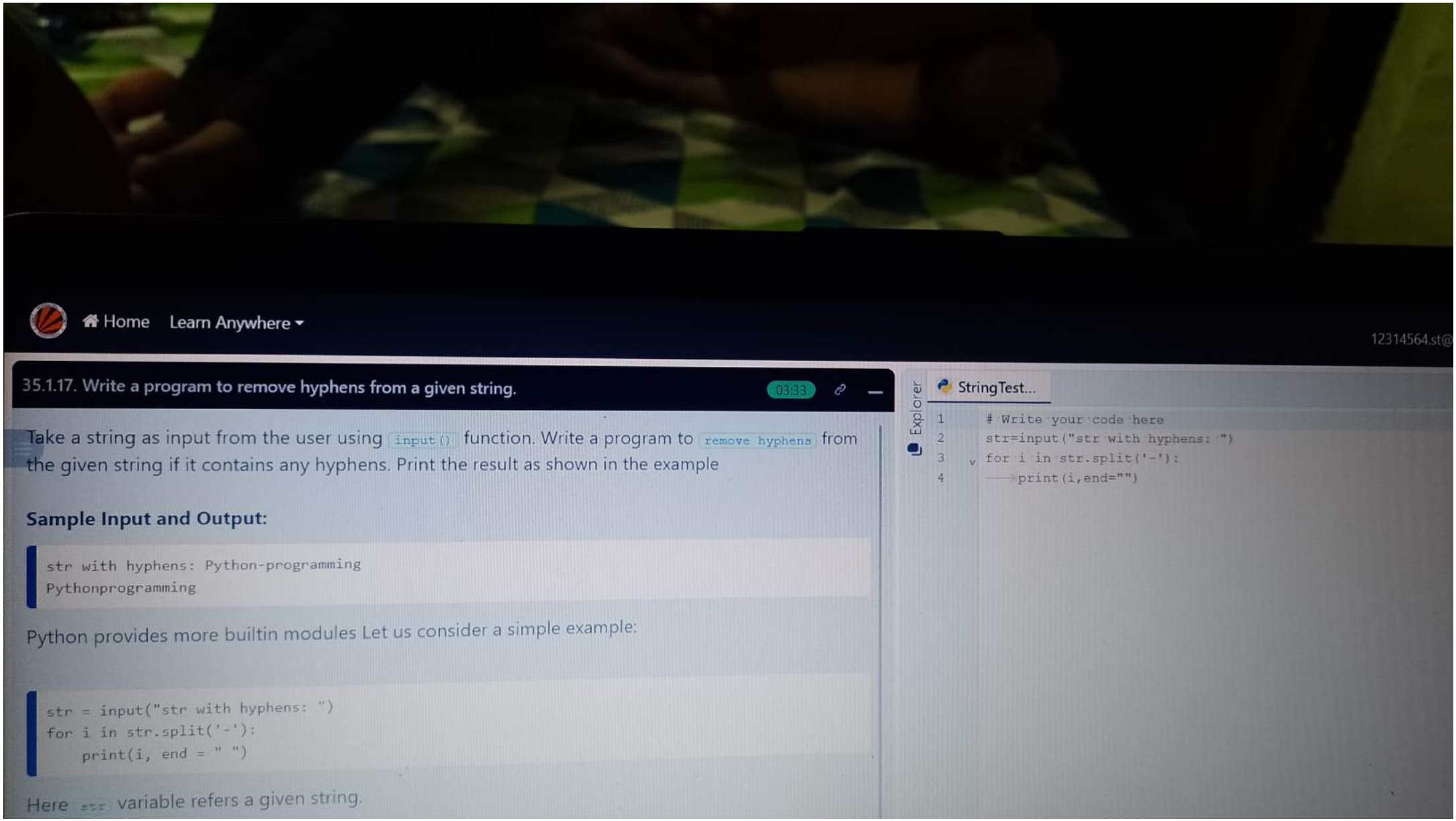


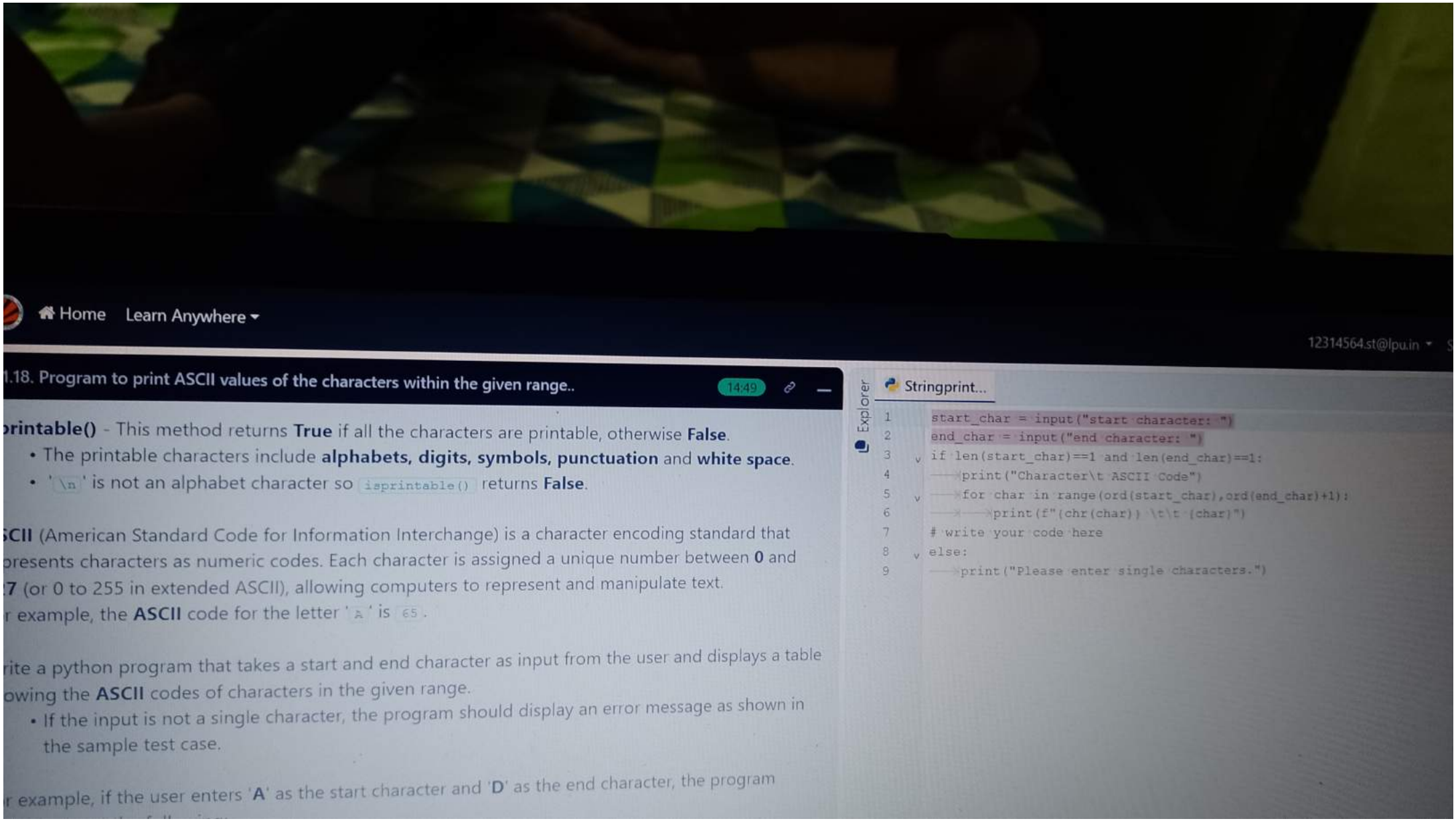
```
21
22  # Displaying the first and second strings
23  print("first:", fstr)
24  print("second:", sstr)
25
26  # Combine the alternate characters to recreate the original string
27  original=""
28  if len(str)>1 and len(fstr)!=len(sstr):
29      for i in range (0,len(fstr)-1):
30          original=original+fstr[i]+sstr[i]
31  # Add any remaining characters from the longer string
32  if len(str)>1 and len(fstr)>len(sstr):
33      original=original+fstr[-1]
34  elif len(fstr)==len(sstr):
35      for i in range(0,len(fstr)):
36          original+=fstr[i]+sstr[i]
37  elif len(str)>1 and len(fstr)<len(sstr):
38      original=original+sstr[-1]
39  elif len(str)==1:
40      original=fstr
41
42  # Displaying the original reconstructed string
43  print("original:", original)
44
```

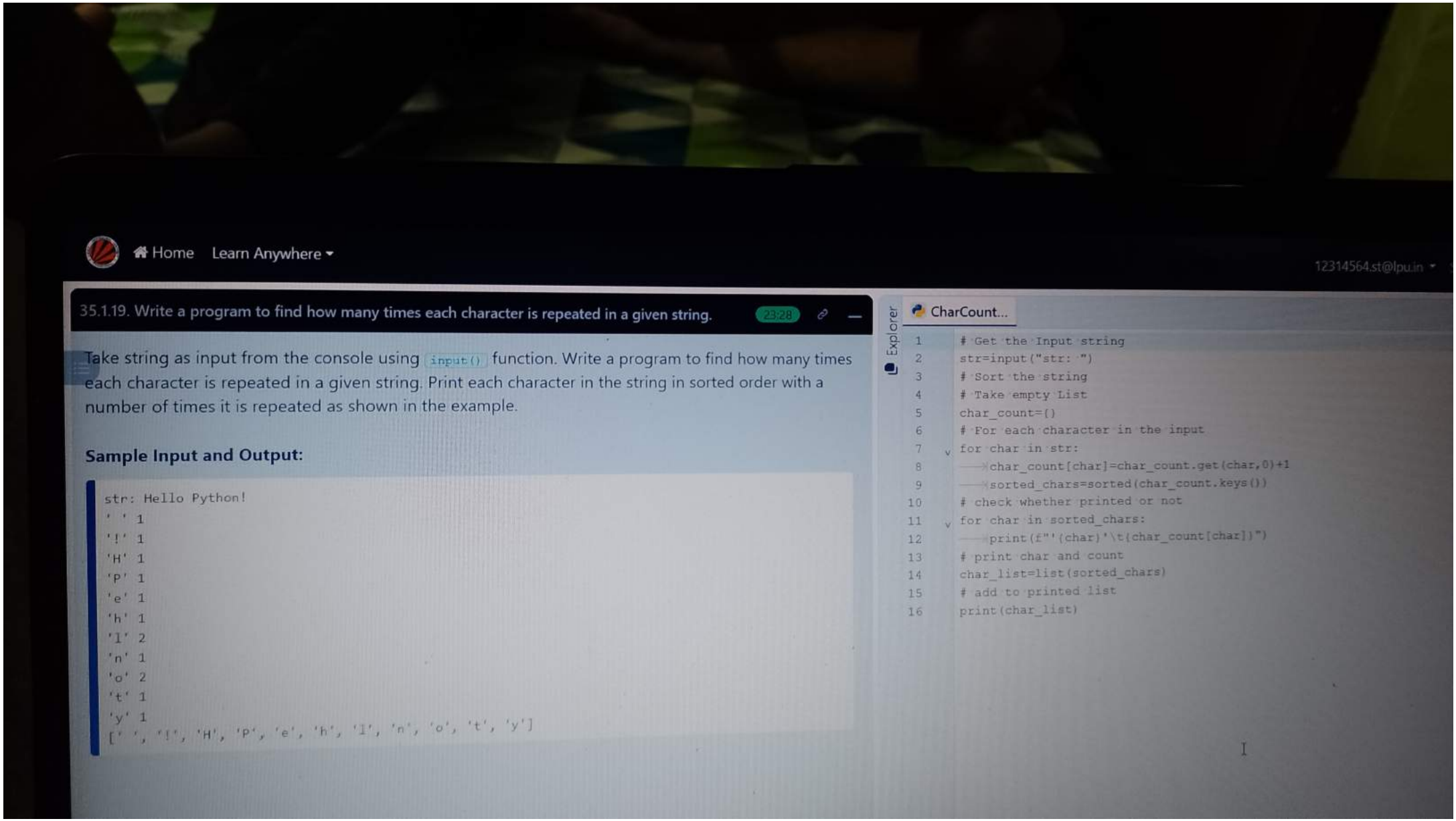
Terminal Test cases

< Prev Reset Submit Next >


Home
Learn Anywhere
12314564.st@
35.1.16. Write a program to print each character of a string in incremental order.
05:05
Write a program to print each character of a string in incremental order as shown in the sample test cases.
Explorer
StringTest...
1 str = input("str: ")
2 inc=""
3 # Write your code here
4 for i in range(0,len(str)):
5 inc+=str[:i] + str[i]
6 print("incremental order:", inc)


