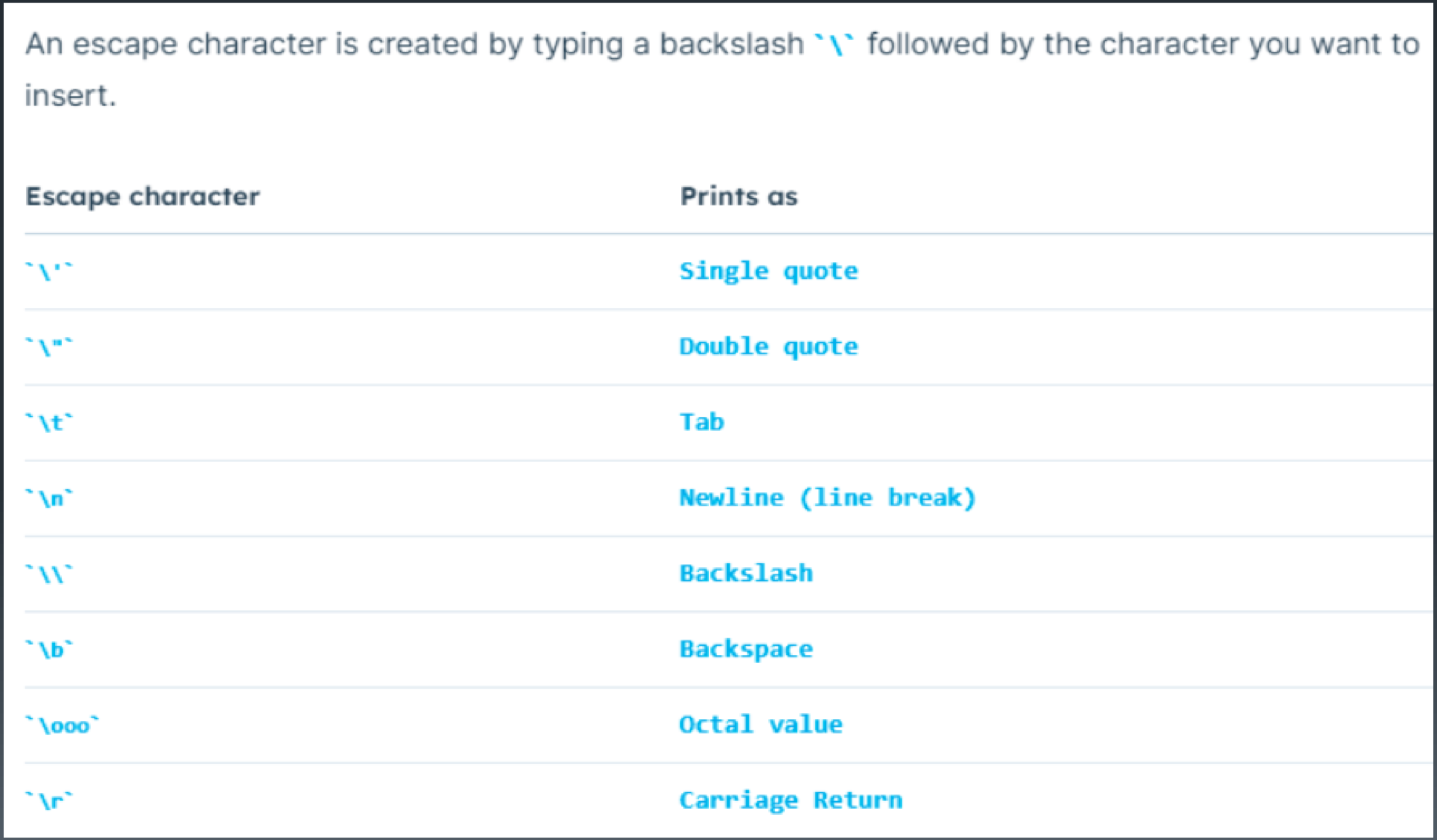
STRING MANIPULATION IN PYTHON

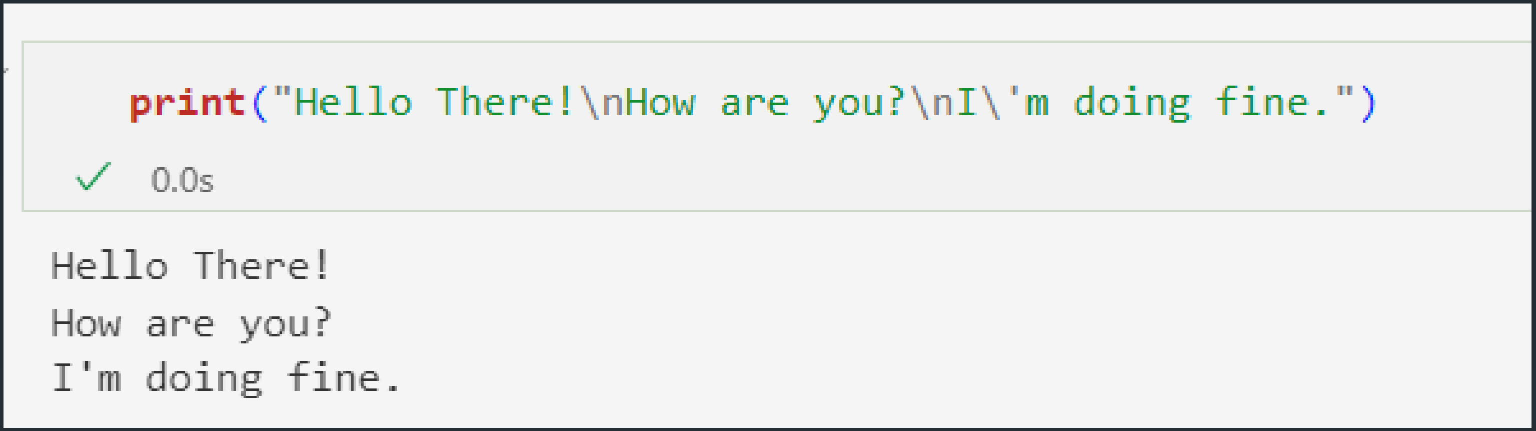
Made by: SYED IRTIZA ABBAS ZAIDI

ESCAPE CHARACTER:



An escape character is created by typing a backslash `\` followed by the character you want to insert.

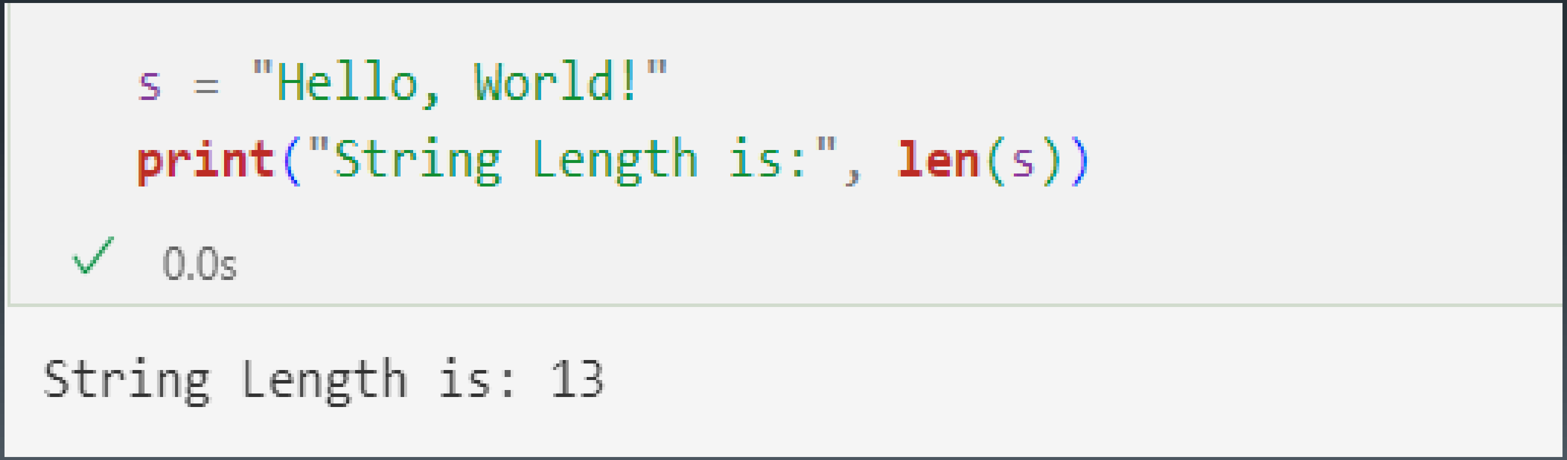
Escape character	Prints as
`\'`	Single quote
`\"`	Double quote
`\t`	Tab
`\n`	Newline (line break)
`\\`	Backslash
`\b`	Backspace
`\ooo`	Octal value
`\r`	Carriage Return



```
print("Hello There!\nHow are you?\nI\'m doing fine.")
✓ 0.0s
```

```
Hello There!
How are you?
I'm doing fine.
```

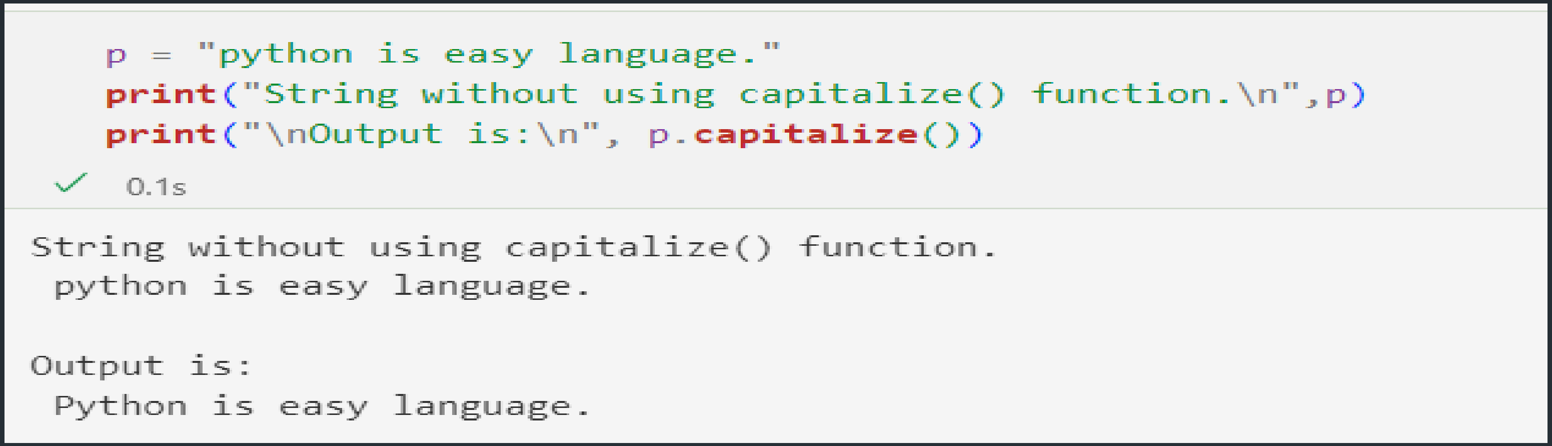
1) Length: len() Returns the length of the string.



```
s = "Hello, World!"
print("String Length is:", len(s))
✓ 0.0s
```

```
String Length is: 13
```

2) capitalize(): Coverts the first character to uppercase.

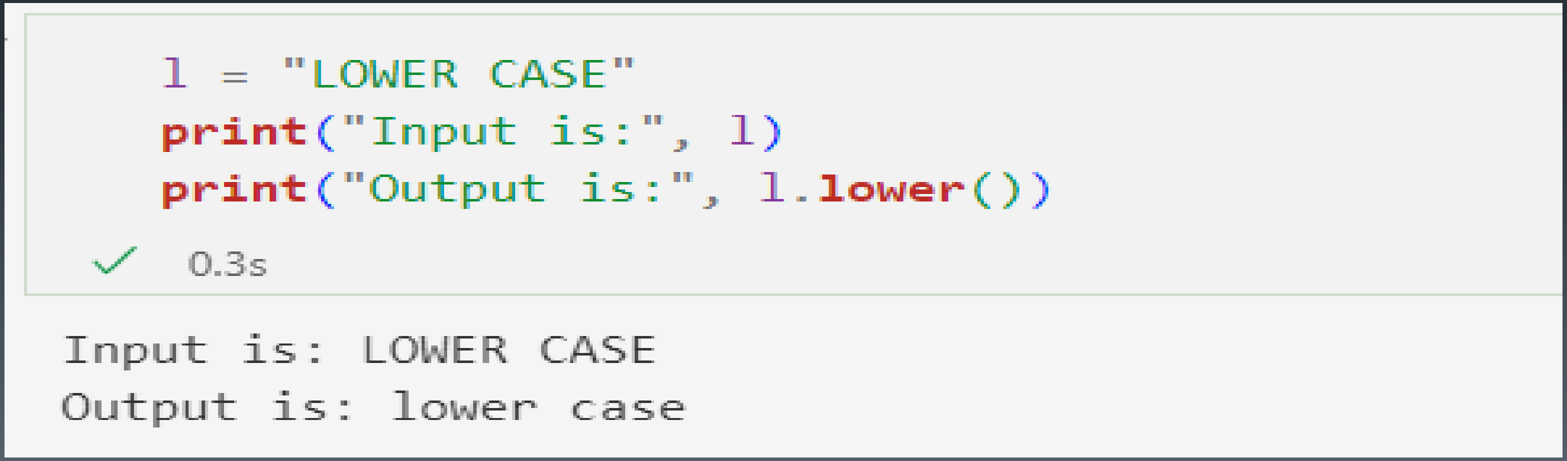


```
p = "python is easy language."
print("String without using capitalize() function.\n",p)
print("\nOutput is:\n", p.capitalize())
```

✓ 0.1s

```
String without using capitalize() function.
 python is easy language.

Output is:
 Python is easy language.
```

3) lower(): Converts all characters to lowercase.

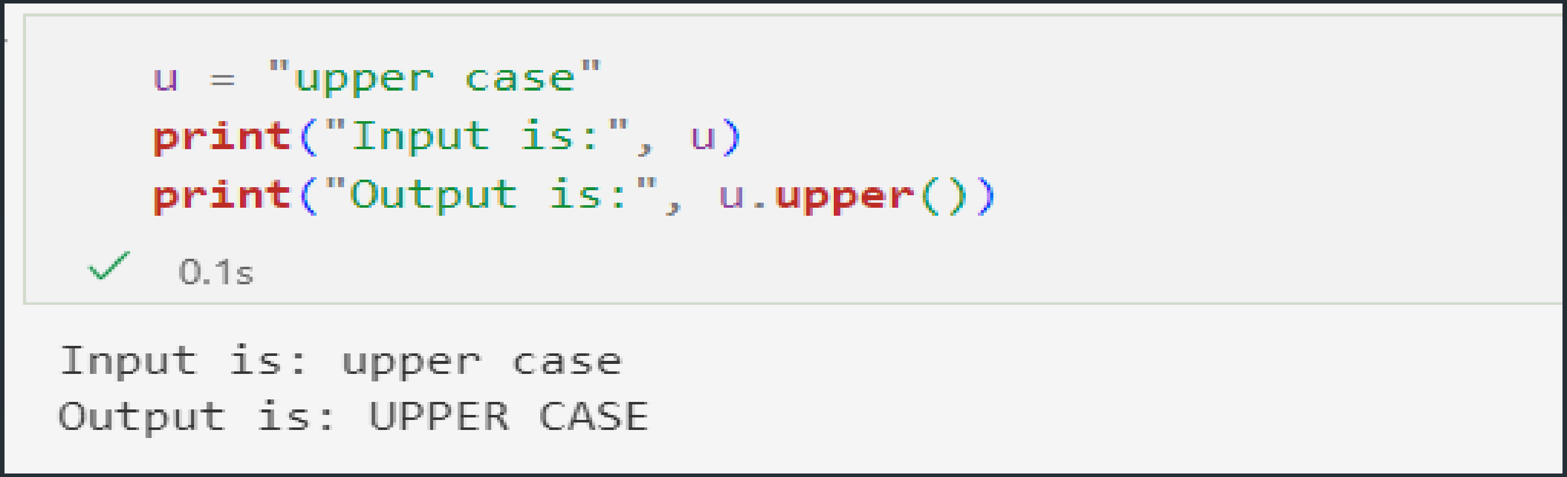


```
l = "LOWER CASE"
print("Input is:", l)
print("Output is:", l.lower())
```

✓ 0.3s

```
Input is: LOWER CASE
Output is: lower case
```

4) upper() : Converts all characters to uppercase.