Home
Learn Anywhere
12314564.st@
35.1.17. Write a program to remove hyphens from a given string.
03:33
Take a string as input from the user using input() function. Write a program to remove hyphens from the given string if it contains any hyphens. Print the result as shown in the example
Sample Input and Output:
str with hyphens: Python-programming
Pythonprogramming
Python provides more builtin modules Let us consider a simple example:
str = input("str with hyphens: ")
for i in str.split('-'):
print(i, end = " ")
Here str variable refers a given string.
Explorer
StringTest...
1 # Write your code here
2 str=input("str with hyphens: ")
3 for i in str.split('-'):
4 print(i,end="")


Home
Learn Anywhere
12314564.st@lpu.in
1.18. Program to print ASCII values of the characters within the given range..
14:49
rintable() - This method returns True if all the characters are printable, otherwise False.
• The printable characters include alphabets, digits, symbols, punctuation and white space.
• '\n' is not an alphabet character so isprintable() returns False.
CII (American Standard Code for Information Interchange) is a character encoding standard that presents characters as numeric codes. Each character is assigned a unique number between 0 and 7 (or 0 to 255 in extended ASCII), allowing computers to represent and manipulate text.
r example, the ASCII code for the letter 'A' is 65.
rite a python program that takes a start and end character as input from the user and displays a table owing the ASCII codes of characters in the given range.
• If the input is not a single character, the program should display an error message as shown in the sample test case.
r example, if the user enters 'A' as the start character and 'D' as the end character, the program
Explorer
Stringprint...
start_char = input("start character: ")
end_char = input("end character: ")
if len(start_char)==1 and len(end_char)==1:
print("Character\t ASCII Code")
for char in range(ord(start_char),ord(end_char)+1):
print(f"{chr(char)} \t\t {char}")
# write your code here
else:
print("Please enter single characters.")


Home
Learn Anywhere
12314564.st@lpu.in
35.1.19. Write a program to find how many times each character is repeated in a given string.
23:28
Take string as input from the console using input() function. Write a program to find how many times each character is repeated in a given string. Print each character in the string in sorted order with a number of times it is repeated as shown in the example.
Sample Input and Output:
str: Hello Python!
' ' 1
'!' 1
'H' 1
'P' 1
'e' 1
'h' 1
'l' 2
'n' 1
'o' 2
't' 1
'y' 1
[' ', '!', 'H', 'P', 'e', 'h', 'l', 'n', 'o', 't', 'y']
Explorer
CharCount...
1 # Get the Input string
2 str=input("str: ")
3 # Sort the string
4 # Take empty List
5 char_count={}
6 # For each character in the input
7 for char in str:
8 char_count[char]=char_count.get(char,0)+1
9 sorted_chars=sorted(char_count.keys())
10 # check whether printed or not
11 for char in sorted_chars:
12 print(f"'{char}'\t{char_count[char]}")
13 # print char and count
14 char_list=list(sorted_chars)
15 # add to printed list
16 print(char_list)

Home Learn Anywhere

12314564.st@lpu.in

5.1.20. Write a program to find how many times a single digit is repeated in a given number. 20:24

Write a program to find how many times a single digit is repeated in a given number, and print the result as shown in the example.

Sample Input and Output:

```
str: 11454
1    2
4    2
5    1
```

Note: The output should be printed in the order of the appearance of digits in the given number.

Explorer

CountDigi...

```
def count_digits(str):
    digit_count={}
    for digit in str:
        if digit.isdigit():
            if digit in digit_count:
                digit_count[digit]+=1
            else:
                digit_count[digit]=1
    for digit,count in digit_count.items():
        print(f"{digit} \t {count}")
str=input("str: ")
count_digits(str)
```

Home Learn Anywhere

12314564.st@lpu.in Support Log

36.1.1. Introduction to Lists

00:56

List in Python is a fundamental data structure that serves as a container for holding an **ordered** collection of items/objects.

- The items of a list need not be of same data type.
- We can retrieve the elements of a list using "`index`".
- List can be arbitrary mixture of types like numbers, strings and other lists as well.
- They are of variable size i.e they can **grow** or **shrink** as required
- They are **mutable** which means the elements in the list can be changed/modified

Let us consider an example:

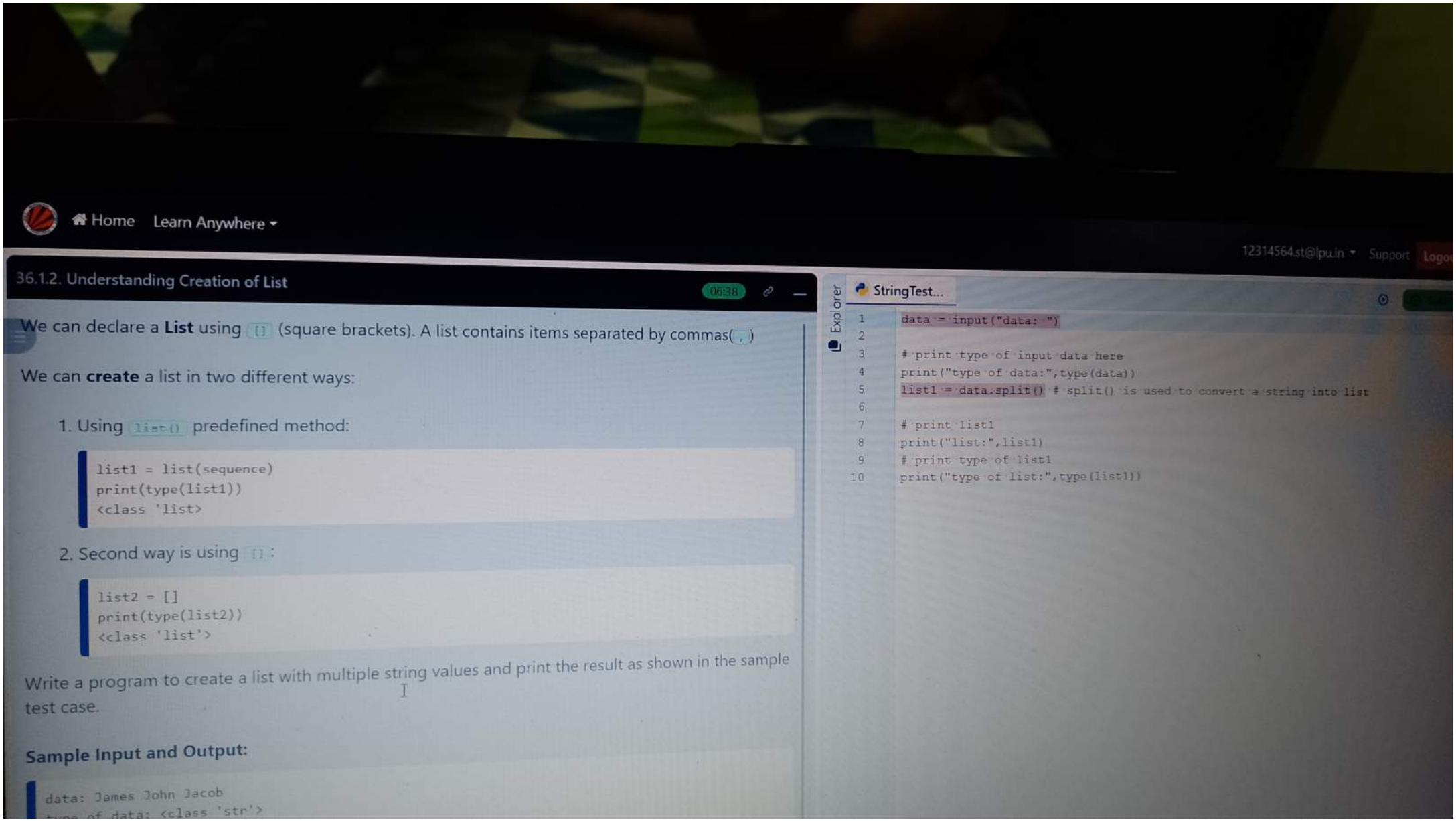
```
L1 = [56, 78.94, "India"]
```

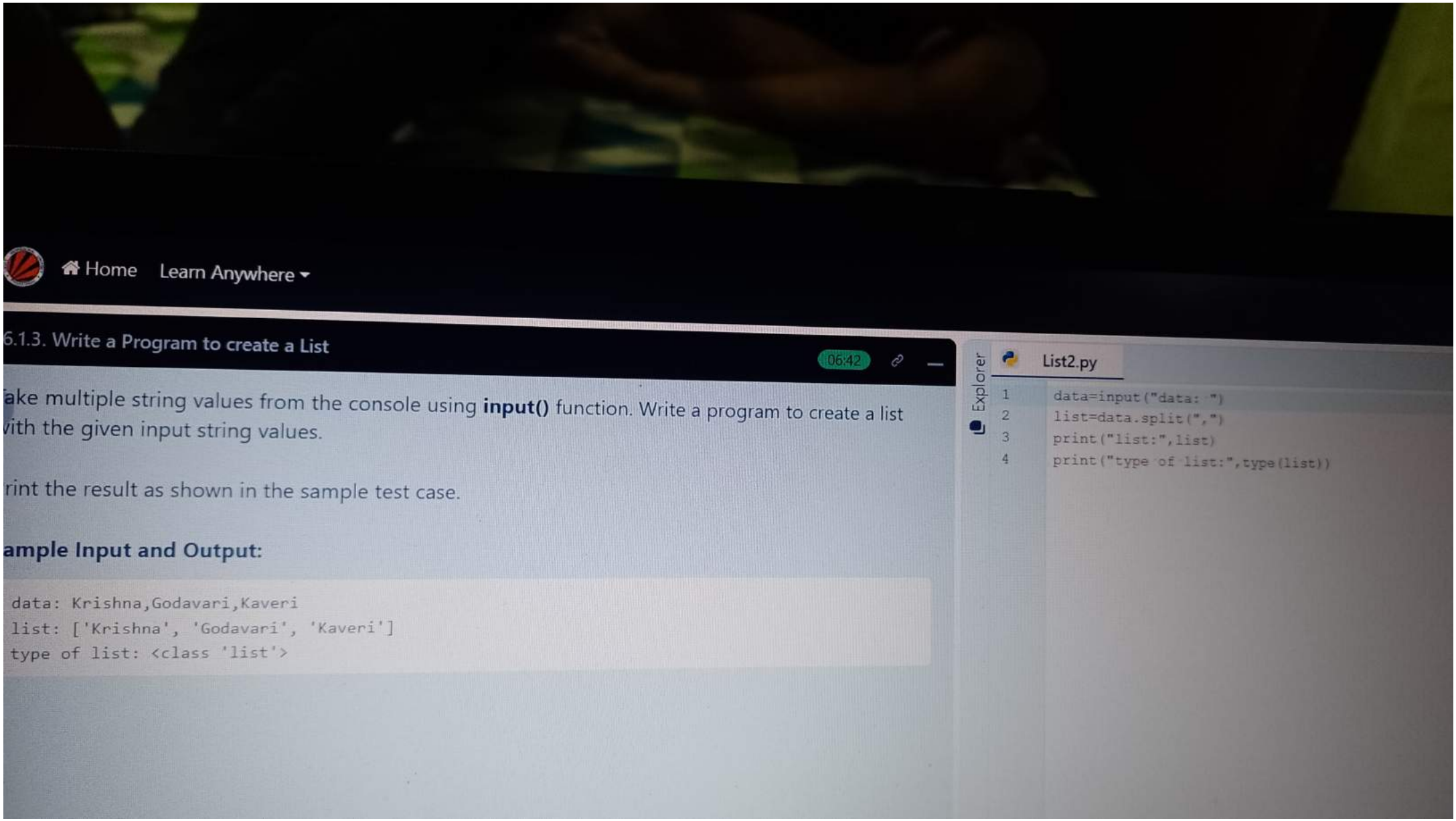
In the above example, `L1` is a list, which contains `3` elements. The first element is **56** (integer), the second is **78.94** (float), third is "**India**" (string).

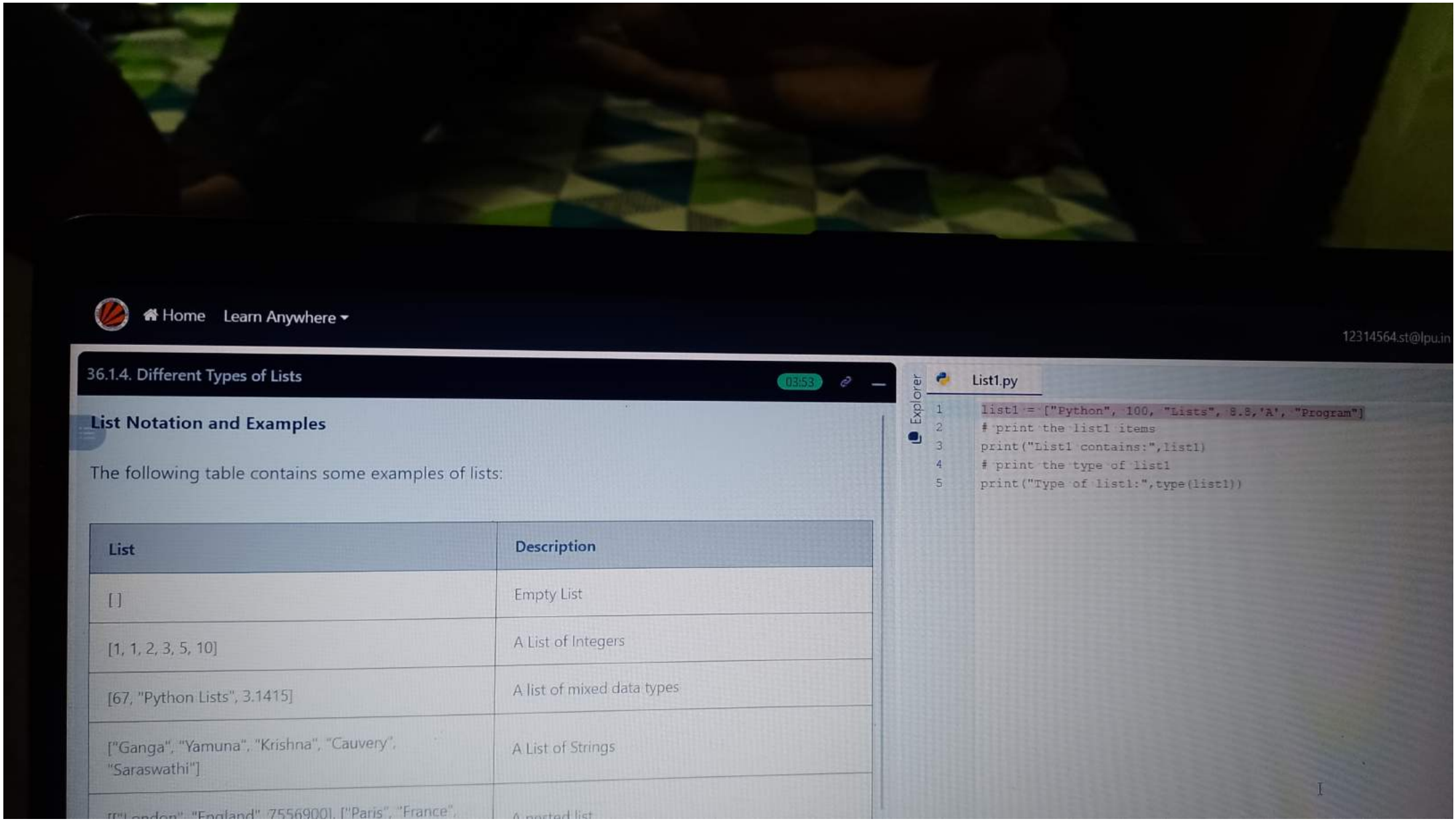
Click on Live Demo button to know List in **Python**.

Select all the correct statements given below.

- [ ] All elements in a list should be of same data type.
- [ ] Lists are immutable.
- [x] Lists are ordered and can contain other lists as elements.
- [ ] Indexing can be used with lists to access individual items only if there are no nested lists.


Home
Learn Anywhere
12314564.st@lpu.in
Support
36.1.2. Understanding Creation of List
06:38
We can declare a List using [] (square brackets). A list contains items separated by commas(,)
We can create a list in two different ways:
1. Using list() predefined method:
list1 = list(sequence)
print(type(list1))
<class 'list>
2. Second way is using [] :
list2 = []
print(type(list2))
<class 'list'>
Write a program to create a list with multiple string values and print the result as shown in the sample test case.
Sample Input and Output:
data: James John Jacob
Explorer
StringTest...
1 data = input("data: ")
2
3 # print type of input data here
4 print("type of data:",type(data))
5 list1 = data.split() # split() is used to convert a string into list
6
7 # print list1
8 print("list:",list1)
9 # print type of list1
10 print("type of list:",type(list1))


Home
Learn Anywhere
6.1.3. Write a Program to create a List
06:42
ake multiple string values from the console using input() function. Write a program to create a list
ith the given input string values.
rint the result as shown in the sample test case.
ample Input and Output:
data: Krishna,Godavari,Kaveri
list: ['Krishna', 'Godavari', 'Kaveri']
type of list: <class 'list'>
Explorer
List2.py
1 data=input("data: ")
2 list=data.split(",")
3 print("list:",list)
4 print("type of list:",type(list))


Home
Learn Anywhere
12314564.st@lpu.in
36.1.4. Different Types of Lists
03:53
List Notation and Examples
The following table contains some examples of lists:
List
Description
[]
Empty List
[1, 1, 2, 3, 5, 10]
A List of Integers
[67, "Python Lists", 3.1415]
A list of mixed data types
["Ganga", "Yamuna", "Krishna", "Cauvery", "Saraswathi"]
A List of Strings
Explorer
List1.py
1 list1 = ["Python", 100, "Lists", 8.8,'A', "Program"]
2 # print the list1 items
3 print("List1 contains:",list1)
4 # print the type of list1
5 print("Type of list1:",type(list1))

Home Learn Anywhere

12314564.st@lpu.in Support Logout

37.1.1. Understand the List Operations 03:59

Here is a list of operations that are possible on a list sequence:

Operations	Example	Description
Create a List	`a = [2 ,3, 4, 5, 6, 7, 8,` `print(a) # [2 ,3, 4, 5, 6,`	Create a comma separated list of elements and assign to variable 'a'
Indexing	`print(a[0])  # 2` `print(a[-1])  # 10`	Access the item at position '0' Access the last item using negative indexing
Slicing	`print(a[0:3]) # [2, 3, 4]` `print(a[0:])  # [2, 3, 4,`	Print a part of list starting at index '0' till index '2' Print a part of the list starting at index '0' till the end of list
Concatenation	`b = [20, 30]` `print(a + b)  # [2, 3, 4,`	Concatenate two lists and display its output
		Update the list element at index

- [x] l1 = [1, 20, 30, 40] index of l1 starts from **0**.
- [ ] Lists are immutable we are unable to update the list.
- [ ] a = [100, 23, 32] print(a * 0) returns [100, 23, 32].

Home Learn Anywhere

12314564.st@lpu.in Supp

37.1.2. Create and access a List

30:40

A list is created by using square brackets `[]` to enclose elements separated by commas `,`.For example:

```
a = [1, 2, 3, 4, 5, 6, 7, 8]
print(a)        # Output: [1, 2, 3, 4, 5, 6, 7, 8]
```

We can also use the `list()` constructor to create a list:

```
colors = list(("red", "blue", "green", "yellow"))
type(colors)    # <class 'list'>
print(colors)   # Output: ['red', 'blue', 'green', 'yellow']
```

To access elements in a list, we use the indexing operator `[]` with the index of the element.

- Indices range from `0` to `N-1`, where `N` is the number of elements in the list, when moving from left to right. We can also access elements in reverse, from right to left.
- The starting index is `-1`, which points to the last element and it goes up to `-N`, where `N` is the total number of elements.
- Trying to access an element beyond the index range will result in an error.

Examples:

```
print(a[0])       # Output: 1
print(a[-1])      # Output: 8
print(a[8])       # Returns: "IndexError" because valid indices are 0 to 7
print(a[-8])      # Output: 1
```

Explorer

List01.py

```
1   # write your code here
2   data=input("data: ")
3   list=data.split(",")
4   print("list:",list)
5   index=int(input("index: "))
6   if index>0 and index>=len(list):
7       print("invalid")
8   elif index<0 and -index>len(list):
9       print("invalid")
10  else:
11      print("element:",list[index])
```

Home Learn Anywhere

12314564.st@lp

37.1.3. Write a program to find whether a given element exists in a list or not

03:18

In **Python**, we can use the **in** and **not in** operators to determine whether an element is present in a list. These operators return either **True** or **False** based on the element's presence in the list.

For Example, given the list `x = [1, 2, 3, 4, 5, 6, 7]`:

```
print(2 in x)        # Returns True.
print(8 not in x)    # Returns True.
print(10 in x)       # Returns False.
```

Write a program to determine if a given element is present in a list or not. Print the result as shown in the sample test cases.

Sample Input and Output 1:

```
data: Python,Java,Perl
element: Java
True
```

Explorer

Listmem01...

```
1  data=input("data: ")
2  x=data.split(",")
3  element=input("element: ")
4  print(element in x)
```

Home Learn Anywhere

12314564.st@lpu.in

37.1.4. Write a Program to print the first and last elements of a List.

02:57

Create a list by taking input from the user for its elements. Write a program to find the first and last elements in the list.

Print the result as shown in the sample test case.

Sample Input and Output 1:

```
data: Godavari,Krishna,Kaveri
first, last elements: Godavari Kaveri
```

Sample Input and Output 2:

```
data: 10,20,30,45,56
first, last elements: 10 56
```

Explorer

List4.py

```
1  data=input("data: ")
2  x=data.split(",")
3  print("first, last elements:",x[0],x[-1])
```

Home Learn Anywhere

12314564.st@lpu.in

37.1.5. Understanding List slicing

24:28

List slicing extracts a subset of elements from a list to create a new list. The syntax for List slicing:

```
Listname[start:stop]
Listname[start:stop:steps]
```

- Default start: 0
- Default stop: n-1
- [:] prints the whole list
- [2:2] creates an empty slice