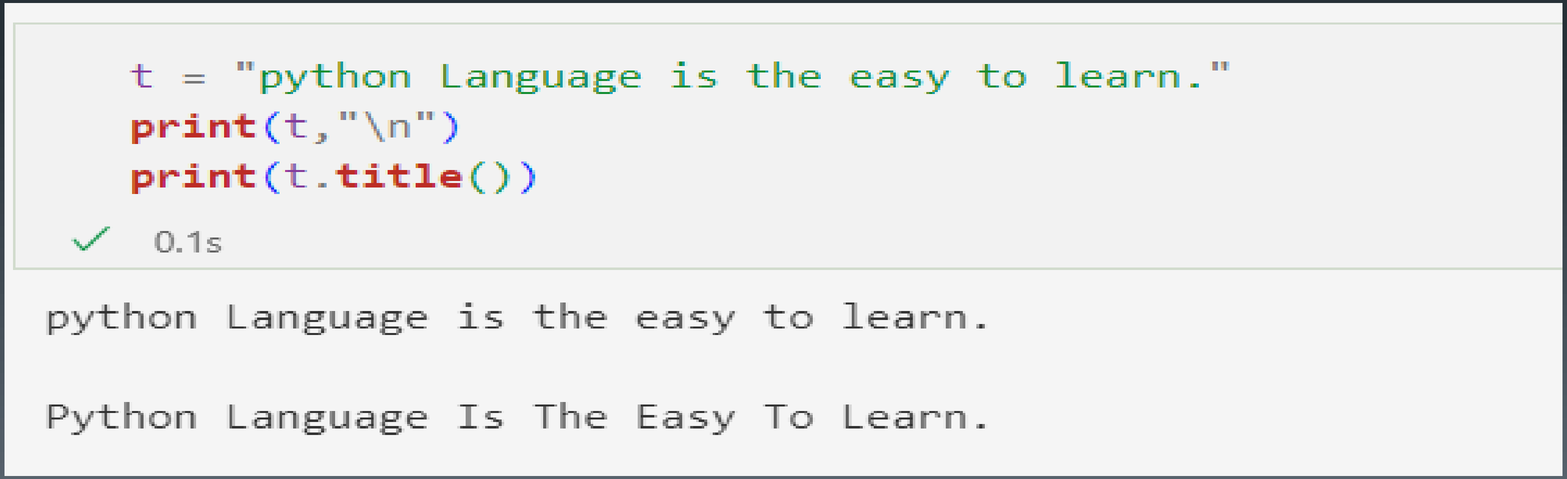


```
u = "upper case"
print("Input is:", u)
print("Output is:", u.upper())
```

✓ 0.1s

```
Input is: upper case
Output is: UPPER CASE
```

5) title(): Converts the first character of each word to uppercase.

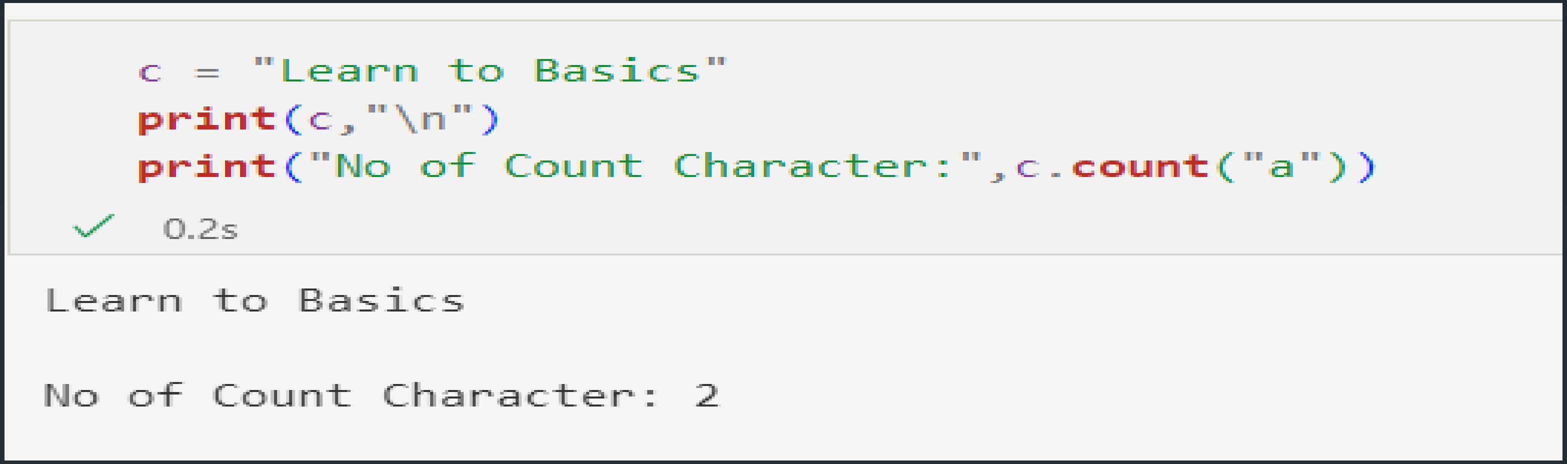


```
t = "python Language is the easy to learn."
print(t,"\n")
print(t.title())
```

✓ 0.1s

```
python Language is the easy to learn.

Python Language Is The Easy To Learn.
```

6) count() : Returns the number of occurrences of a substring.

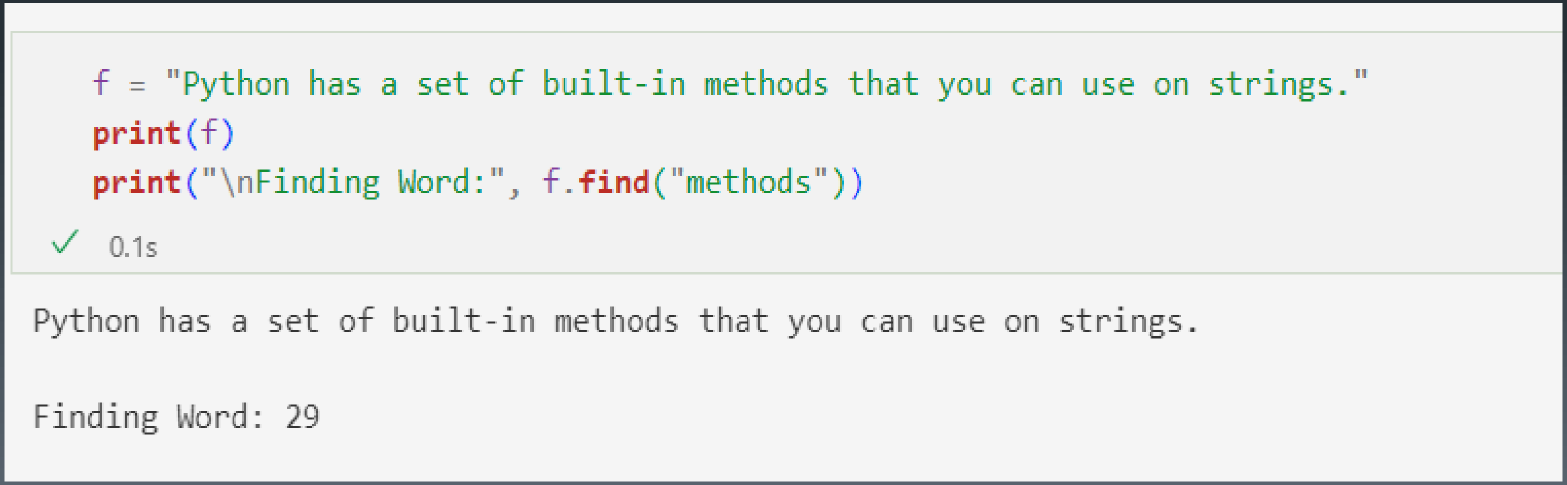


```
c = "Learn to Basics"
print(c,"\n")
print("No of Count Character:",c.count("a"))
```

✓ 0.2s

```
Learn to Basics

No of Count Character: 2
```

7) find(): Returns the index of the first occurrence of a substring (-1 if not found).

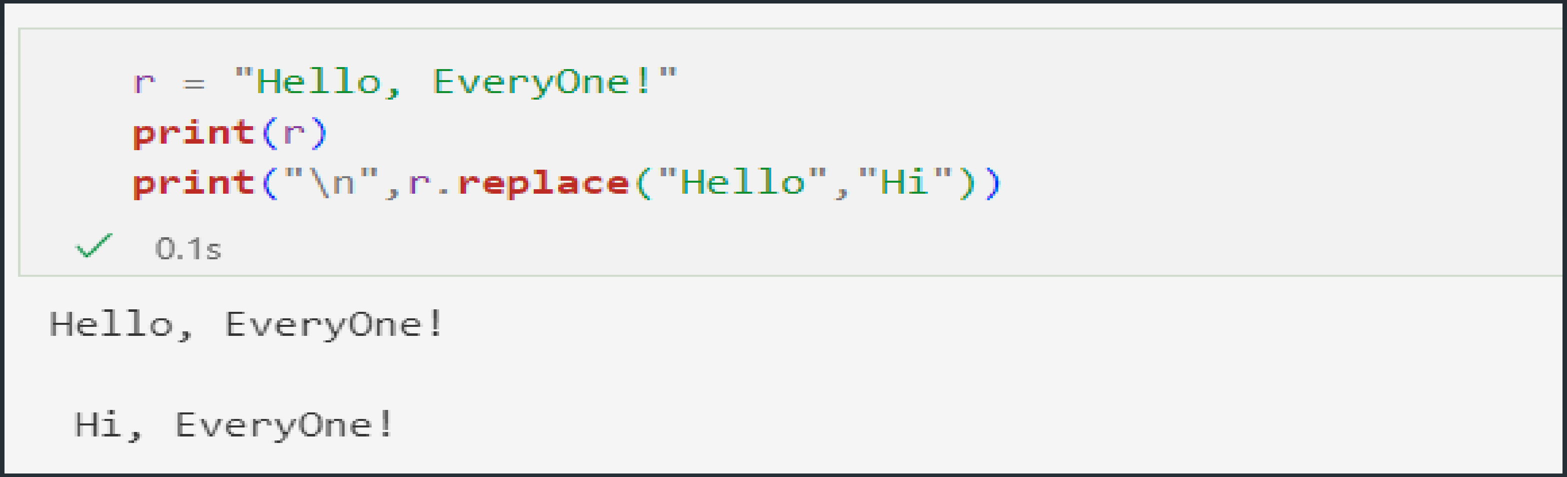


```
f = "Python has a set of built-in methods that you can use on strings."
print(f)
print("\nFinding Word:", f.find("methods"))
```

✓ 0.1s

```
Python has a set of built-in methods that you can use on strings.

Finding Word: 29
```

8) replace(): Replaces a substring with another substring.

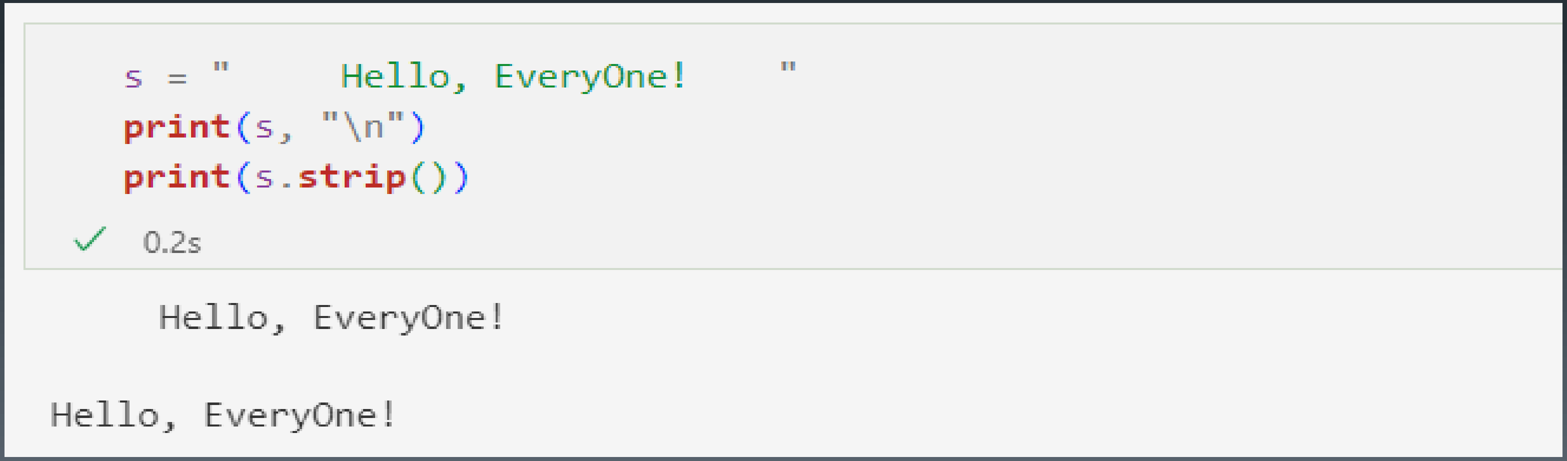


```
r = "Hello, EveryOne!"
print(r)
print("\n",r.replace("Hello","Hi"))
```

✓ 0.1s

```
Hello, EveryOne!

 Hi, EveryOne!
```

9) strip() : Removes leading and trailing whitespaces.

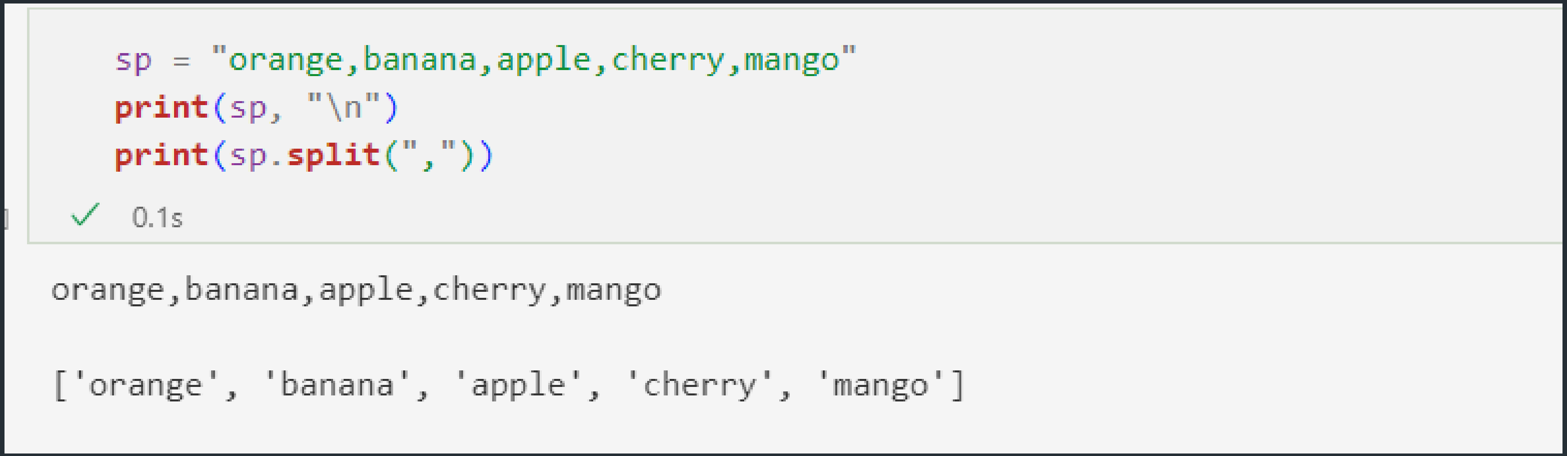


```
s = "      Hello, EveryOne!     "
print(s, "\n")
print(s.strip())
```

✓ 0.2s

```
      Hello, EveryOne!      

Hello, EveryOne!
```

10) split(): Splits the string into a list of substrings based on a delimiter.