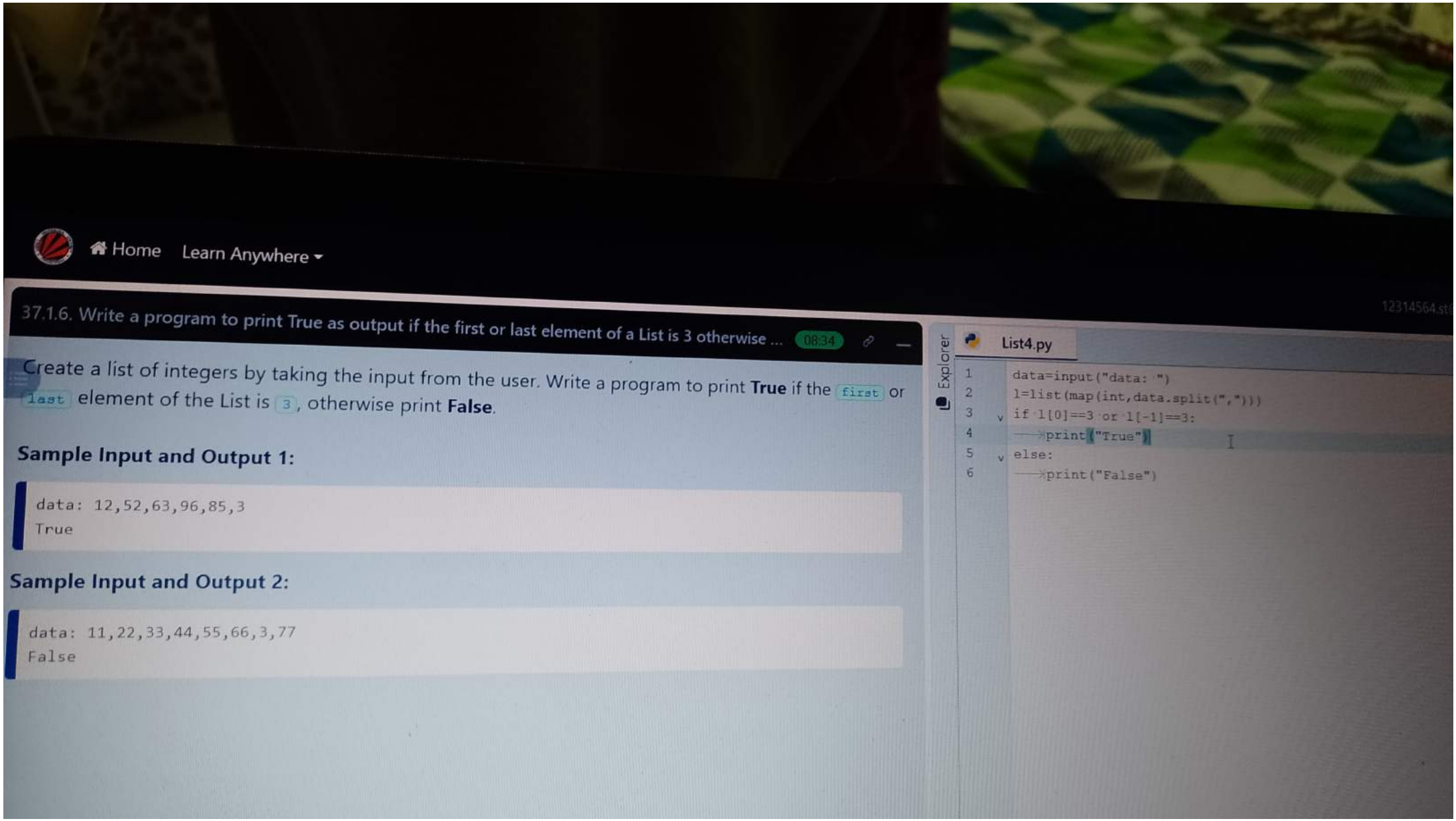
Let's consider `a = [9, 8, 7, 6, 5, 4]`, observe the following examples:

Slices	Example	Description
a[0:3]	a[0:3] [9, 8, 7]	Print a part of list from index 0 to 2
a[:4]	a[:4] [9, 8, 7, 6]	Default start value is 0. Prints the values from index 0 to 3

Explorer

Listslicing....

```
1  input_str = input("Enter a list of numbers separated by spaces: ")
2  # Convert input string into a list of integers
3  l1=input_str.split()
4  l1=[int(i) for i in l1]
5  # Get user inputs start,stop and step for slicing parameters
6  n1=int(input("start index: "))
7  n2=int(input("stop index: "))
8  n3=int(input("step value: "))
9
10
11 # Create a new list using slicing
12 l2=l1[n1:n2:n3]
13 # Display the sliced list
14 print("Sliced List:",l2)
15
```


Home
Learn Anywhere
37.1.6. Write a program to print True as output if the first or last element of a List is 3 otherwise ...
08:34
Create a list of integers by taking the input from the user. Write a program to print True if the first or last element of the List is 3, otherwise print False.
Sample Input and Output 1:
data: 12,52,63,96,85,3
True
Sample Input and Output 2:
data: 11,22,33,44,55,66,3,77
False
Explorer
List4.py
1 data=input("data: ")
2 l=list(map(int,data.split(",")))
3 if l[0]==3 or l[-1]==3:
4 print("True")
5 else:
6 print("False")

Home Learn Anywhere

37.1.7. Understanding List Repetition and Concatenation

46:21

List repetition uses the `*` operator to create a new list with elements repeated 'n' times.

For example, if y = [2, 3, 4]:

```
print(y * 0)  # Output: []
print(y * 2)  # Output: [2, 3, 4, 2, 3, 4]
print(y * 3)  # Output: [2, 3, 4, 2, 3, 4, 2, 3, 4]
```

List concatenation combines two lists using the `+` operator:

For example,

```
x = [1, 5, 6]
y = [2, 3, 4]
print(x + y)  # Output: [1, 5, 6, 2, 3, 4]
```

Python also offers the `extend()` method for this purpose. Call this method on the first list and pass the second list as an argument. This appends the second list to the first:

```
x.extend(y)
print(x)      # Output: [1, 5, 6, 2, 3, 4]
```

Explorer

Listcatrep...

```
1  # write your code here
2  data1=input("data1: ")
3  data2=input("data2: ")
4  num=int(input("num: "))
5  l1=data1.split(",")
6  l2=data2.split(",")
7  print(l1*num)
8  print(l2*num)
9  l1.extend(l2)
10 print("extending list1 with list2:",l1)
```

Home Learn Anywhere

12314564.st@lpu.in

37.1.8. Understanding list comparison

04:12

List comparison checks if two lists are equal or not using the `==` and `!=` operators, which returns a boolean value **True** or **False**.

For example,

```
x = [1, 2, 3, 4, 5, 6, 7]
y = [2, 3, 4]
print(x == y)    # Returns: False
print(x != y)    # Returns: True
```

Create two lists with the user-given inputs. Write a program to find whether the two lists contain the same elements or not.

Print the result as shown in the sample test cases.

Sample Input and Output 1:

```
data1: 45,56,58,59
data2: 59,45,58,56
is equal: False
is not equal: True
```

Explorer

Listcomp0...

```
1  data1=input("data1: ")
2  data2=input("data2: ")
3  l1=data1.split(",")
4  l2=data2.split(",")
5  if l1==l2:
6      print("is equal:",True)
7      print("is not equal:",False)
8  else:
9      print("is equal:",False)
10     print("is not equal:",True)
```

Home Learn Anywhere

37.1.9. Write a program to print EQUAL if first and last elements of a list are same otherwise prin... 0236

Create a list with the user-given inputs. Write a program to print **EQUAL** if first and last elements of a list are same, otherwise print **NOT EQUAL**.

Sample Input and Output 1:

```
data: Oliver,John,George,Oliver
equal
```

Sample Input and Output 2:

```
data: James,Charlie,Chaplin
not equal
```

Explorer

List6.py

```
1  list1=input("data: ").split(",")
2  if list1[0]==list1[-1]:
3      print("equal")
4  else:
5      print("not equal")
```

Home Learn Anywhere

12314564.st@lpu.in

37.1.10. Understanding Mutability in Lists

2106

Lists are **mutable** which means the items in it can be changed. **Mutability** involves altering specific data without complete recreation. List items can be changed by assigning new values using the indexing operator (`[]`).

Example	Description

`a = [1, 2, 3, 4, 5]`
`a[0] = 100`
`print(a)  # Output: [100, 2, 3, 4, 5]`
	Changing a single element

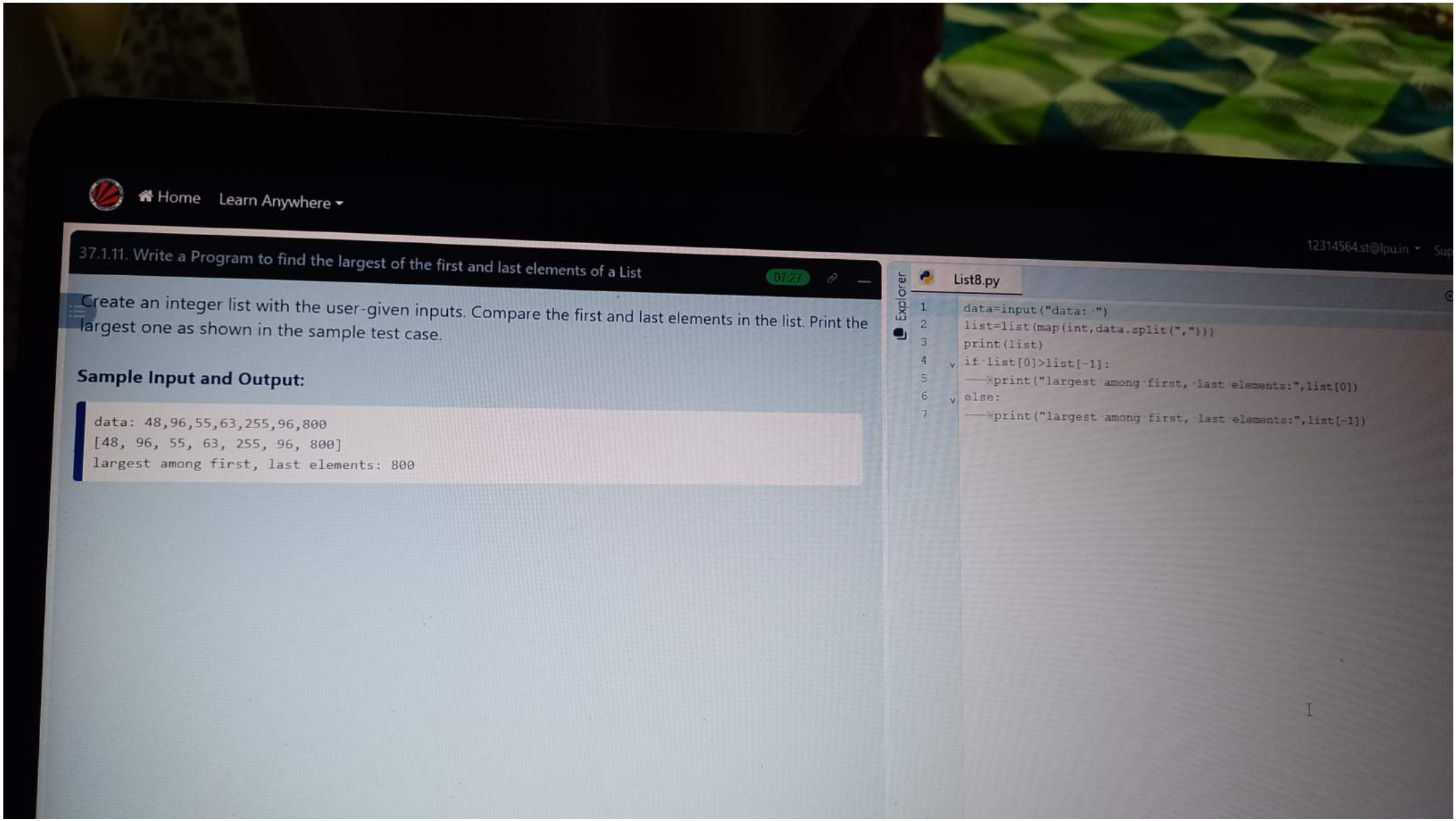
`a = [1, 2, 3, 4, 5]`
`a[0:3] = [100, 100, 100]`
`print(a)  # Output: [100, 100, 100, 4, 5]`
	Changing multiple elements

`a = [1, 2, 3, 4, 5]`
`a[0:3] = [ ]`
`print(a)  # Output: [4, 5]`
	Certain elements from a list can also be removed by assigning an empty list to them

Explorer

Listmutab....

```
# write your code here
data=input("data: ")
l=data.split(",")
print("before updation:",l)
n=int(input("index: "))
if n>0 and n<len(l):
    element=input("element: ")
    l[n]=element
    print("after updation:",l)
elif n<0 and -n<len(l):
    element=input("element: ")
    l[n]=element
    print("after updation:",l)
else:
    print("invalid")
```


Home
Learn Anywhere
12314564.st@lpu.in
37.1.11. Write a Program to find the largest of the first and last elements of a List
07:27
Create an integer list with the user-given inputs. Compare the first and last elements in the list. Print the largest one as shown in the sample test case.
Sample Input and Output:
data: 48,96,55,63,255,96,800
[48, 96, 55, 63, 255, 96, 800]
largest among first, last elements: 800
Explorer
List8.py
1 data=input("data: ")
2 list=list(map(int,data.split(",")))
3 print(list)
4 if list[0]>list[-1]:
5 print("largest among first, last elements:",list[0])
6 else:
7 print("largest among first, last elements:",list[-1])

Home Learn Anywhere

12314564:st@lpu.in

37.1.12. Understanding List aliasing

27:03

Aliasing occurs when a list is assigned to another variable, making both point to the same memory location. This leads to the same list being referenced by different names, causing changes from one reference to affect the other.