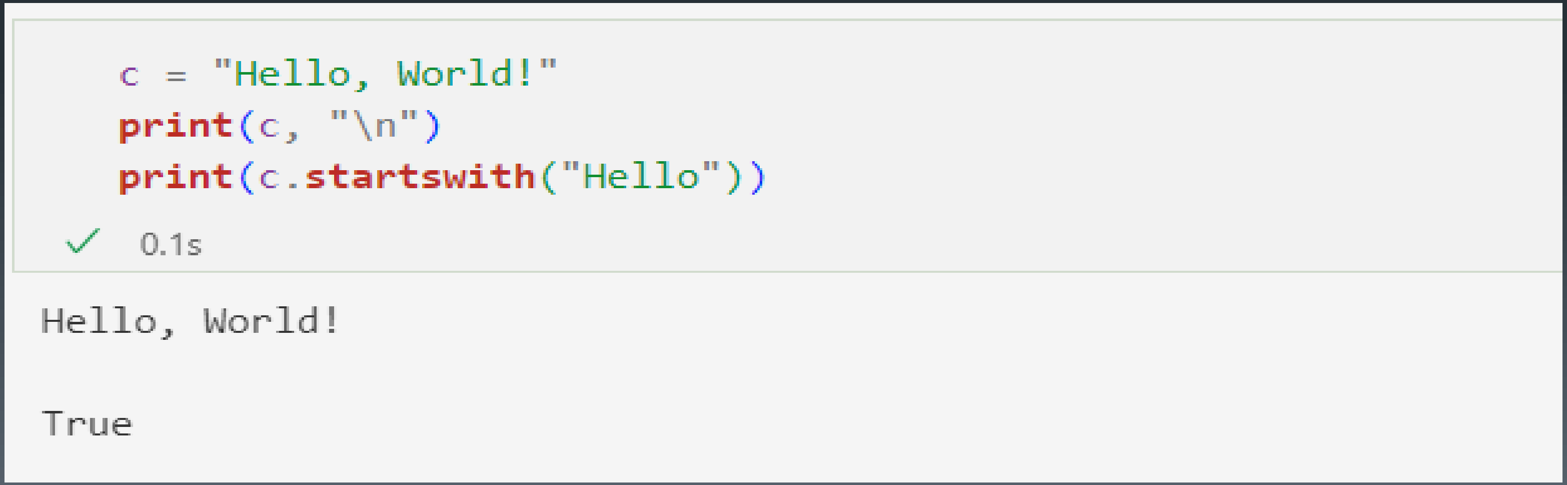


```
sp = "orange,banana,apple,cherry,mango"
print(sp, "\n")
print(sp.split(","))
```

✓ 0.1s

```
orange,banana,apple,cherry,mango

['orange', 'banana', 'apple', 'cherry', 'mango']
```

11) startswith() : Checks if the string starts with a specified prefix.

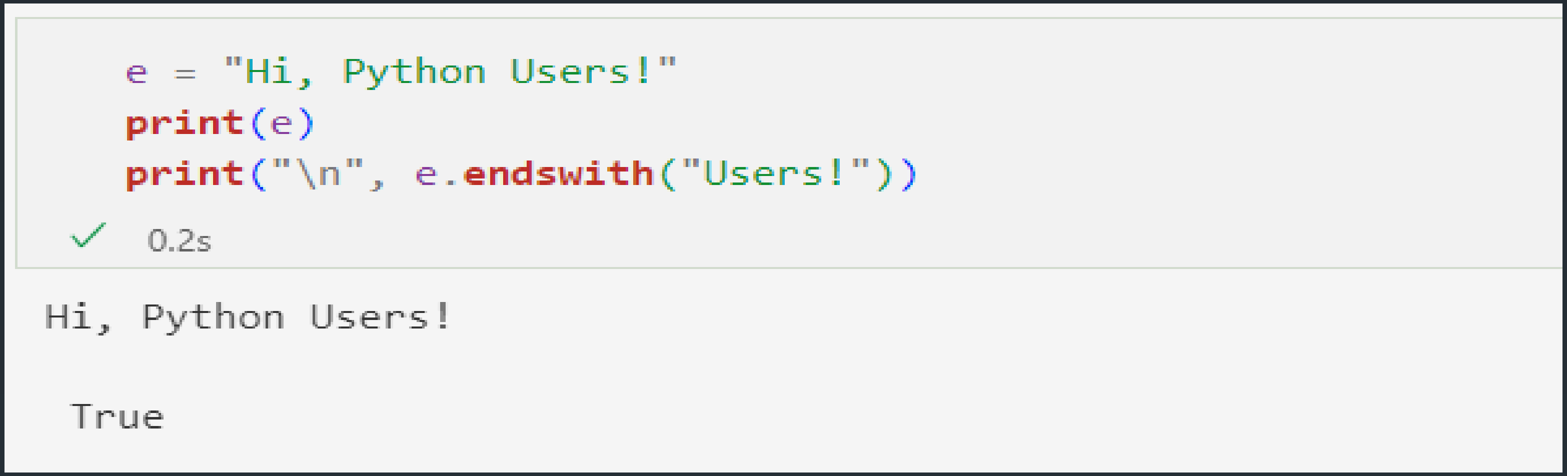


```
c = "Hello, World!"
print(c, "\n")
print(c.startswith("Hello"))
```

✓ 0.1s

```
Hello, World!

True
```

12) endswith(): Checks if the string ends with a specified suffix.

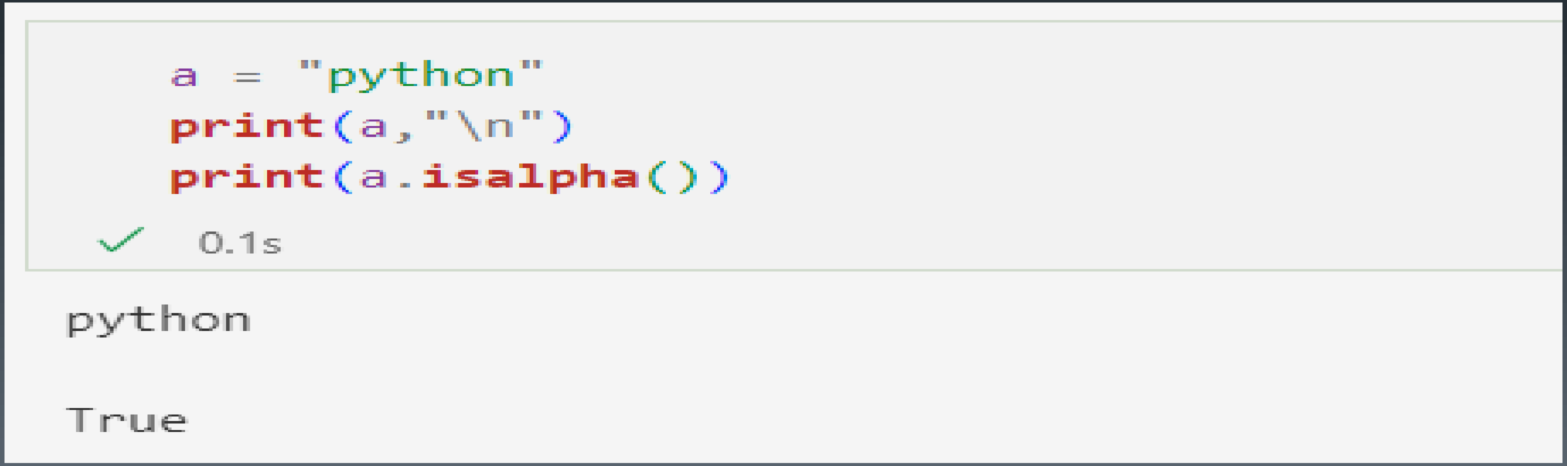


```
e = "Hi, Python Users!"
print(e)
print("\n", e.endswith("Users!"))
```

✓ 0.2s

```
Hi, Python Users!

 True
```

13) isalpha(): Returns True if all characters in the string are alphabetic.

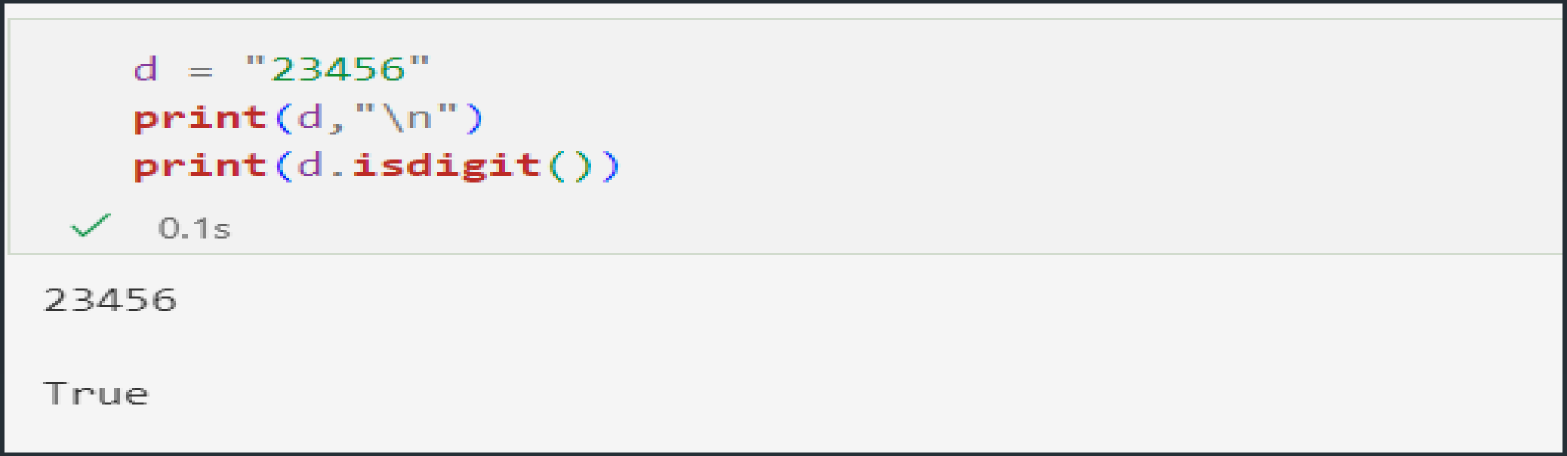


```
a = "python"
print(a,"\n")
print(a.isalpha())
```

✓ 0.1s

```
python

True
```

14) isdigit(): Returns True if all characters in the string are digits.

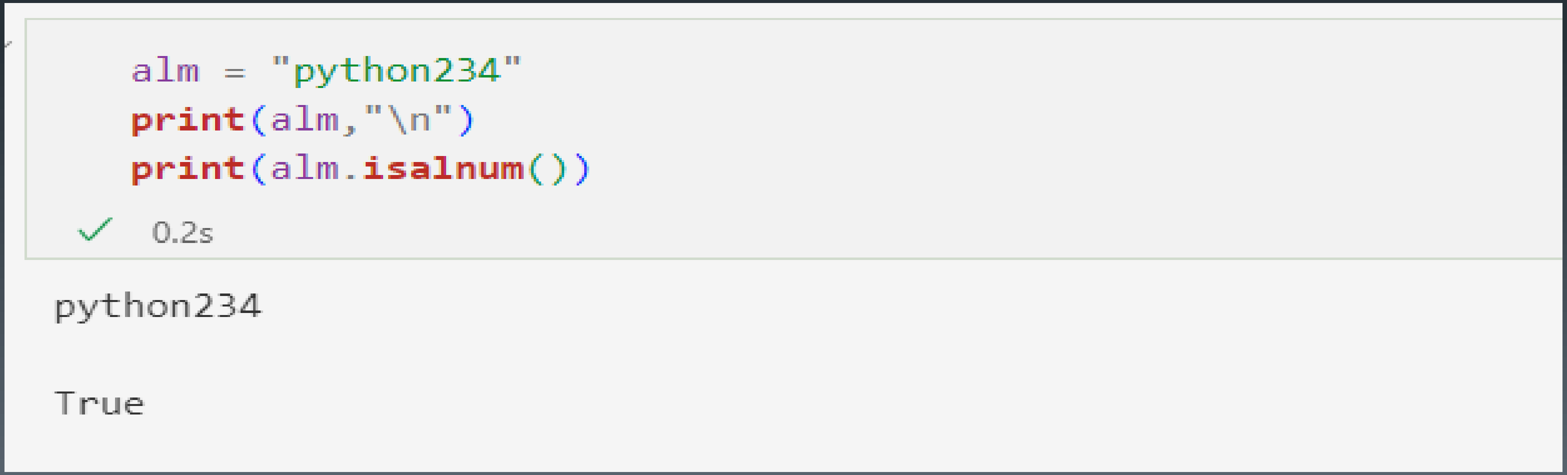


```
d = "23456"
print(d,"\n")
print(d.isdigit())
```

✓ 0.1s

```
23456

True
```

15) isalnum(): Returns True if all characters in the string are alphanumeric.

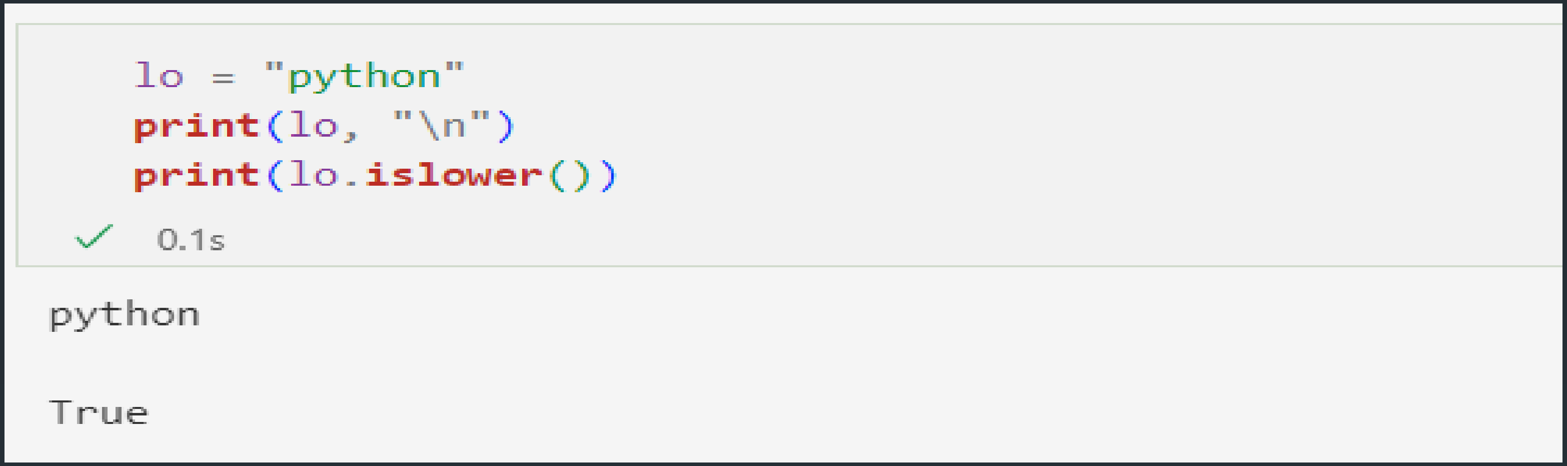


```
alm = "python234"
print(alm,"\n")
print(alm.isalnum())
```

✓ 0.2s

```
python234

True
```

16) islower(): Returns True if all alphabetic characters in the string are lowercase.