For example,

```
a = [1, 2, 3, 4, 5, 6]
b = a
print(a)  # Output: [1, 2, 3, 4, 5, 6]
print(b)  # Output: [1, 2, 3, 4, 5, 6]
b is a    # Returns: True
a is b    # Returns: True
a[0] = 100
print(a)  # Output: [100, 2, 3, 4, 5, 6]
print(b)  # Output: [100, 2, 3, 4, 5, 6]
```

Write a program to create a list with the user-given elements and assign this list to the new list. Update the first list with user-given values based on the user-given index. If the index does not exist in the list then print, **Please enter a valid index**. Follow the given sample test cases.

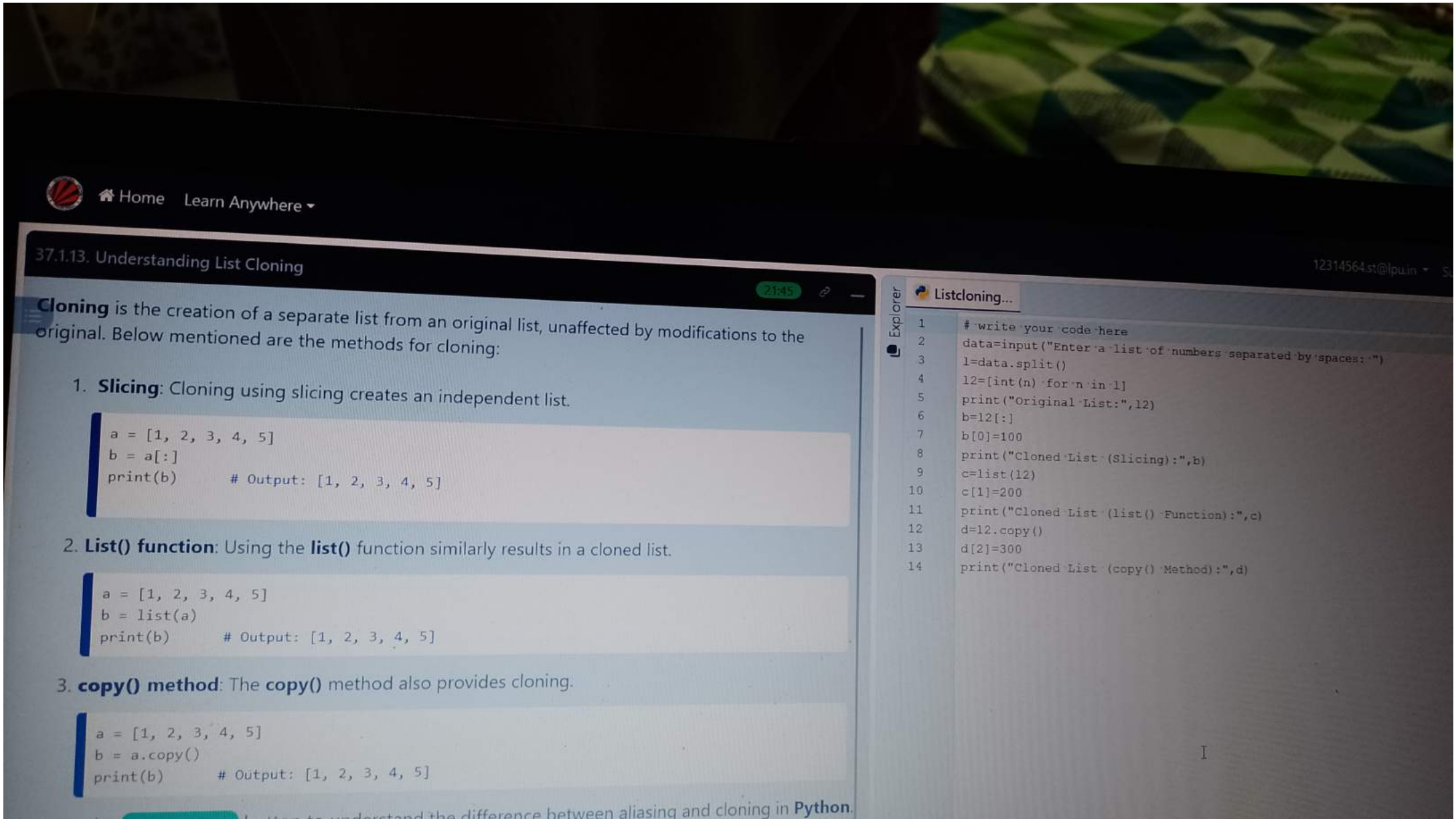
Sample Input and Output 1:

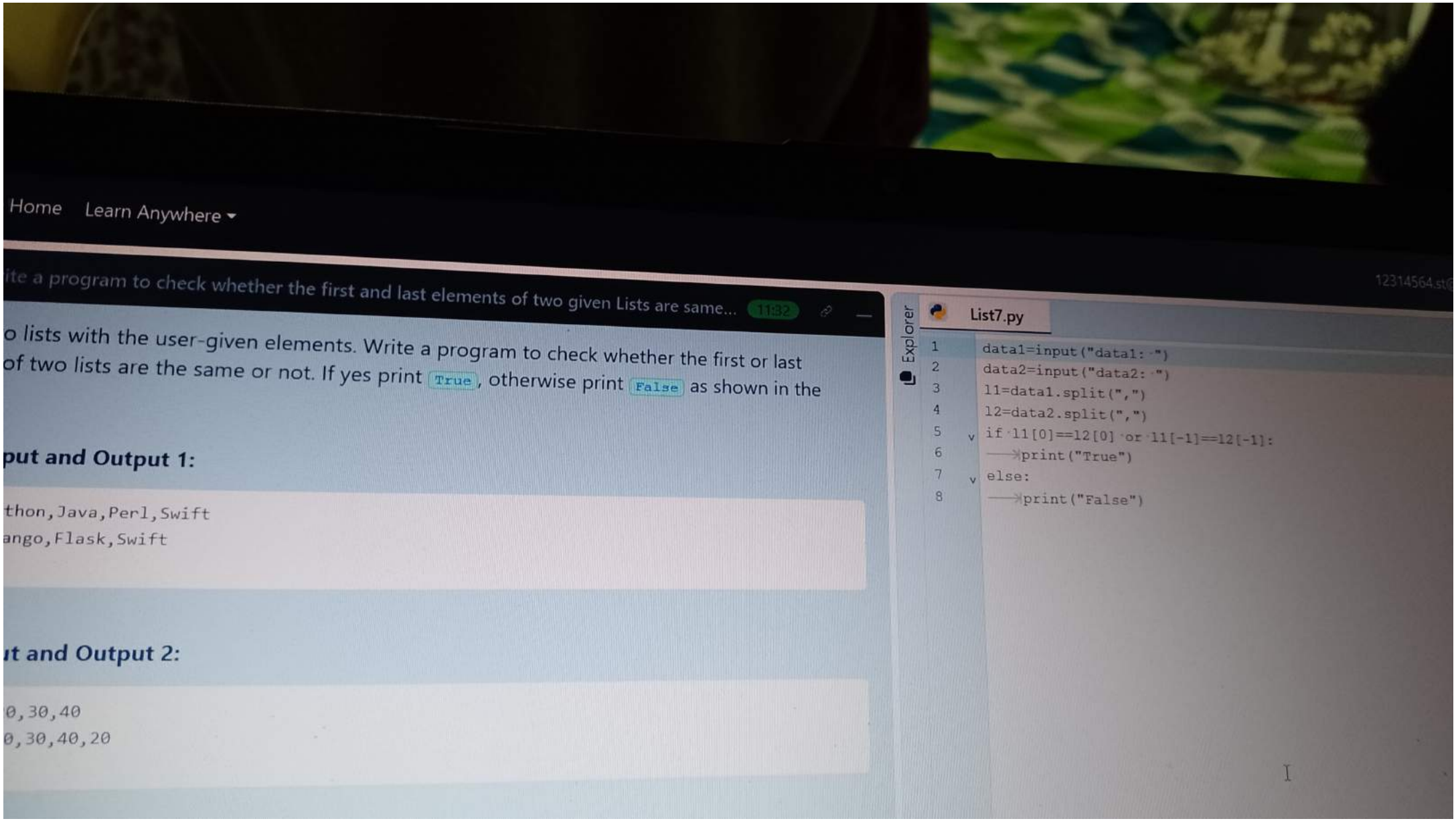
```
data: 10,20,30,40
list1 is list2: True
```

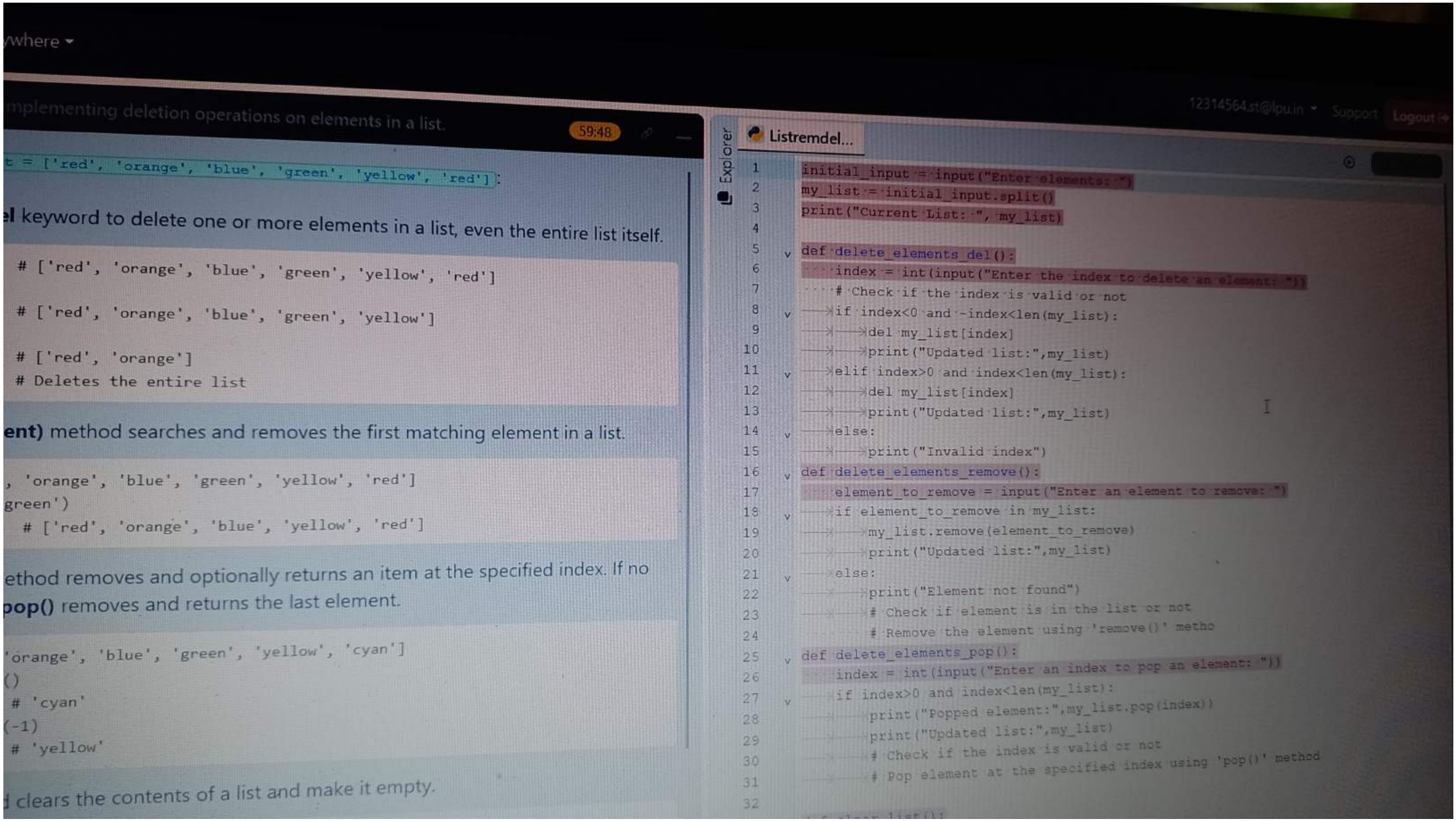
Explorer

Listalias.py

```
# write your code here
data1=input("data: ")
l1=list(data1.split(","))
l2=l1
print("list1 is list2:",l1 is l2)
print("list2 is list1:",l2 is l1)
n=int(input("index: "))
if n>0 and n<len(l1):
    element=input("element: ")
    l1[n]=element
    print("list1 is list2:",l1 is l2)
    print("list2 is list1:",l2 is l1)
elif n<0 and -n<=len(l1):
    element=input("element: ")
    l1[n]=element
    print("list1 is list2:",l1 is l2)
    print("list2 is list1:",l2 is l1)
else:
    print("enter valid index")
```