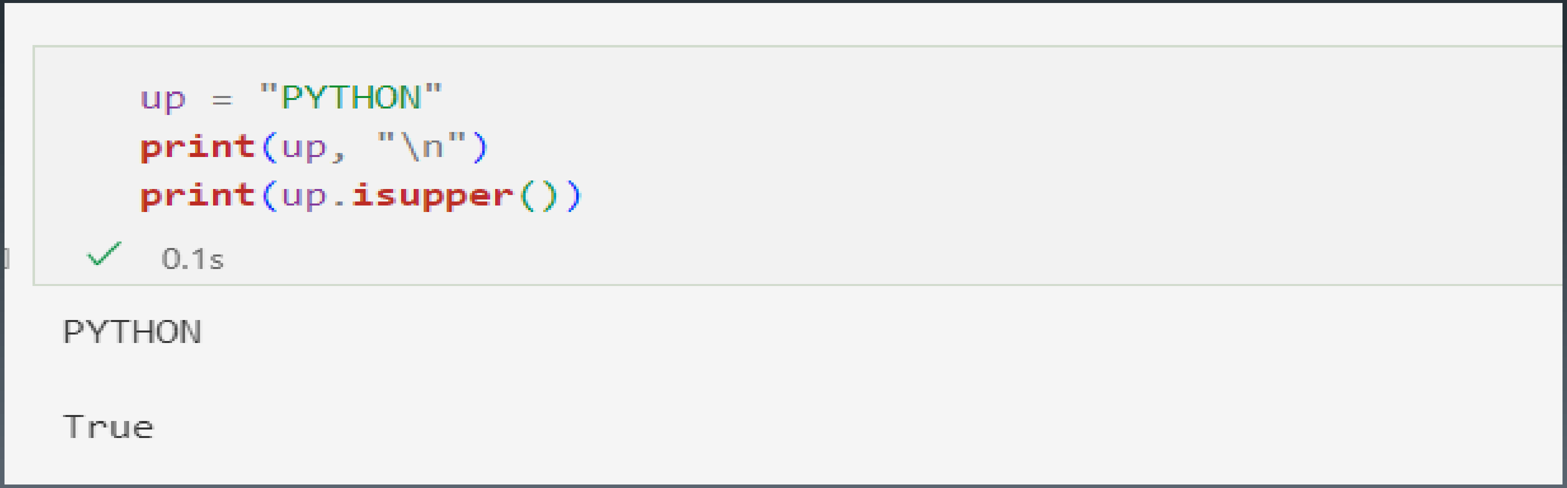


```
lo = "python"
print(lo, "\n")
print(lo.islower())
```

✓ 0.1s

```
python

True
```

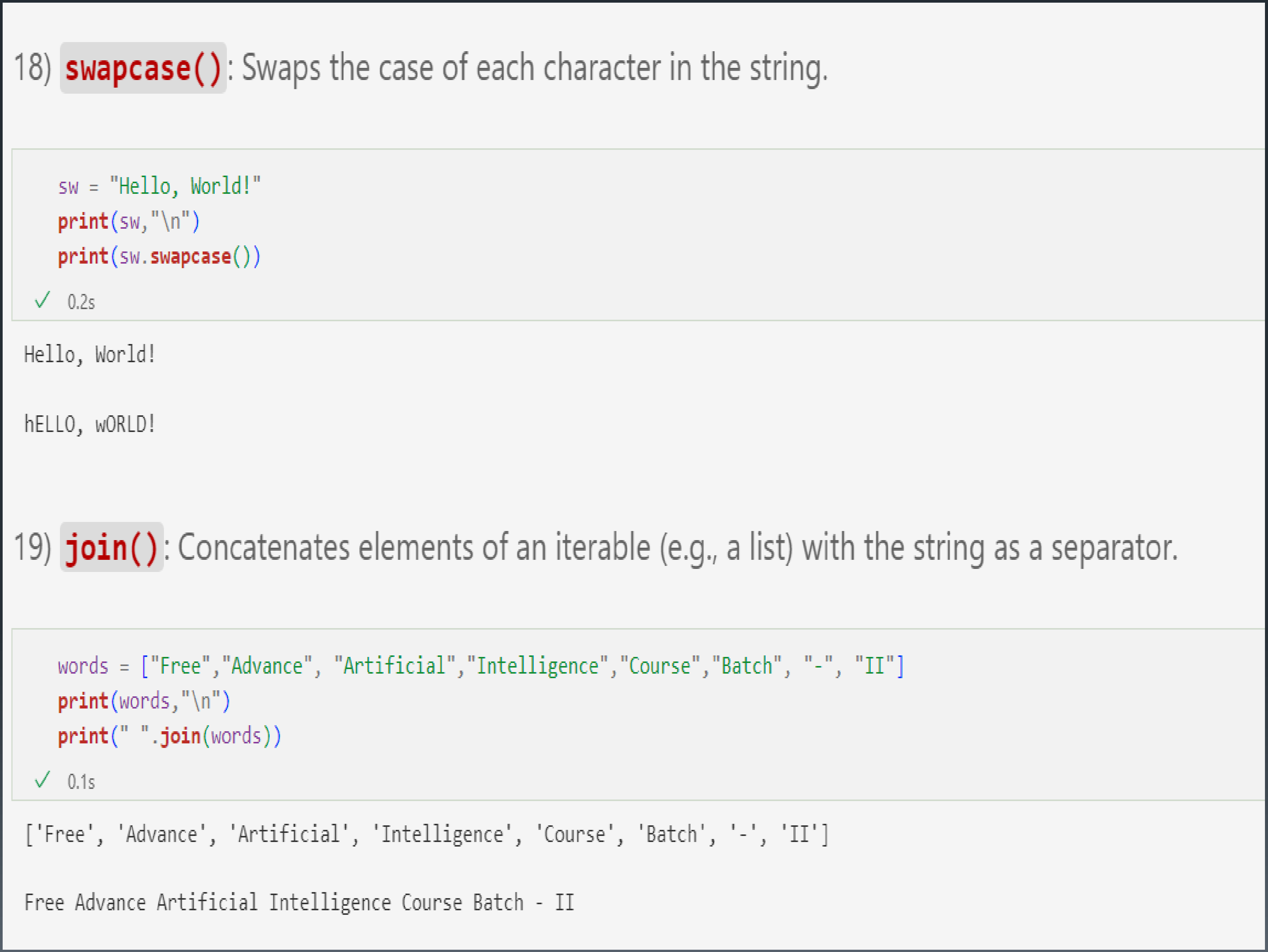
17) isupper(): Returns True if all alphabetic characters in the string are uppercase.



```
up = "PYTHON"
print(up, "\n")
print(up.isupper())
```

✓ 0.1s

```
PYTHON

True
```



18) `swapcase()`: Swaps the case of each character in the string.

```
sw = "Hello, World!"
print(sw,"\n")
print(sw.swapcase())
```

✓ 0.2s

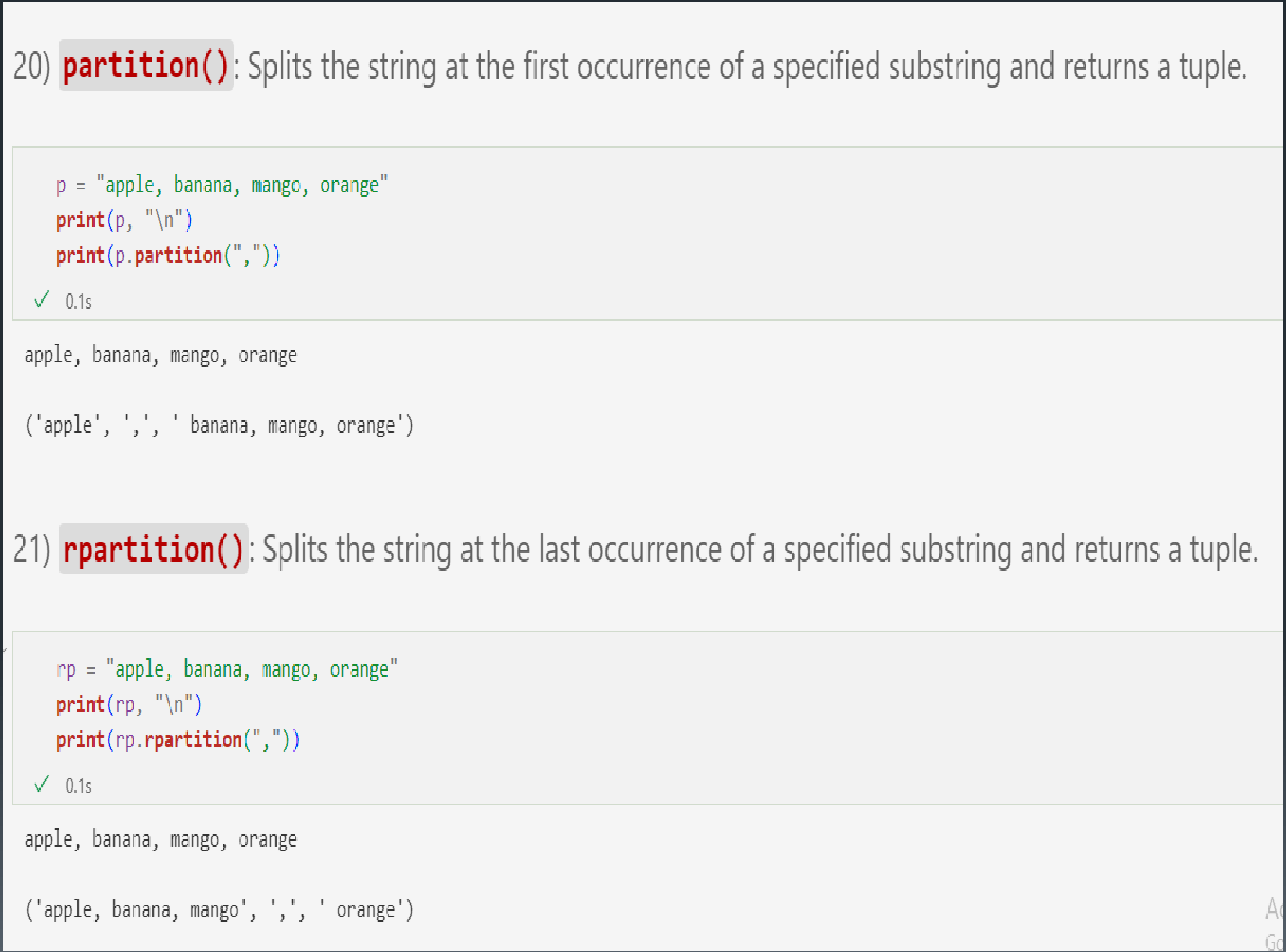
```
Hello, World!

hELLO, wORLD!
```

19) `join()`: Concatenates elements of an iterable (e.g., a list) with the string as a separator.

```
words = ["Free","Advance", "Artificial","Intelligence","Course","Batch", "-", "II"]
print(words,"\n")
print(" ".join(words))
```

✓ 0.1s

```
['Free', 'Advance', 'Artificial', 'Intelligence', 'Course', 'Batch', '-', 'II']

Free Advance Artificial Intelligence Course Batch - II
```



20) `partition()`: Splits the string at the first occurrence of a specified substring and returns a tuple.

```
p = "apple, banana, mango, orange"
print(p, "\n")
print(p.partition(","))
```

✓ 0.1s

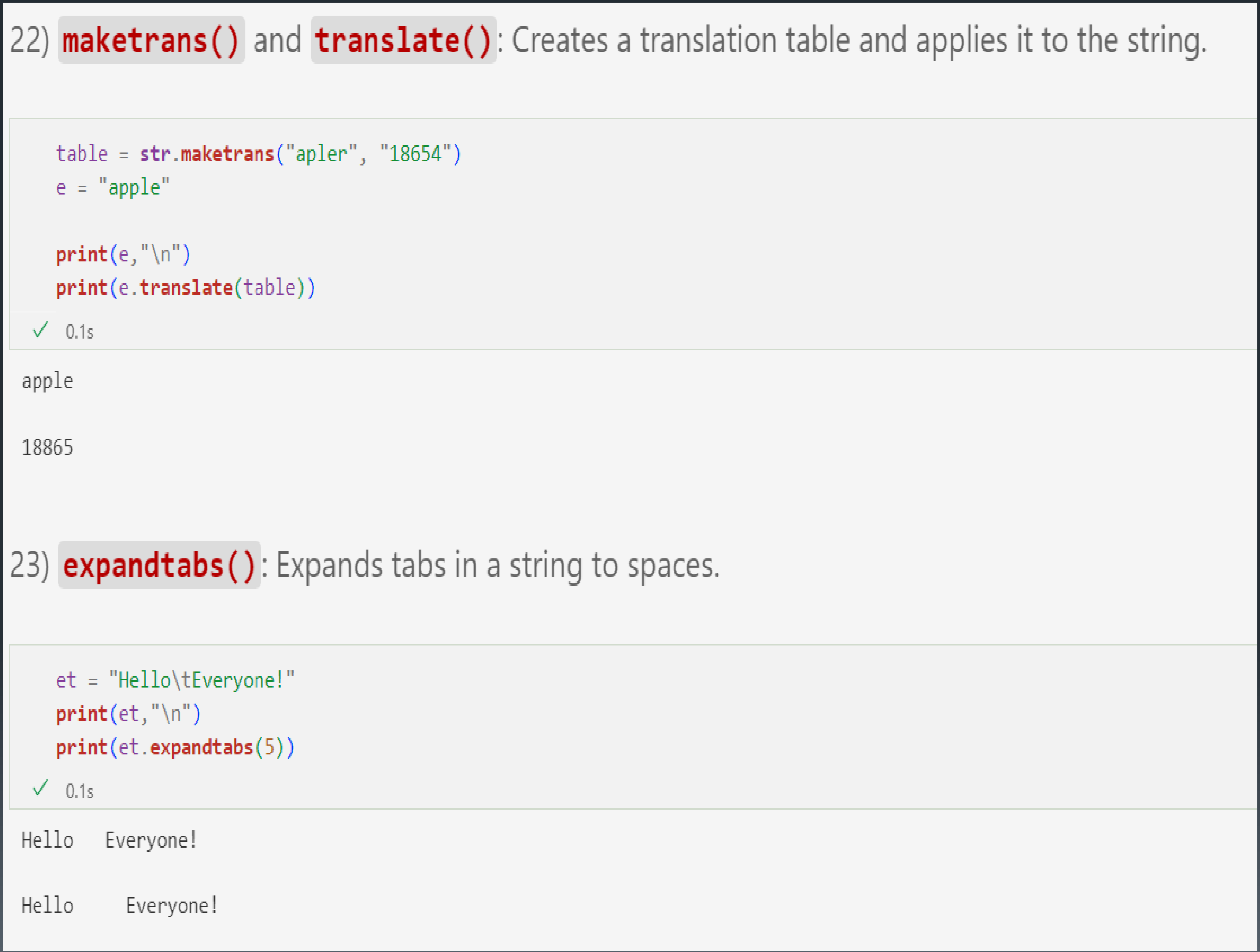
```
apple, banana, mango, orange

('apple', ',', ' banana, mango, orange')
```

21) `rpartition()`: Splits the string at the last occurrence of a specified substring and returns a tuple.

```
rp = "apple, banana, mango, orange"
print(rp, "\n")
print(rp.rpartition(","))
```

✓ 0.1s

```
apple, banana, mango, orange

('apple, banana, mango', ',', ' orange')
```



22) `maketrans()` and `translate()`: Creates a translation table and applies it to the string.

```python
table = str.maketrans("apler", "18654")
e = "apple"

print(e,"\n")
print(e.translate(table))
```