Home
Learn Anywhere
37.1.13. Understanding List Cloning
21:45
12314564.st@lpu.in
Cloning is the creation of a separate list from an original list, unaffected by modifications to the original. Below mentioned are the methods for cloning:
1. Slicing: Cloning using slicing creates an independent list.
a = [1, 2, 3, 4, 5]
b = a[:]
print(b) # Output: [1, 2, 3, 4, 5]
2. List() function: Using the list() function similarly results in a cloned list.
a = [1, 2, 3, 4, 5]
b = list(a)
print(b) # Output: [1, 2, 3, 4, 5]
3. copy() method: The copy() method also provides cloning.
a = [1, 2, 3, 4, 5]
b = a.copy()
print(b) # Output: [1, 2, 3, 4, 5]
understand the difference between aliasing and cloning in Python.
Explorer
Listcloning...
# write your code here
data=input("Enter a list of numbers separated by spaces: ")
l=data.split()
l2=[int(n) for n in l]
print("Original List:",l2)
b=l2[:]
b[0]=100
print("Cloned List (Slicing):",b)
c=list(l2)
c[1]=200
print("Cloned List (list() Function):",c)
d=l2.copy()
d[2]=300
print("Cloned List (copy() Method):",d)


Home
Learn Anywhere
ite a program to check whether the first and last elements of two given Lists are same...
11:32
o lists with the user-given elements. Write a program to check whether the first or last
of two lists are the same or not. If yes print True, otherwise print False as shown in the
put and Output 1:
thon,Java,Perl,Swift
ango,Flask,Swift
t and Output 2:
0,30,40
0,30,40,20
Explorer
List7.py
1 data1=input("data1: ")
2 data2=input("data2: ")
3 l1=data1.split(",")
4 l2=data2.split(",")
5 if l1[0]==l2[0] or l1[-1]==l2[-1]:
6 print("True")
7 else:
8 print("False")



wwhere

mplementing deletion operations on elements in a list.

59:48

t = ['red', 'orange', 'blue', 'green', 'yellow', 'red']:

el keyword to delete one or more elements in a list, even the entire list itself.

```
# ['red', 'orange', 'blue', 'green', 'yellow', 'red']

# ['red', 'orange', 'blue', 'green', 'yellow']

# ['red', 'orange']
# Deletes the entire list
```

ent) method searches and removes the first matching element in a list.

```
, 'orange', 'blue', 'green', 'yellow', 'red']
green')
  # ['red', 'orange', 'blue', 'yellow', 'red']
```

ethod removes and optionally returns an item at the specified index. If no pop() removes and returns the last element.

```
'orange', 'blue', 'green', 'yellow', 'cyan']
()
 # 'cyan'
(-1)
 # 'yellow'
```

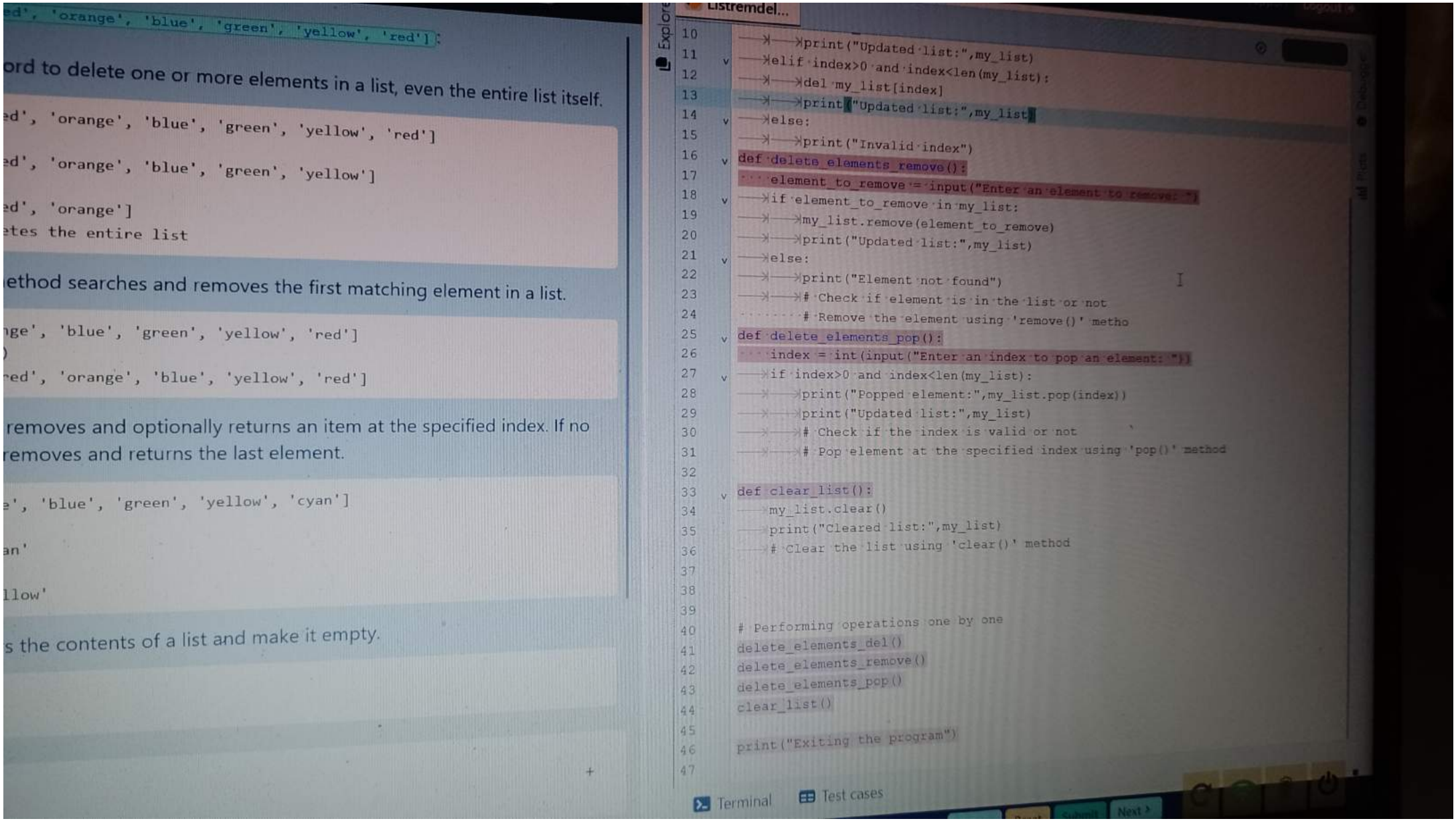
d clears the contents of a list and make it empty.

12314564.st@lpu.in Support Logout

Explorer

Listremdel...

```
initial_input = input("Enter elements: ")
my_list = initial_input.split()
print("Current List: ", my_list)

def delete_elements_del():
    index = int(input("Enter the index to delete an element: "))
    # Check if the index is valid or not
    if index<0 and -index<len(my_list):
        del my_list[index]
        print("Updated list:",my_list)
    elif index>0 and index<len(my_list):
        del my_list[index]
        print("Updated list:",my_list)
    else:
        print("Invalid index")
def delete_elements_remove():
    element_to_remove = input("Enter an element to remove: ")
    if element_to_remove in my_list:
        my_list.remove(element_to_remove)
        print("Updated list:",my_list)
    else:
        print("Element not found")
        # Check if element is in the list or not
        # Remove the element using 'remove()' metho
def delete_elements_pop():
    index = int(input("Enter an index to pop an element: "))
    if index>0 and index<len(my_list):
        print("Popped element:",my_list.pop(index))
        print("Updated list:",my_list)
        # Check if the index is valid or not
        # Pop element at the specified index using 'pop()' method

def clear_list():
```


ed', 'orange', 'blue', 'green', 'yellow', 'red']:
ord to delete one or more elements in a list, even the entire list itself.
ed', 'orange', 'blue', 'green', 'yellow', 'red']
ed', 'orange', 'blue', 'green', 'yellow']
ed', 'orange']
etes the entire list
ethod searches and removes the first matching element in a list.
nge', 'blue', 'green', 'yellow', 'red']
red', 'orange', 'blue', 'yellow', 'red']
removes and optionally returns an item at the specified index. If no
removes and returns the last element.
e', 'blue', 'green', 'yellow', 'cyan']
an'
llow'
s the contents of a list and make it empty.
Listremdel...
print("Updated list:",my_list)
elif index>0 and index<len(my_list):
del my_list[index]
print("Updated list:",my_list)
else:
print("Invalid index")
def delete_elements_remove():
element_to_remove = input("Enter an element to remove: ")
if element_to_remove in my_list:
my_list.remove(element_to_remove)
print("Updated list:",my_list)
else:
print("Element not found")
# Check if element is in the list or not
# Remove the element using 'remove()' metho
def delete_elements_pop():
index = int(input("Enter an index to pop an element: "))
if index>0 and index<len(my_list):
print("Popped element:",my_list.pop(index))
print("Updated list:",my_list)
# Check if the index is valid or not
# Pop element at the specified index using 'pop()' method
def clear_list():
my_list.clear()
print("Cleared list:",my_list)
# Clear the list using 'clear()' method
# Performing operations one by one
delete_elements_del()
delete_elements_remove()
delete_elements_pop()
clear_list()
print("Exiting the program")
Terminal
Test cases
Next

Home Learn Anywhere

37.1.16. Write a program to remove all the duplicates from a List.

59:40

Create a list with the user-given input elements. Write a program to remove all the duplicate elements from the given input list.

Print the result as shown in the sample test case.

Sample Input and Output:

```
data: 10,20,10,30
['10', '20', '10', '30']
after removing duplicates: ['10', '20', '30']
```

12314564.st@lpu.in Support Logo

Explorer

List11.py

```
# write your code here
"""
data=input("data: ")
l=data.split(",")
print(l)
if type(l[0])==int):
    print("after removing duplicates:",sorted(list(set(l))))
else:
    print("after removing duplicates:",l)
"""
def rem(l):
    unique_list=list(sorted(set(l)))
    return unique_list

data=input('data: ')
l=data.split(",")
print(l)
print("after removing duplicates:",rem(l))
```

Home Learn Anywhere

12314564.st@lpu.in Support Logout

38.1.1. Understanding List functions

Here is a list of built-in functions that can be applied on a list:

1. **all()**: Returns True if all the items in the list have a true value.

```
print(all([' ', ',', '1', '2']))
True
```

2. **any()**: Returns True if atleast one item in the list has a true value.

```
print(any([' ', ',', '1', '2']))
True
```

3. **enumerate()**: Returns an enumerate object consisting of the index and the values of all items of the list as a tuple pair.

```
print(list(enumerate(['a','b','c','d','e'])))
[(0, 'a'), (1, 'b'), (2, 'c'), (3, 'd'), (4, 'e')]
```

4. **len()**: It calculates the length of the list i.e., the number of elements in the list.

```
print(len(['a', 'b', 'c', 'd', 'e']))
5
```

5. **max()**: Returns the item with the highest value from the list

Explorer

Listfuncs.py

```
1  data = input("data: ")
2  list1 = data.split(",")
3  #find the length
4  size = len(list1)
5
6  #convert the elements into integer type
7  for i in range(size):
8      list1[i] = int(list1[i])
9  print("length:",size )#print the length of list1
10 print("list enumerate:",list(enumerate(list1)))#print the enumerate of list1
11 print("max:",max(list1))#print the maximum value of list1
12 print("min:",min(list1))#print the minimum value of list1
13 print("list:",sorted(list1))#print the sorted list
14
```

Home Learn Anywhere

12314564.st@lpu.in

38.1.2. Write a Program to print the difference between maximum and minimum values of a List

24:17

Write a program to **find the difference between the maximum and the minimum** elements of a list and print the result as shown in the example.

Sample Input and Output:

```
data: 12,52,96,85
min: 12
max: 96
difference: 84
```

Explorer

List22.py

```
1  data=input("data: ")
2  l=data.split(",")
3  print("min:",min(l))
4  print("max:",max(l))
5  a=int(max(l))
6  b=int(min(l))
7  print("difference:",a-b)
```

Home Learn Anywhere

38.1.3. Write a program to find the sum of elements of a given integer List 01:12

Write a program **to find the sum of elements** of the given integer list and print the result as shown in the example.

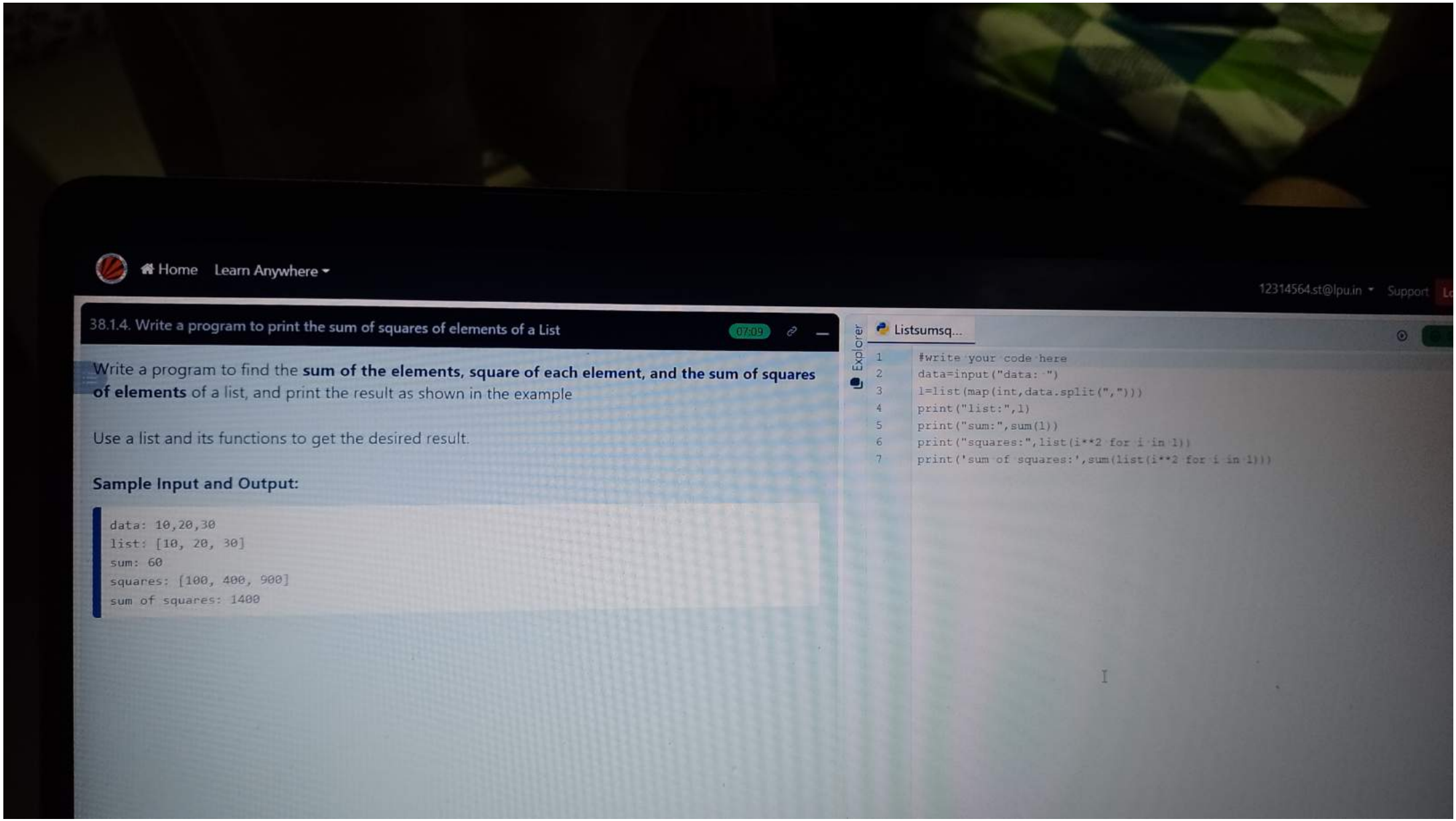
Sample Input and Output:

```
data: 10,20,30,40,50,60,70,80,90,100
sum: 550
```

Explorer

List9.py

```
1  data=input("data: ")
2  l=list(map(int,data.split(",")))
3  print("sum:",sum(l))
```


Home
Learn Anywhere
12314564.st@lpu.in
Support
38.1.4. Write a program to print the sum of squares of elements of a List
07:09
Write a program to find the sum of the elements, square of each element, and the sum of squares of elements of a list, and print the result as shown in the example
Use a list and its functions to get the desired result.
Sample Input and Output:
data: 10,20,30
list: [10, 20, 30]
sum: 60
squares: [100, 400, 900]
sum of squares: 1400
Explorer
Listsumsq...
1 #write your code here
2 data=input("data: ")
3 l=list(map(int,data.split(",")))
4 print("list:",l)
5 print("sum:",sum(l))
6 print("squares:",list(i**2 for i in l))
7 print('sum of squares:',sum(list(i**2 for i in l)))

Home Learn Anywhere

12314564.st@lpu.in

38.1.5. Write a program to print a dictionary using two given Lists

14:54

Write a program to print a dictionary using two input lists **list1** and **list2**. Consider **list1** elements as keys and **list2** elements as values. Print the output as shown in the example.

Steps to construct the output string:

Step 1:

create a string with the value = "{"

i.e., outstr = "{"

Step 2:

Combine the elements from **list1** and **list2** within **single quotes**, joined by a colon ':' and ending with **comma**(,) for all elements of **list1** and add it to the output string.

Step 3:

Delete the last **comma** (,) and add "}". Print the output string.

Sample Input and Output 1:

```
data1: Python,Java,Swift
data2: 111,222,333
{'Python':'111','Java':'222','Swift':'333'}
```

Sample Input and Output 2:

```
data1: A,U,G,S,N
```

Explorer

DictUsingL...

```
#write your code here
data1=input("data1: ")
data2=input("data2: ")
l1=data1.split(",")
l2=data2.split(",")
if len(l1)==len(l2):
    outstr="{"
    for i in range(len(l1)):
        outstr+="'"+l1[i]+"':'"+l2[i]+"',"
    outstr=outstr[:-1]+"}"
    print(outstr)
else:
    print("lists are different lengths")
```

Home Learn Anywhere

12314564.st@lpu.in Support

38.1.6. Write a program to print a list created from the absolute differences of elements in the c... 11:50

Write a program to read two integer lists of equal lengths, and if two lists differ in length then print **Please enter two integer lists of same size.**

Create a new list that contains the absolute differences between corresponding elements of the given lists. Print the result as shown in the examples.

Sample Input and Output 1:

```
data1: 10,20,30
data2: 5,10,15
[5, 10, 15]
```

Sample Input and Output 2:

```
data1: 15,45,65,85,96
data2: 20,25,32,65,45,68,69,45
lists are different lengths
```

Explorer

List10.py

```
#write your code here
data1=input("data1: ")
data2=input("data2: ")
l1=data1.split(",")
l2=data2.split(",")
if len(l1)==len(l2):
    diff_list=[(int(l1[i])-int(l2[i])) for i in range(len(l1))]
    print(diff_list)
else:
    print("lists are different lengths")
```

39.1.1. Overview of List methods

0247

Here is a list of methods that can be applied to list:

1. **append()**: Adda a single item at the end of the list.

```
x = ['a', 'b', 'c', 'd']
x.append('e')
print(x)
['a', 'b', 'c', 'd', 'e']
x.append([1, 2, 3])
print(x)
['a', 'b', 'c', 'd', [1, 2, 3]]
```

Click on Live Demo button to know how to **append** elements to a List in Python.

2. **extend()**: It is used to extend a list by appending elements from an iterable (such as another list, tuple, or string) to the end of the list. This method modifies the original list and does not create a new list.

```
x = ['a', 'b', 'c', 'd']
x.extend([1, 2, 3, 4])
print(x)
['a', 'b', 'c', 'd', 1, 2, 3, 4]
```

3. **insert(index, item)**: It is used to insert a specified element (item) at a specific position (index) in a list.

- [x] append()
- [x] extend()
- [x] insert(index, item)
- [x] remove(element)
- [x] pop()

Home Learn Anywhere

39.1.2. Overview of List methods 09:25

6. **count(element)**: It is used to count the occurrences of a specified element within a list.

```
x = ['a', 'b', 'a', 'c', 'd', 'e', 'a']
x.count('a')
3
```

7. **sort(key = None, reverse = False)**: It is used to sort the elements of a list in ascending order. It modifies the original list in-place and does not return a new list. If the parameter reverse = True, then the list is sorted in descending order.

```
x = ['z', 'f', 'e', 'a','b', 'g', 't']
x.sort()
print(x)
['a', 'b', 'e', 'f', 'g', 't', 'z']
x.sort(key = None, reverse = True)
print(x)
['z', 't', 'g', 'f', 'e', 'b', 'a']
```

8. **reverse()**: It is used to reverse the order of elements of a list in place.

```
x = ['z', 'f', 'e', 'a','b', 'g', 't']
```

Explorer

Listmetho...

```
1  #write your code here
2  data1=input("data1: ")
3  l=data1.split(",")
4  element=input("element: ")
5  l.append(element)
6  print("after append:",l)
7  data2=input("data2: ")
8  l2=data2.split(",")
9  l.append(l2)
10 print("after append:",l)
11 l.extend(l2)
12 print("after extending:",l)
```

Home Learn Anywhere

12314564.st

39.1.3. Write a Program to create a List using append() method 08:49

Write a program that takes the **size** and creates a list using **append()** method, print the result as shown in the example.

Sample Input and Output:

```
size: 3
element: 123
element: 145
element: 789
list: ['123', '145', '789']
```

Explorer

List9.py

```
#write your code here
size=int(input('size: '))
l=[]
for i in range(size):
    element=input("element: ")
    l.append(element)
print("list:",l)
```

Home Learn Anywhere

39.1.4. Write a program to reverse a given List

17:03

Create a list with the user-given elements. Write a program to reverse the given list, and print the result to the console as shown in the example.