✓ 0.1s

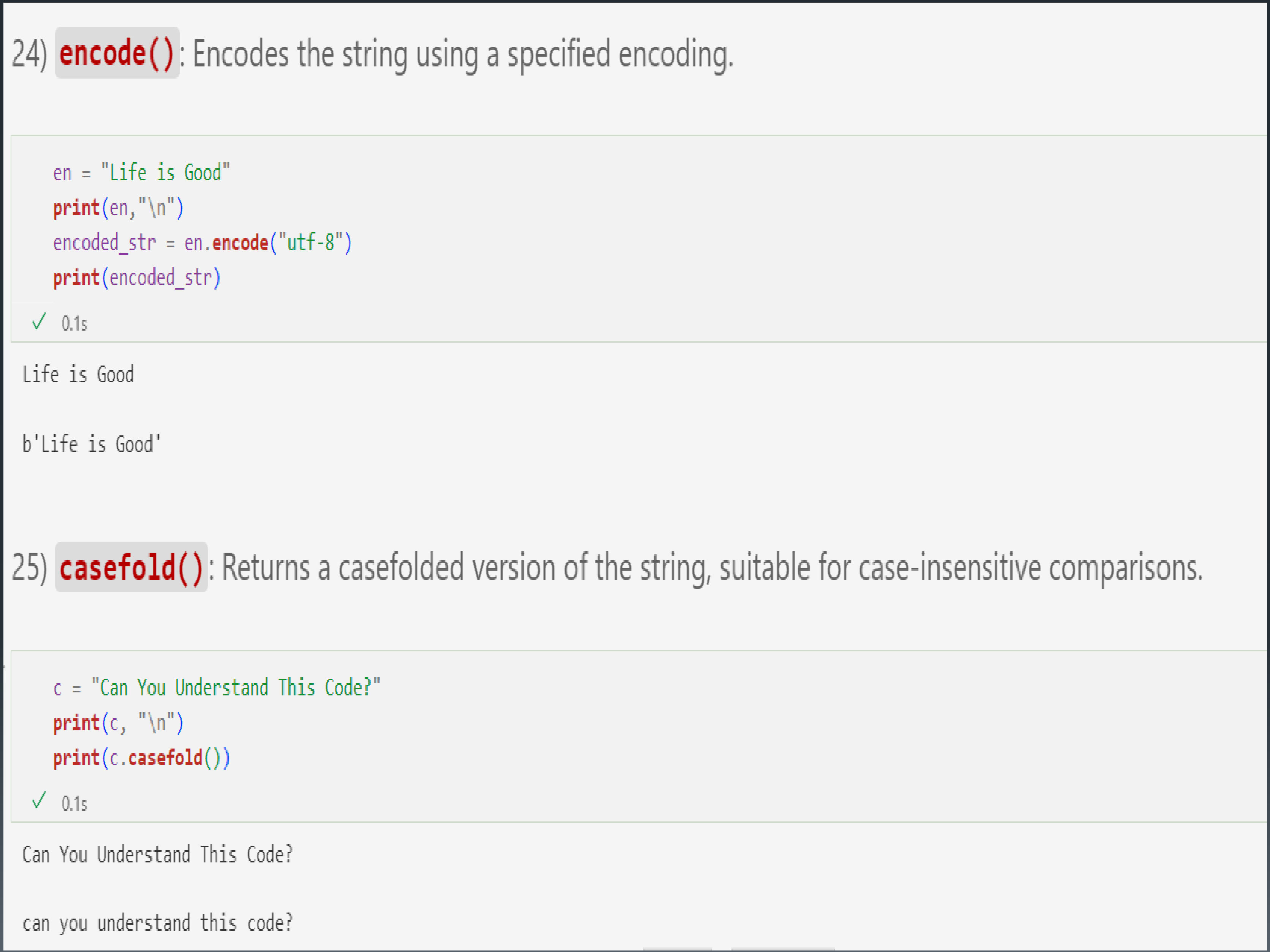
```
apple

18865
```

23) `expandtabs()`: Expands tabs in a string to spaces.

```python
et = "Hello\tEveryone!"
print(et,"\n")
print(et.expandtabs(5))
```

✓ 0.1s

```
Hello   Everyone!

Hello     Everyone!
```



24) **`encode()`**: Encodes the string using a specified encoding.

```python
en = "Life is Good"
print(en,"\n")
encoded_str = en.encode("utf-8")
print(encoded_str)
```

✓ 0.1s

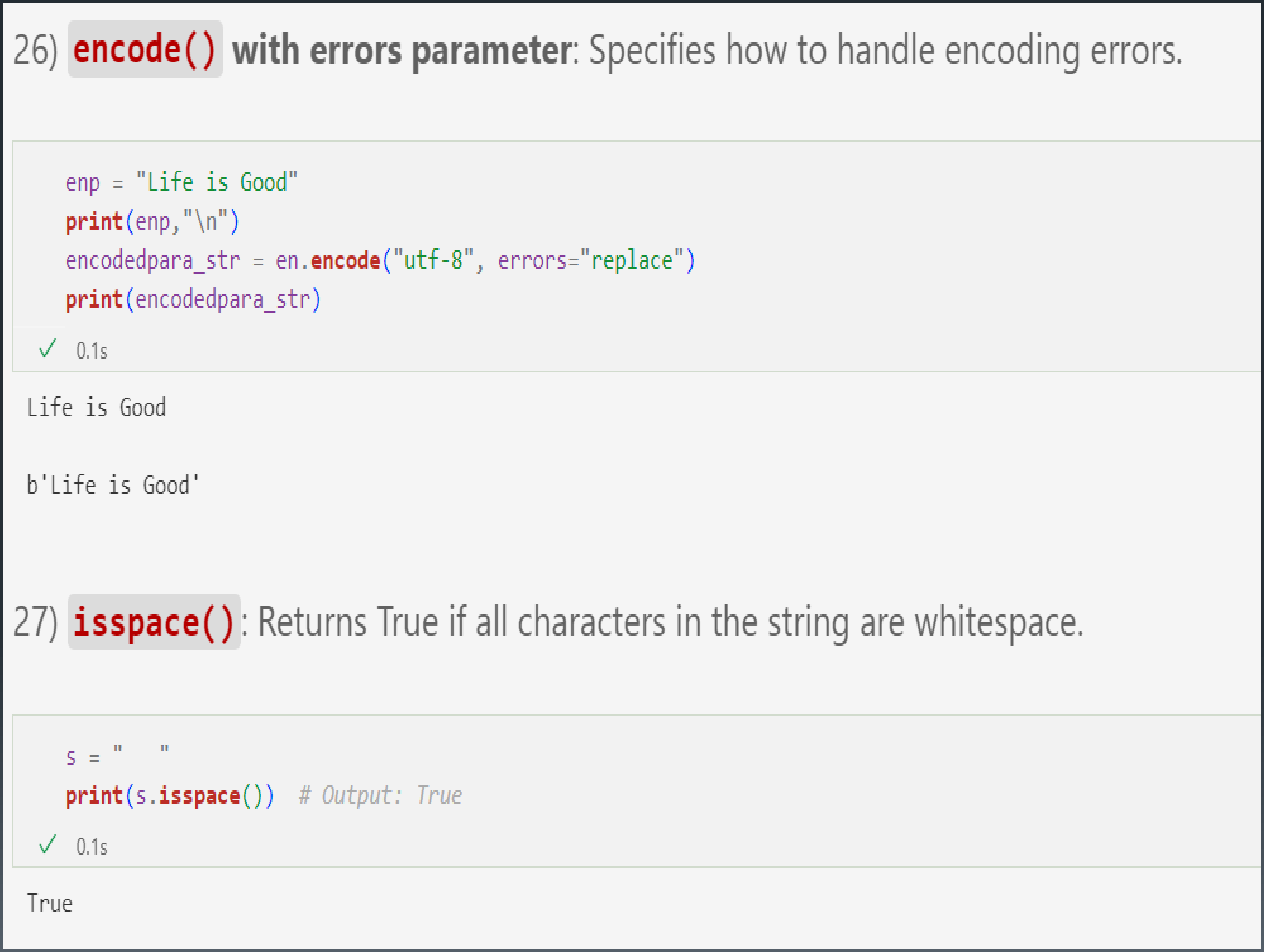
```
Life is Good

b'Life is Good'
```

25) **`casefold()`**: Returns a casefolded version of the string, suitable for case-insensitive comparisons.

```python
c = "Can You Understand This Code?"
print(c, "\n")
print(c.casefold())
```

✓ 0.1s

```
Can You Understand This Code?

can you understand this code?
```



26) **`encode()`** **with errors parameter**: Specifies how to handle encoding errors.

```python
enp = "Life is Good"
print(enp,"\n")
encodedpara_str = en.encode("utf-8", errors="replace")
print(encodedpara_str)
```

✓ 0.1s

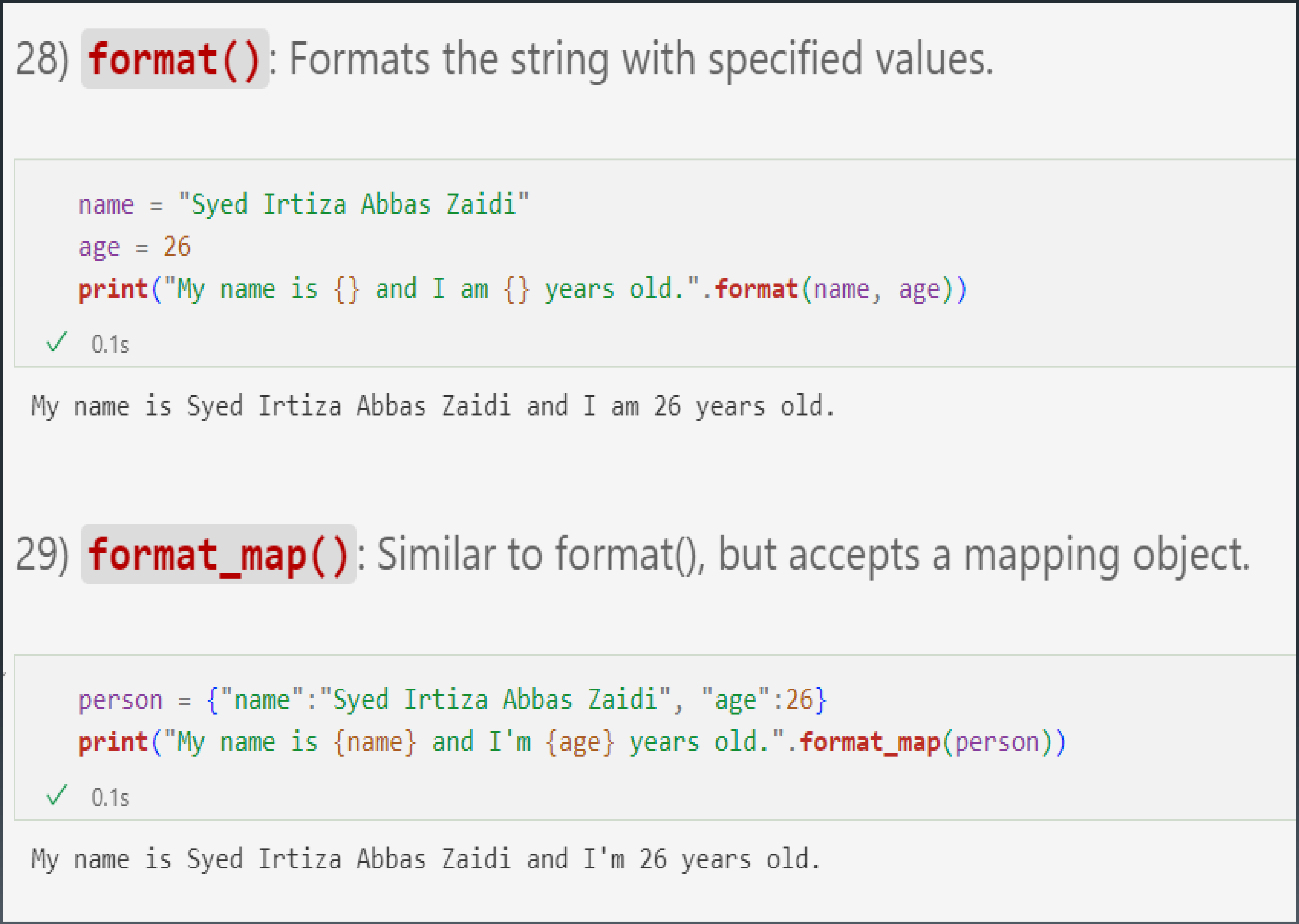
```
Life is Good

b'Life is Good'
```

27) **`isspace()`**: Returns True if all characters in the string are whitespace.

```python
s = "    "
print(s.isspace())  # Output: True
```

✓ 0.1s

```
True
```



28) **`format()`**: Formats the string with specified values.

```python
name = "Syed Irtiza Abbas Zaidi"
age = 26
print("My name is {} and I am {} years old.".format(name, age))
```

✓ 0.1s

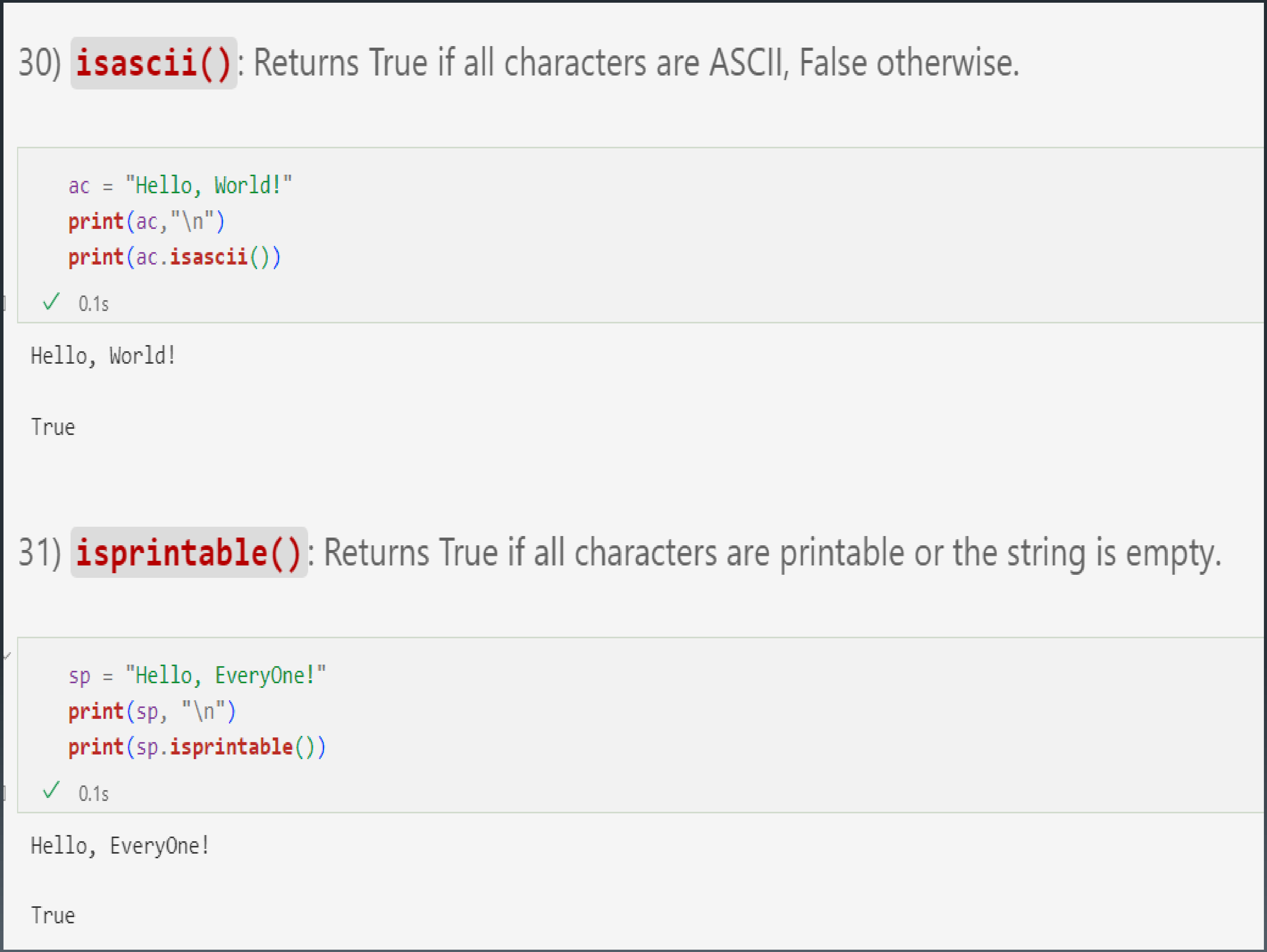
```
My name is Syed Irtiza Abbas Zaidi and I am 26 years old.
```

29) **`format_map()`**: Similar to format(), but accepts a mapping object.

```python
person = {"name":"Syed Irtiza Abbas Zaidi", "age":26}
print("My name is {name} and I'm {age} years old.".format_map(person))
```

✓ 0.1s

```
My name is Syed Irtiza Abbas Zaidi and I'm 26 years old.
```



30) `isascii()`: Returns True if all characters are ASCII, False otherwise.

```python
ac = "Hello, World!"
print(ac,"\n")
print(ac.isascii())
```

✓ 0.1s

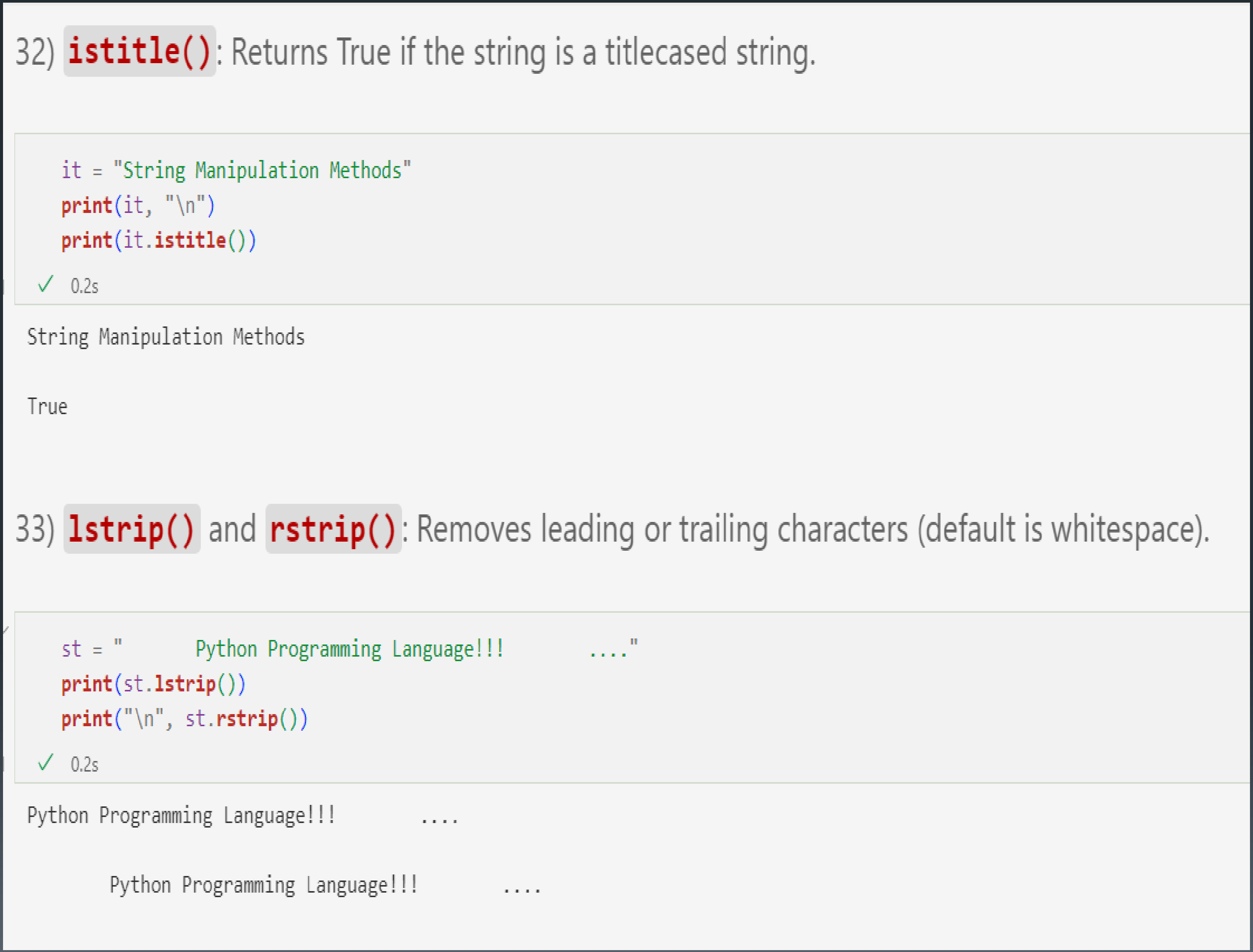
```
Hello, World!

True
```

31) `isprintable()`: Returns True if all characters are printable or the string is empty.

```python
sp = "Hello, EveryOne!"
print(sp, "\n")
print(sp.isprintable())
```

✓ 0.1s

```
Hello, EveryOne!

True
```



32) **istitle()**: Returns True if the string is a titlecased string.

```
it = "String Manipulation Methods"
print(it, "\n")
print(it.istitle())
```

✓ 0.2s

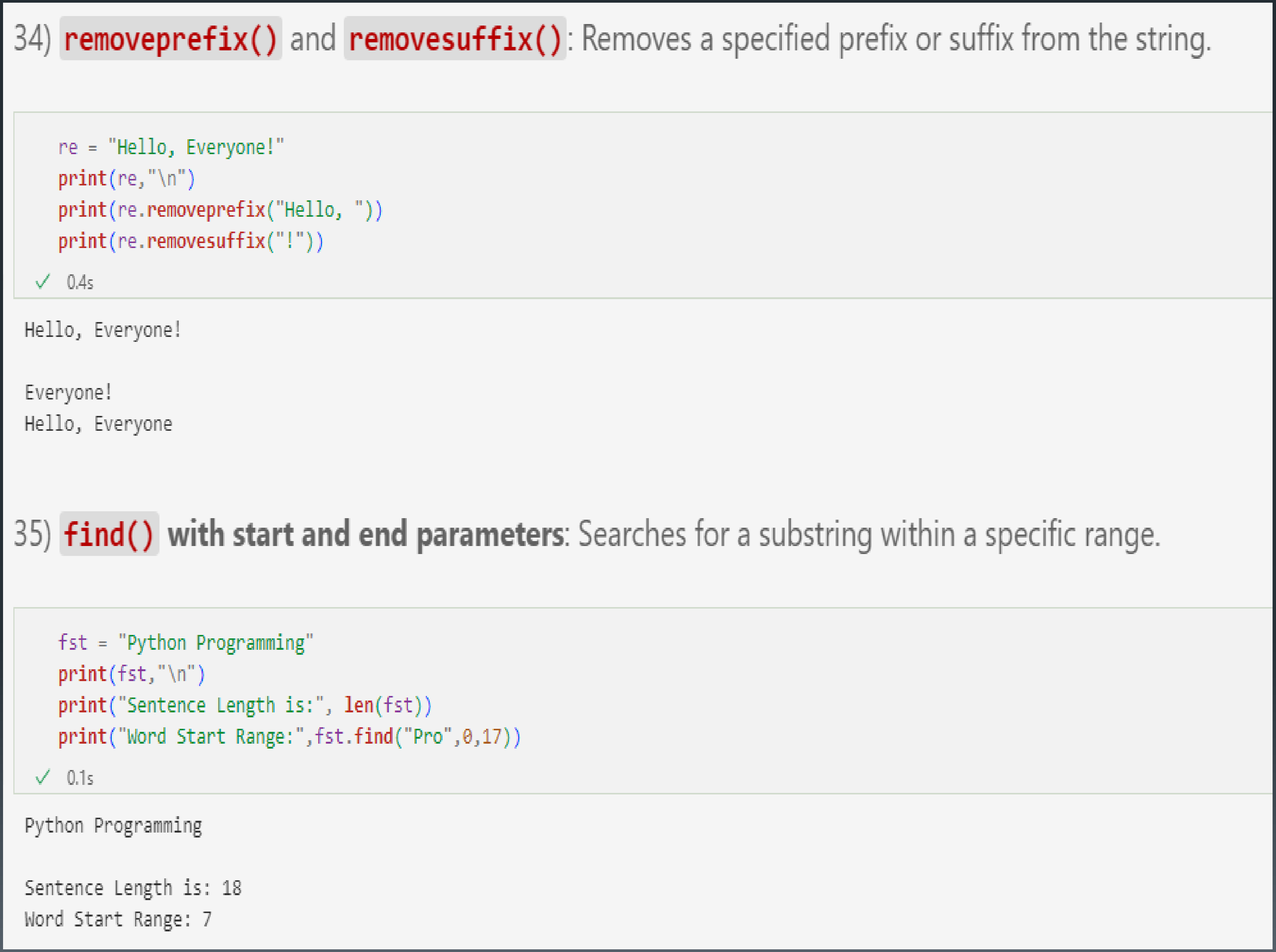
```
String Manipulation Methods

True
```

33) **lstrip()** and **rstrip()**: Removes leading or trailing characters (default is whitespace).

```
st = "       Python Programming Language!!!        ...."
print(st.lstrip())
print("\n", st.rstrip())
```

✓ 0.2s

```
Python Programming Language!!!        ....

        Python Programming Language!!!        ....
```



34) `removeprefix()` and `removesuffix()`: Removes a specified prefix or suffix from the string.

```python
re = "Hello, Everyone!"
print(re,"\n")
print(re.removeprefix("Hello, "))
print(re.removesuffix("!"))
```

✓ 0.4s

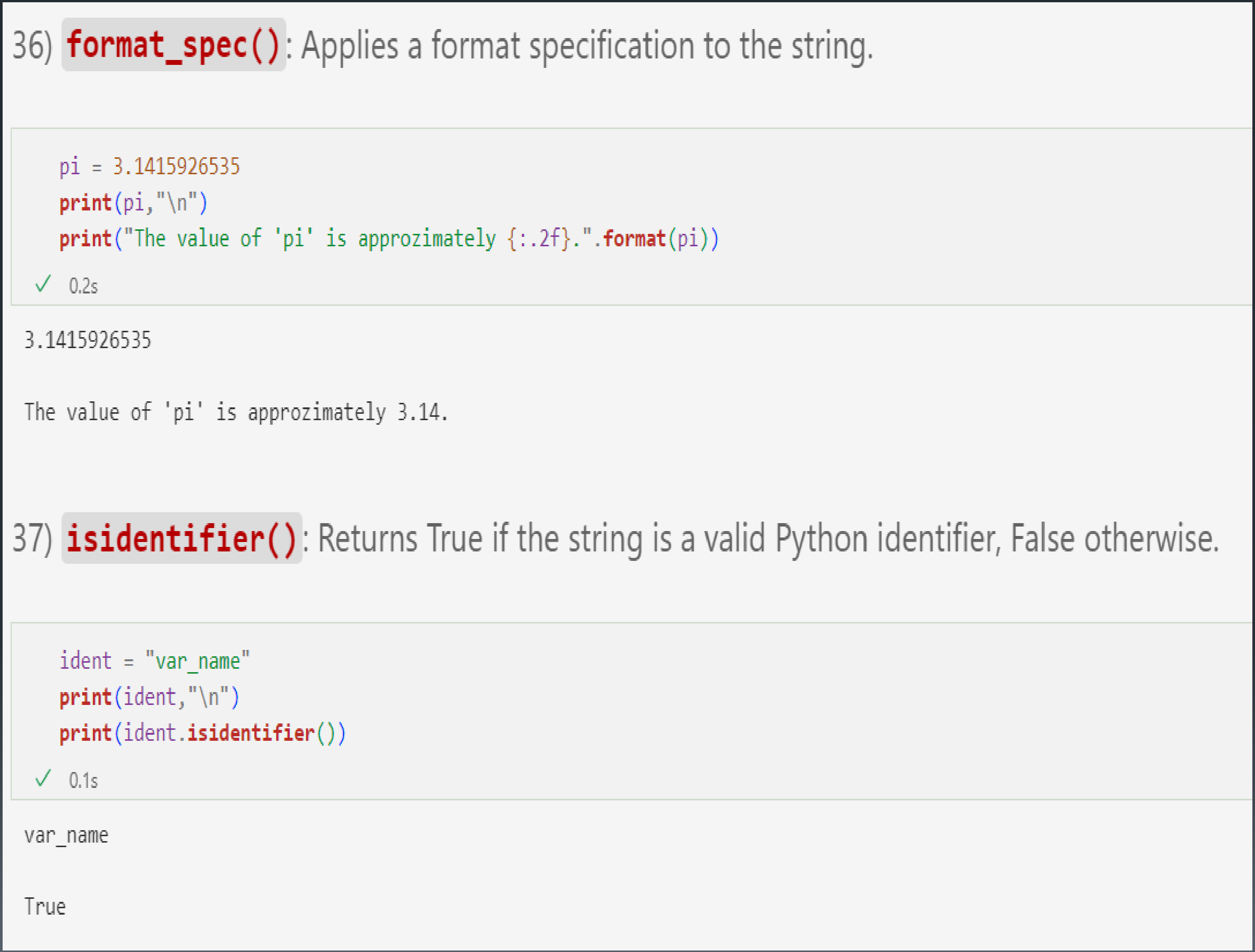
```
Hello, Everyone!

Everyone!
Hello, Everyone
```

35) `find()` **with start and end parameters**: Searches for a substring within a specific range.

```python
fst = "Python Programming"
print(fst,"\n")
print("Sentence Length is:", len(fst))
print("Word Start Range:",fst.find("Pro",0,17))
```

✓ 0.1s

```
Python Programming

Sentence Length is: 18
Word Start Range: 7
```



36) **format_spec()**: Applies a format specification to the string.

```
pi = 3.1415926535
print(pi,"\n")
print("The value of 'pi' is approzimately {:.2f}.".format(pi))
```

✓ 0.2s

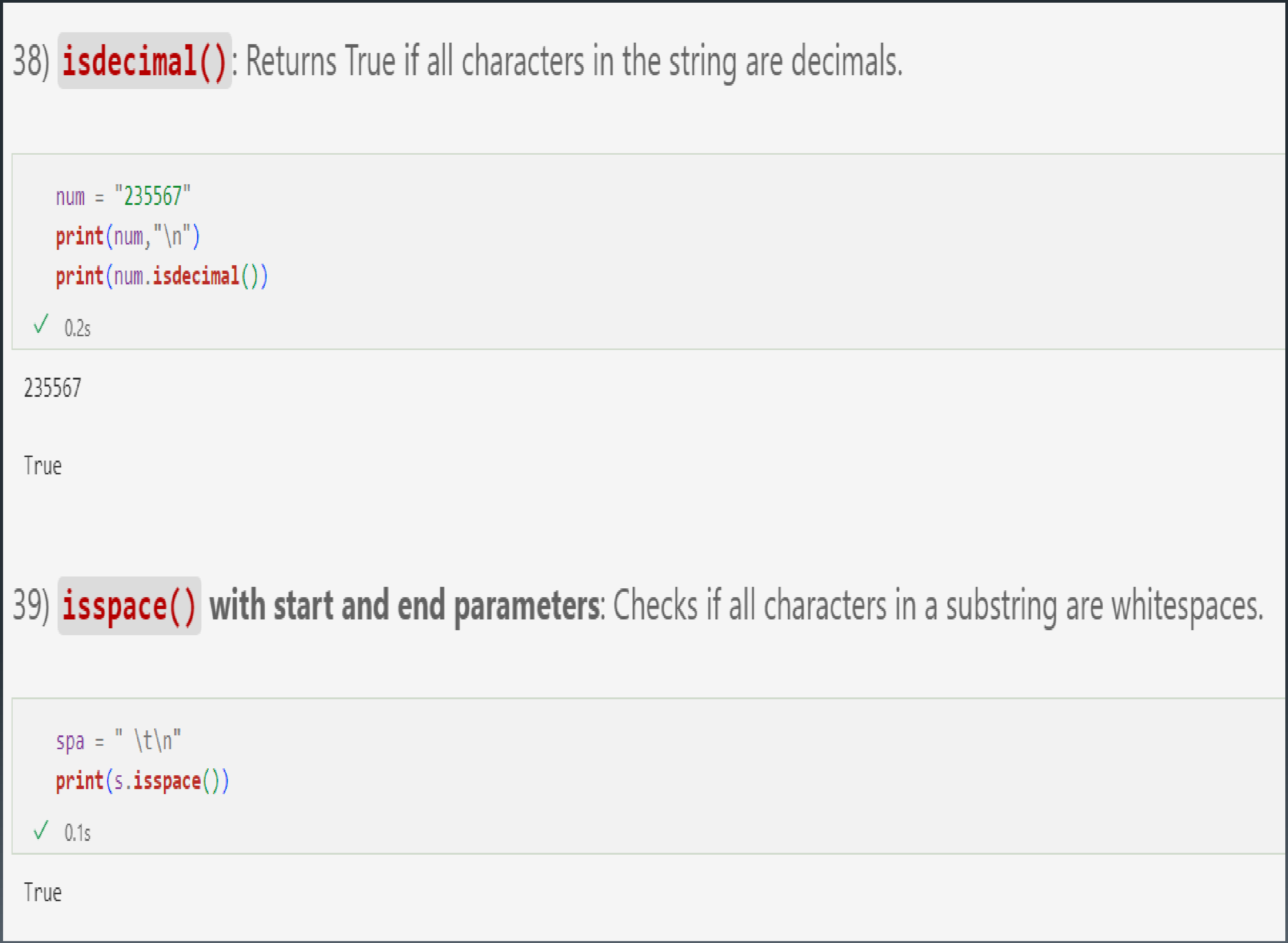
```
3.1415926535

The value of 'pi' is approzimately 3.14.
```