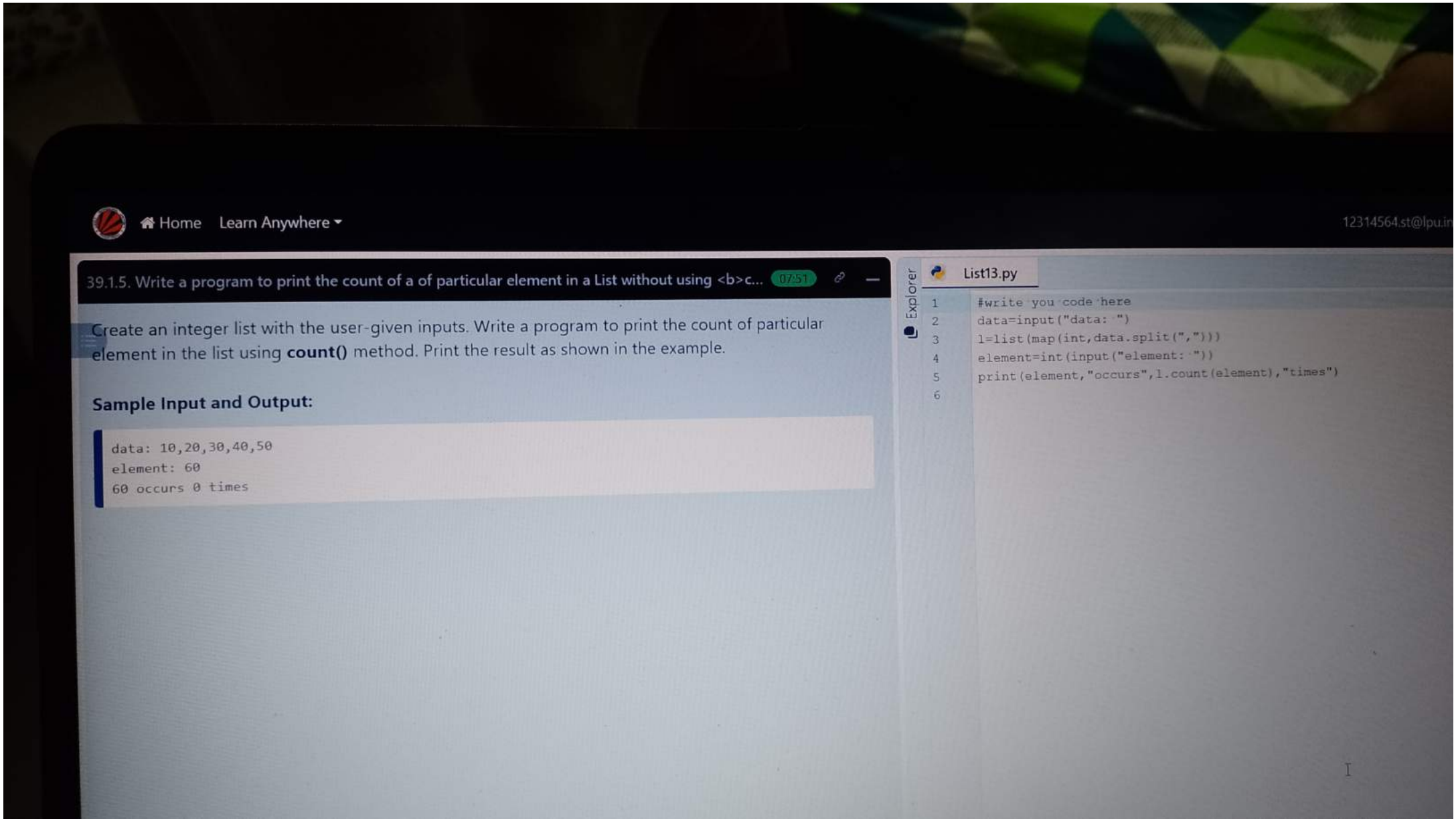
Sample Input and Output:

```
data: Python,Java,Perl,Swift,R
reverse: ['R', 'Swift', 'Perl', 'Java', 'Python']
```

Explorer

List12.py

```
1  #write your code here
2  data=input("data: ")
3  l=data.split(",")
4  l.reverse()
5  print("reverse:",l)
```


Home
Learn Anywhere
12314564.st@lpu.in
39.1.5. Write a program to print the count of a of particular element in a List without using <b>c...
07:51
Create an integer list with the user-given inputs. Write a program to print the count of particular element in the list using count() method. Print the result as shown in the example.
Sample Input and Output:
data: 10,20,30,40,50
element: 60
60 occurs 0 times
Explorer
List13.py
1 #write you code here
2 data=input("data: ")
3 l=list(map(int,data.split(",")))
4 element=int(input("element: "))
5 print(element,"occurs",l.count(element),"times")
6

Home Learn Anywhere

12314564.st@lpu

39.1.6. Write a program to find the sum of elements of a List without using sum() meth... 39:53

Create an integer list with the user-given elements. Write a program to find the sum of elements of the given input list without using the method **sum()**, and print the result as shown in the example.

Sample Input and Output:

```
data: 10,20,30,40,50,60,70,80,90,100
sum: 550
```

Explorer

List13.py

```
1  data = input("data: ")
2  list1 = data.split(",")
3  #complete the loop to convert the elements to integer type
4  for i in range(len(list1)):
5      list1[i] = int(list1[i])
6  sum1 = 0
7  #complete the loop to find the sum
8  for i in range(len(list1)):
9      sum1 = sum1+list1[i]
10 print("sum:",sum1)
11
```

Home Learn Anywhere

12314564.st@lpu.in

39.1.7. Write a program to find a given element, if the element to be found and its next element ... 31:41

Write a program to find the given element in a list. If the element to be found and its next element are the same, then return **True**, otherwise return **False**.

Sample Input and Output - 1:

```
list1: 32,36,36,5
num: 36
True
```

Sample Input and Output - 2:

```
list1: 33,34,35
num: 34
False
```

Explorer

List15.py

```
#write your code here
data=input("data: ")
list1=list(map(int,data.split(",")))
print("list:",list1)
num=int(input("num: "))
s=0
for i in range(len(list1)-1):
    if list1[i]==num and list1[i+1]==num:
        s+=1
if s==1:
    print("True")
else:
    print("False")

```

Home Learn Anywhere

39.1.8. Write a Program to find Sequence of elements in a List

07:13

Create an integer list with the user-given inputs. Write a program to find given **list of elements** are present in the input list. If yes, print **exist**, else print **does not exist** as shown in the examples.

> **Note:** Given elements need not to be present in the same order.

Sample Input and Output 1:

```
data: 10,20,30,40,50,60,70,80,90
seq of elements: 60,70,80,90
exist
```

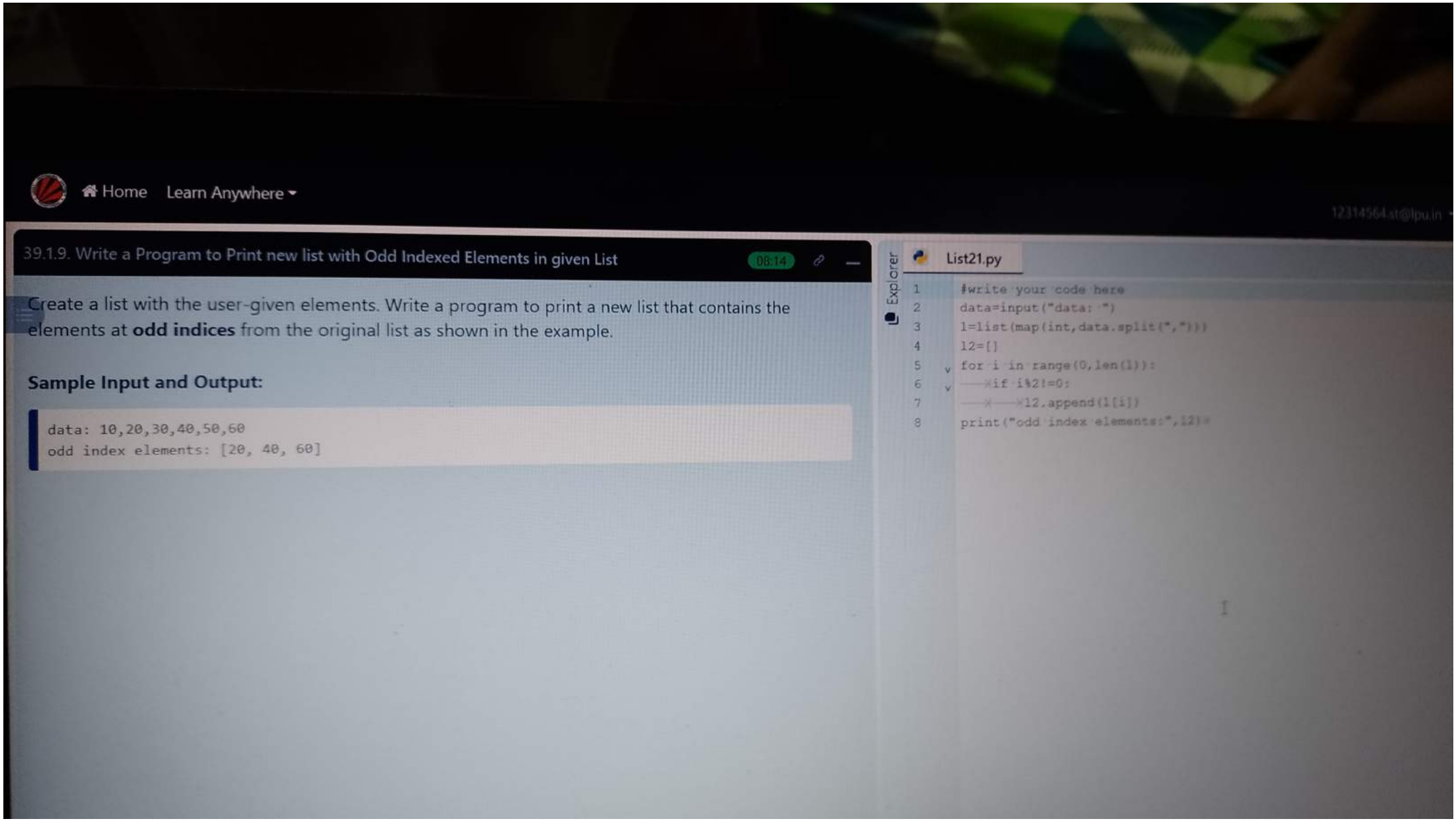
Sample Input and Output 2:

```
data: 85,96,456,12,36,84
seq of elements: 99,12,36,84
does not exist
```

Explorer

List16.py

```
#write your code here
data=input("data: ")
l=data.split(",")
seq=input("seq of elements: ")
l2=seq.split(",")
if all(i in l for i in l2):
    print("exist")
else:
    print("does not exist")

```


Home
Learn Anywhere
39.1.9. Write a Program to Print new list with Odd Indexed Elements in given List
08:14
Create a list with the user-given elements. Write a program to print a new list that contains the elements at **odd indices** from the original list as shown in the example.
Sample Input and Output:
data: 10,20,30,40,50,60
odd index elements: [20, 40, 60]
Explorer
List21.py
#write your code here
data=input("data: ")
l=list(map(int,data.split(",")))
l2=[]
for i in range(0,len(l)):
if i%2!=0:
l2.append(l[i])
print("odd index elements:",l2)

Home Learn Anywhere

12314564.st@lpu.i

39.1.10. Write a Program to Add all alphabets both small and upper case letters into a ... 19:44

Write a program to add all the alphabets of both upper case and lower case letters into a list as shown in the sample input and output.

Pseudo Code:

```
INITIALISE i = ord('A')
INITIALISE j = ord('a')
INITIALISE alpha = [ ]
for (k < 26):
  append chr(i) To alpha
  append chr(j) To alpha
  INCREMENT i by 1
  INCREMENT j by 1
PRINT alpha
```

Note:

1. Use **ord()** method to convert an alphabet(character) into its ASCII values.
2. Use **chr()** method to convert a number into its respective character.

Sample Input and Output:

```
A in upper case: A
```

Explorer

ListAlpha.py

```
#write your code here
i=ord("A")
j=ord("a")
print("Please enter",chr(i),"in upper case: ")
print("Please enter",chr(i),"in lower case: ")
alpha=[]
for k in range(26):
    alpha.append(chr(i))
    alpha.append(chr(j))
    i+=1
    j+=1
print(alpha)
```

Home Learn Anywhere

12314564.st

39.1.11. Write a program to check whether the given list is in sorted order or not 02:24

Create an integer list with the user-given elements. Write a program to check whether the given list is in **sorted order** or not. If the given list is in sorted order, then return **True**, otherwise return **False**

Sample Input and Output 1:

```
data: 10,20,30,40,50,60,70
True
```

Sample Input and Output 2:

```
data: 78,45,195,365,485,5201
False
```

Explorer

ListSorted....

```
#Write your code here
data=input("data: ")
l=list(map(int,data.split(",")))
s=0
for i in range(len(l)-1):
    if l[i]>l[i+1]:
        s+=1
if s==0:
    print("True")
else:
    print("False")
```

Home Learn Anywhere

12314564.st@lpu

39.1.12. Write a Program to Print Number of Characters in a given String using List

55:06

Write a program to find the occurrence of every character in an input string, print the result as shown in the example.

Note: Ignore all letters other than alphabets and convert all the upper case letters into lower case.

Sample Input and Output:

```
Please enter a sentence: Hyderabad Tourism
a    2
b    1
d    2
e    1
h    1
i    1
m    1
o    1
r    2
s    1
t    1
```

Explorer

Charcount...

```
#write your code here
a=input("Please enter a sentence: ")
a.lower()
l={}
for i in a:
    if i.isalpha():
        i=i.lower()
        if i in l:
            l[i]+=1
        else:
            l[i]=1
for i,count in sorted(l.items()):
    print(f"{i} \t {count}")
```