37) **isidentifier()**: Returns True if the string is a valid Python identifier, False otherwise.

```
ident = "var_name"
print(ident,"\n")
print(ident.isidentifier())
```

✓ 0.1s

```
var_name

True
```



38) `isdecimal()`: Returns True if all characters in the string are decimals.

```python
num = "235567"
print(num,"\n")
print(num.isdecimal())
```

✓ 0.2s

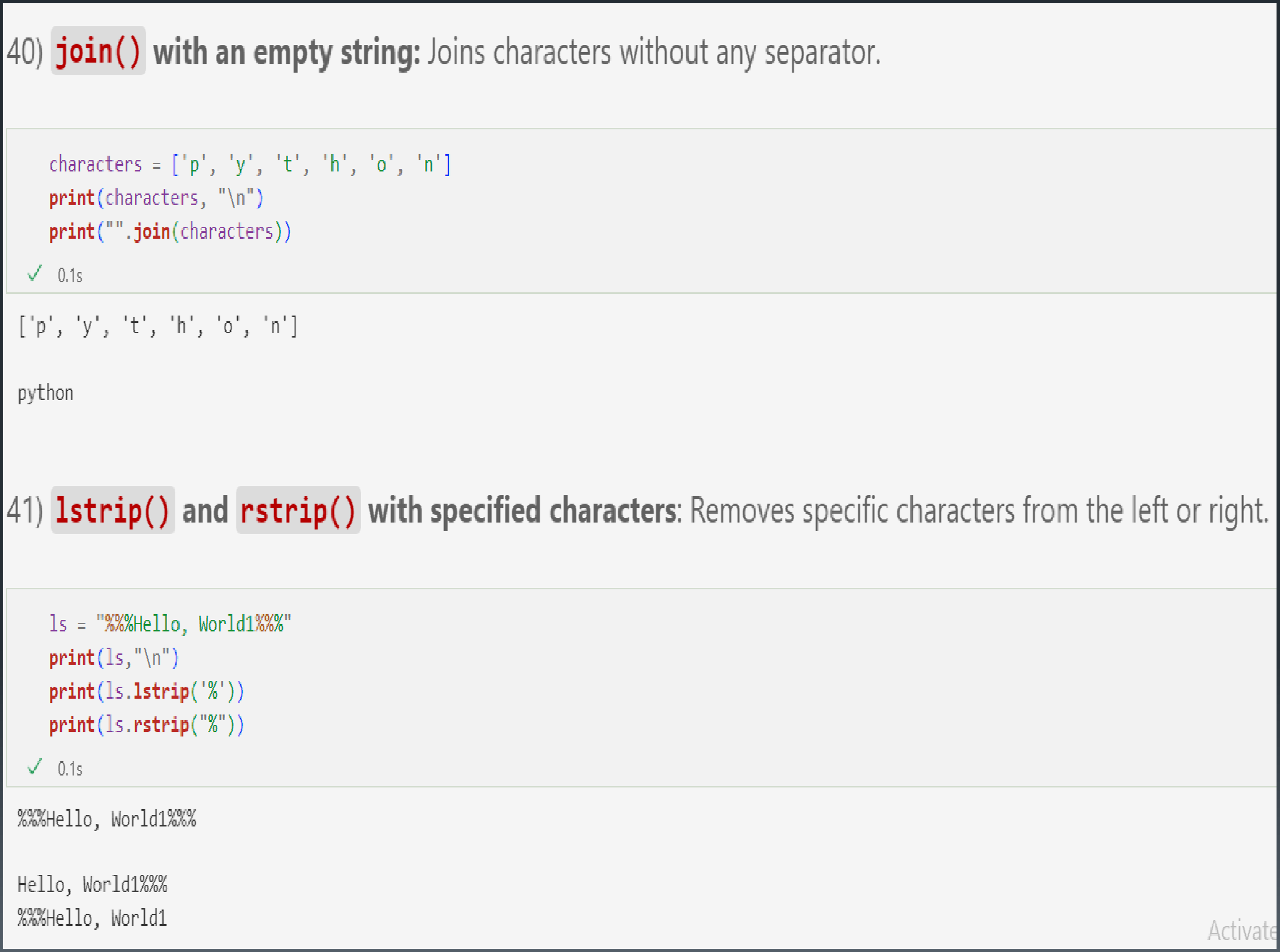
```
235567

True
```

39) `isspace()` **with start and end parameters**: Checks if all characters in a substring are whitespaces.

```python
spa = " \t\n"
print(s.isspace())
```

✓ 0.1s

```
True
```



40) `join()` **with an empty string:** Joins characters without any separator.

```
characters = ['p', 'y', 't', 'h', 'o', 'n']
print(characters, "\n")
print("".join(characters))
```

✓ 0.1s

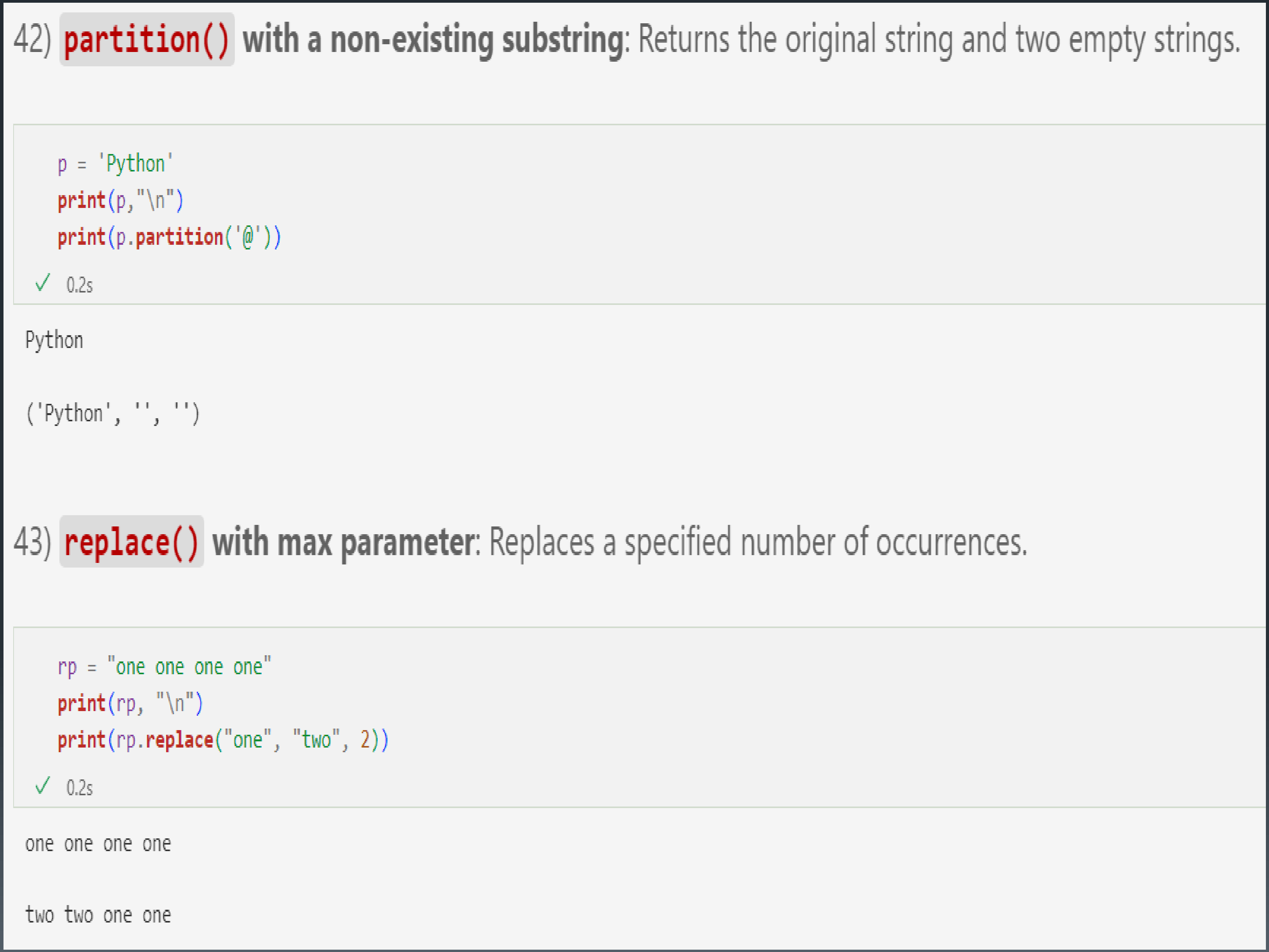
```
['p', 'y', 't', 'h', 'o', 'n'] 

python
```

41) `lstrip()` **and** `rstrip()` **with specified characters**: Removes specific characters from the left or right.

```
ls = "%%%Hello, World1%%%"
print(ls,"\n")
print(ls.lstrip('%'))
print(ls.rstrip("%"))
```

✓ 0.1s

```
%%%Hello, World1%%% 

Hello, World1%%%
%%%Hello, World1
```



42) `partition()` **with a non-existing substring**: Returns the original string and two empty strings.

```python
p = 'Python'
print(p,"\n")
print(p.partition('@'))
```

✓ 0.2s

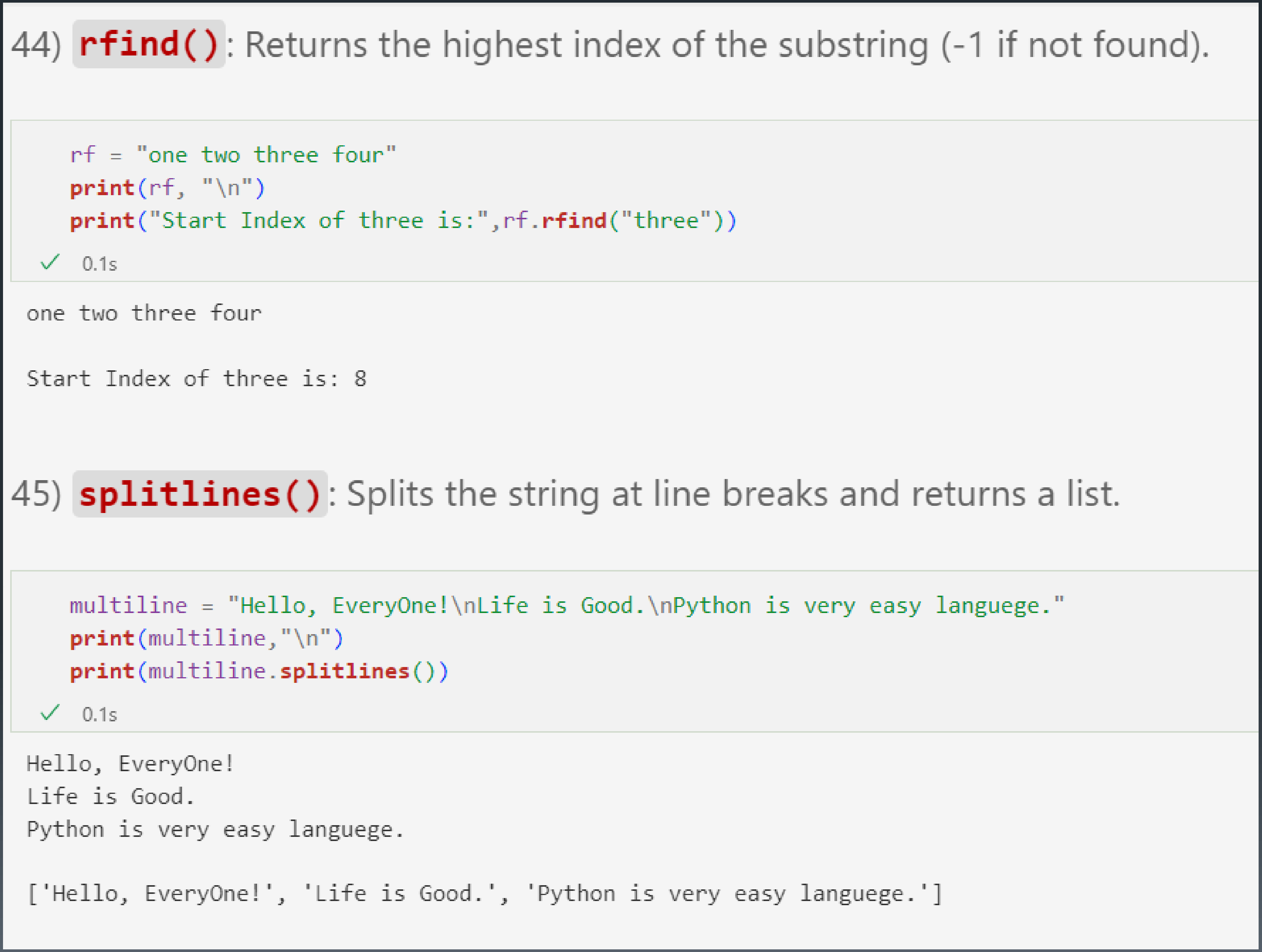
```
Python

('Python', '', '')
```

43) `replace()` **with max parameter**: Replaces a specified number of occurrences.

```python
rp = "one one one one"
print(rp, "\n")
print(rp.replace("one", "two", 2))
```

✓ 0.2s

```
one one one one

two two one one
```



44) `rfind()`: Returns the highest index of the substring (-1 if not found).

```python
rf = "one two three four"
print(rf, "\n")
print("Start Index of three is:",rf.rfind("three"))
```

✓ 0.1s

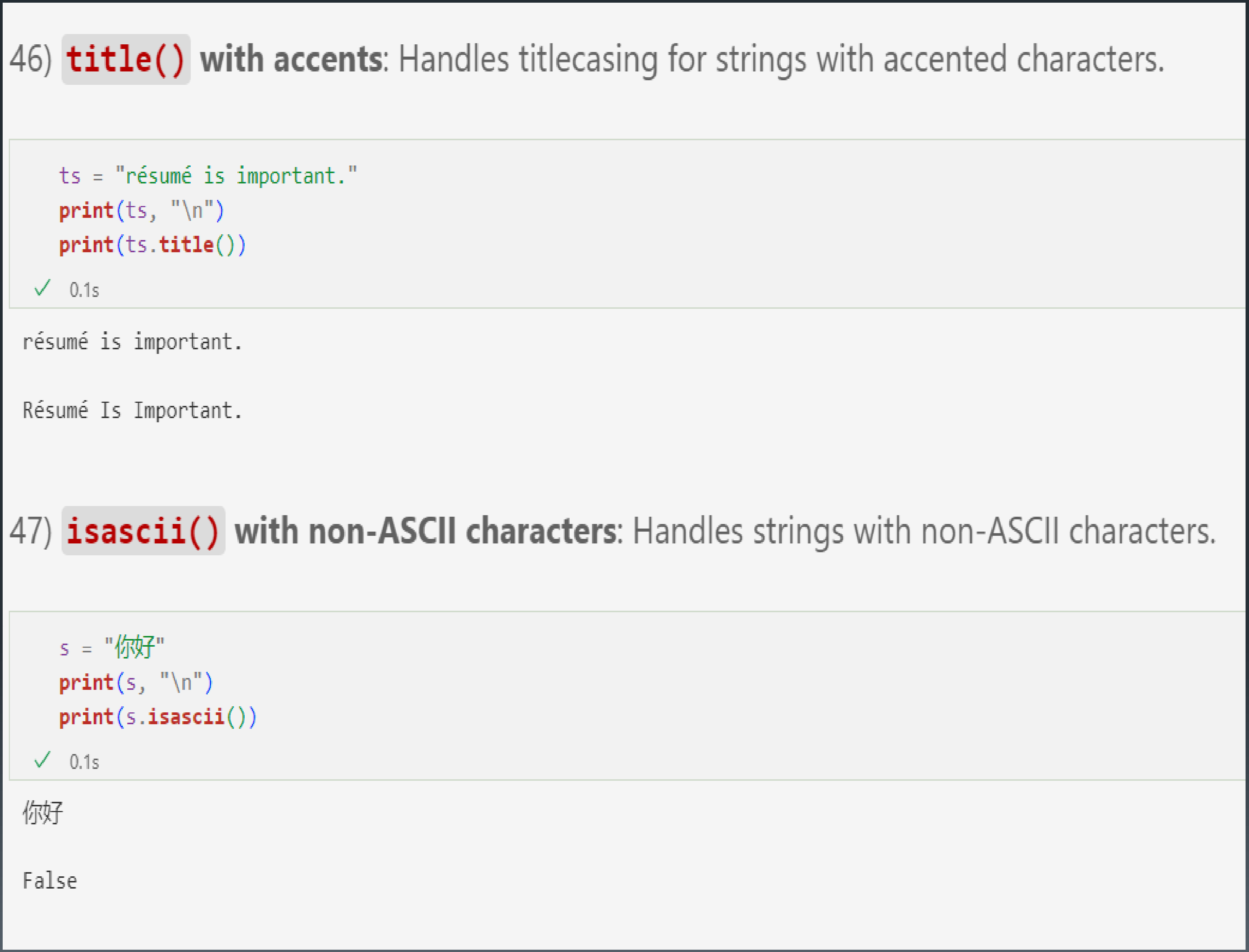
```
one two three four

Start Index of three is: 8
```

45) `splitlines()`: Splits the string at line breaks and returns a list.

```python
multiline = "Hello, EveryOne!\nLife is Good.\nPython is very easy languege."
print(multiline,"\n")
print(multiline.splitlines())
```

✓ 0.1s

```
Hello, EveryOne!
Life is Good.
Python is very easy languege.

['Hello, EveryOne!', 'Life is Good.', 'Python is very easy languege.']
```



46) `title()` **with accents**: Handles titlecasing for strings with accented characters.

```python
ts = "résumé is important."
print(ts, "\n")
print(ts.title())
```

✓ 0.1s

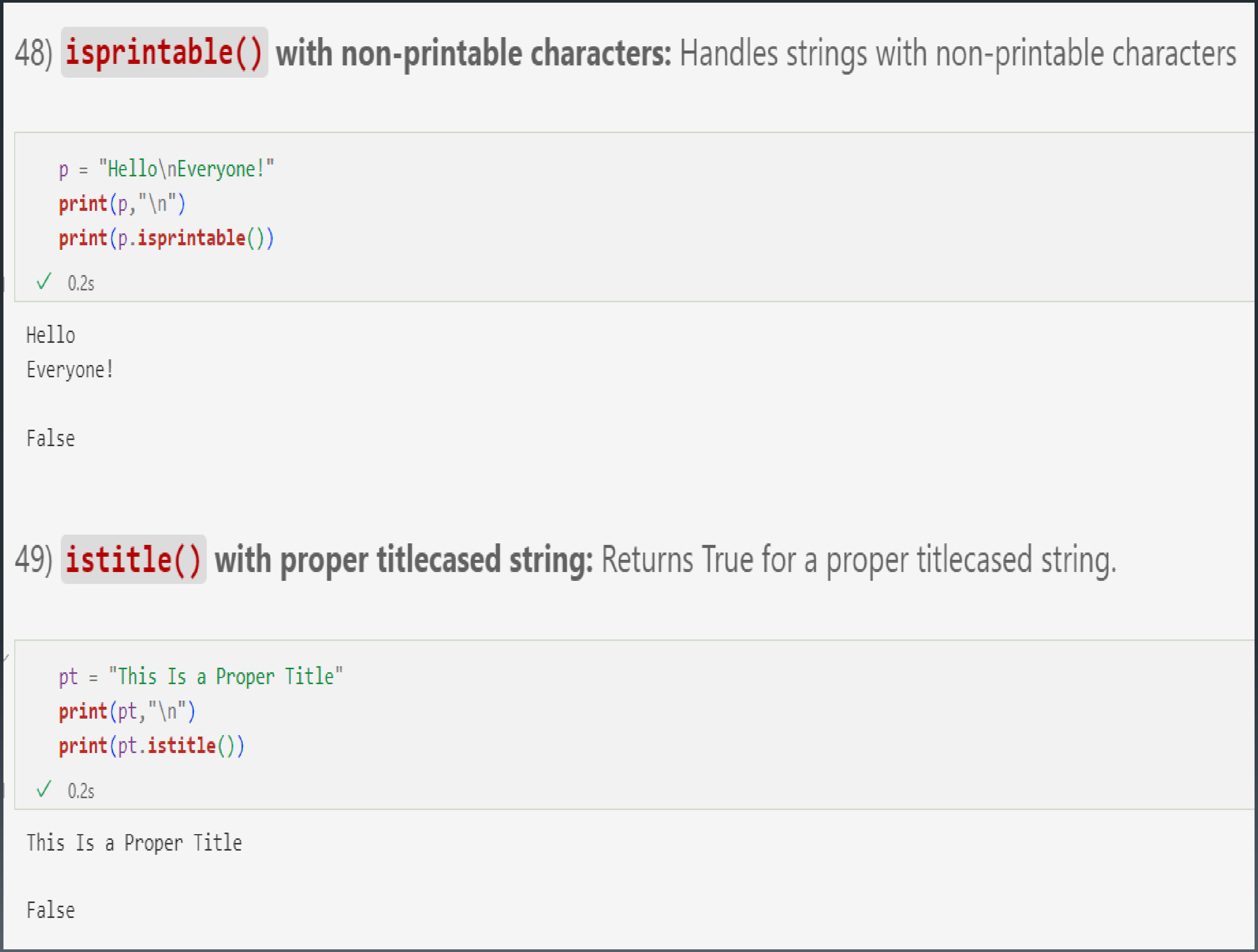
```
résumé is important. 

Résumé Is Important.
```

47) `isascii()` **with non-ASCII characters**: Handles strings with non-ASCII characters.

```python
s = "你好"
print(s, "\n")
print(s.isascii())
```

✓ 0.1s

```
你好 

False
```



48) `isprintable()` **with non-printable characters:** Handles strings with non-printable characters

```
p = "Hello\nEveryone!"
print(p,"\n")
print(p.isprintable())
```

✓ 0.2s

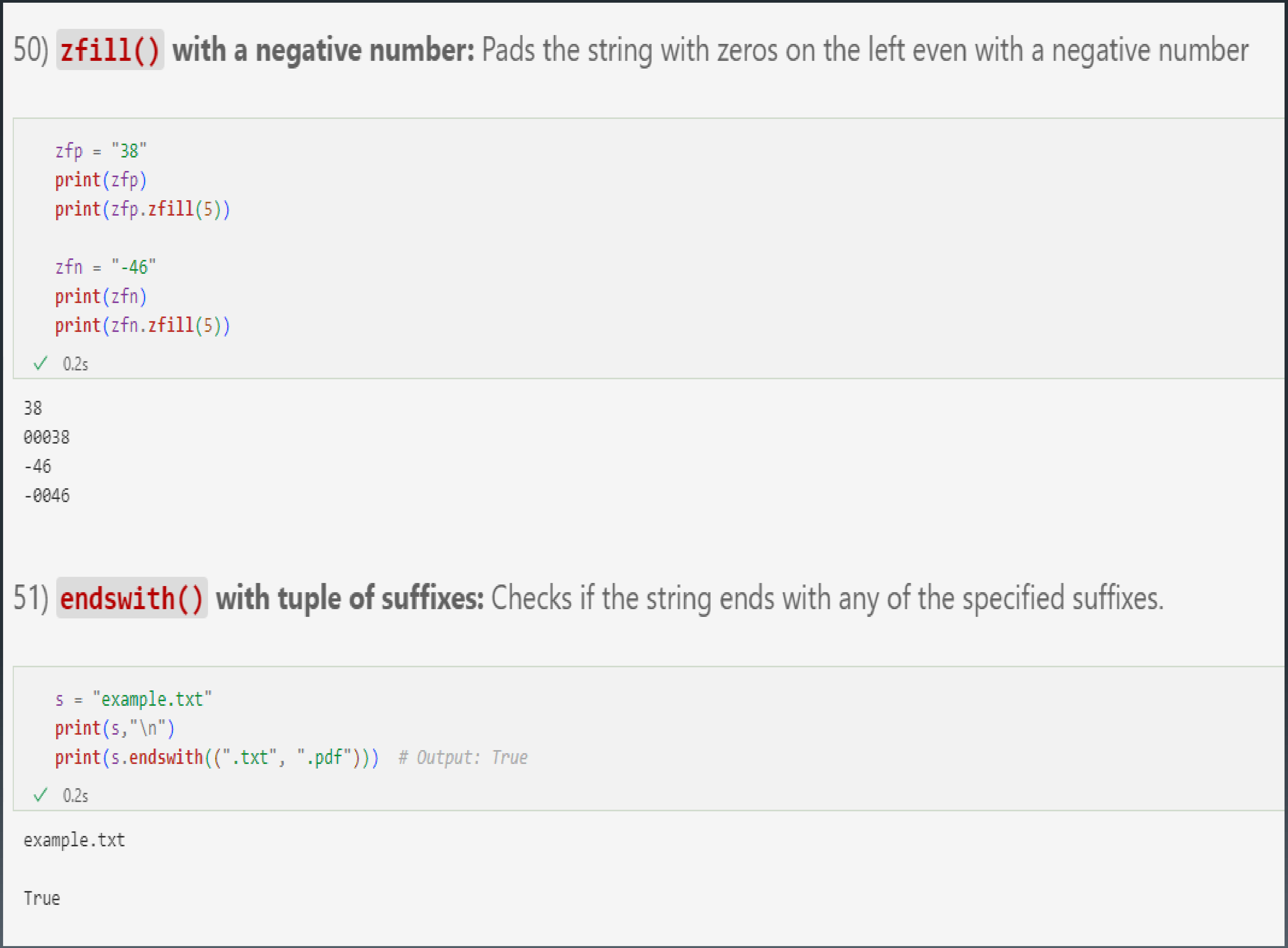
```
Hello
Everyone!

False
```

49) `istitle()` **with proper titlecased string:** Returns True for a proper titlecased string.

```
pt = "This Is a Proper Title"
print(pt,"\n")
print(pt.istitle())
```

✓ 0.2s

```
This Is a Proper Title

False
```



50) `zfill()` **with a negative number:** Pads the string with zeros on the left even with a negative number

```
zfp = "38"
print(zfp)
print(zfp.zfill(5))

zfn = "-46"
print(zfn)
print(zfn.zfill(5))
```

✓ 0.2s

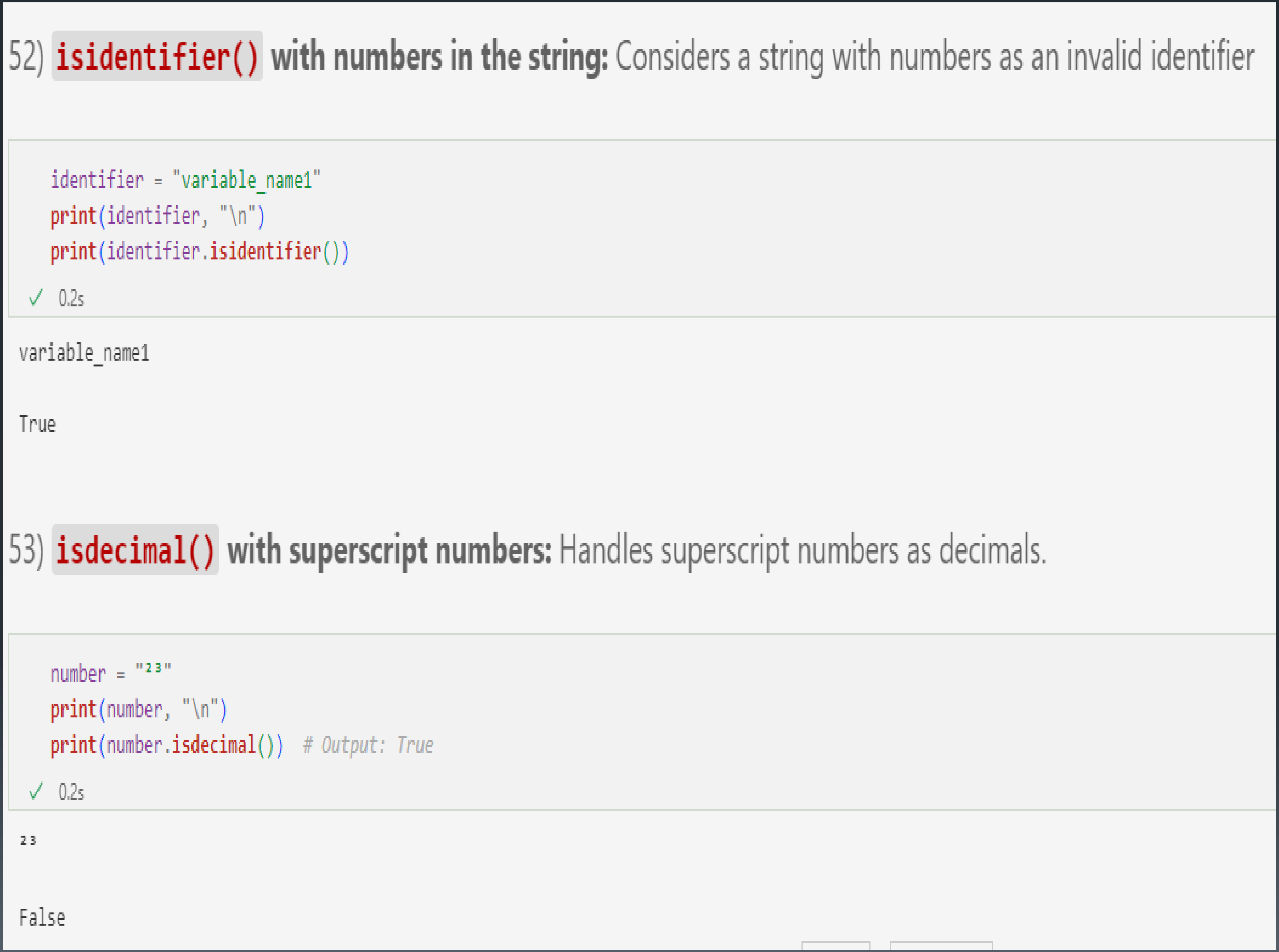
```
38
00038
-46
-0046
```

51) `endswith()` **with tuple of suffixes:** Checks if the string ends with any of the specified suffixes.

```
s = "example.txt"
print(s,"\n")
print(s.endswith((".txt", ".pdf")))  # Output: True
```

✓ 0.2s

```
example.txt

True
```



52) `isidentifier()` **with numbers in the string:** Considers a string with numbers as an invalid identifier

```python
identifier = "variable_name1"
print(identifier, "\n")
print(identifier.isidentifier())
```

✓ 0.2s

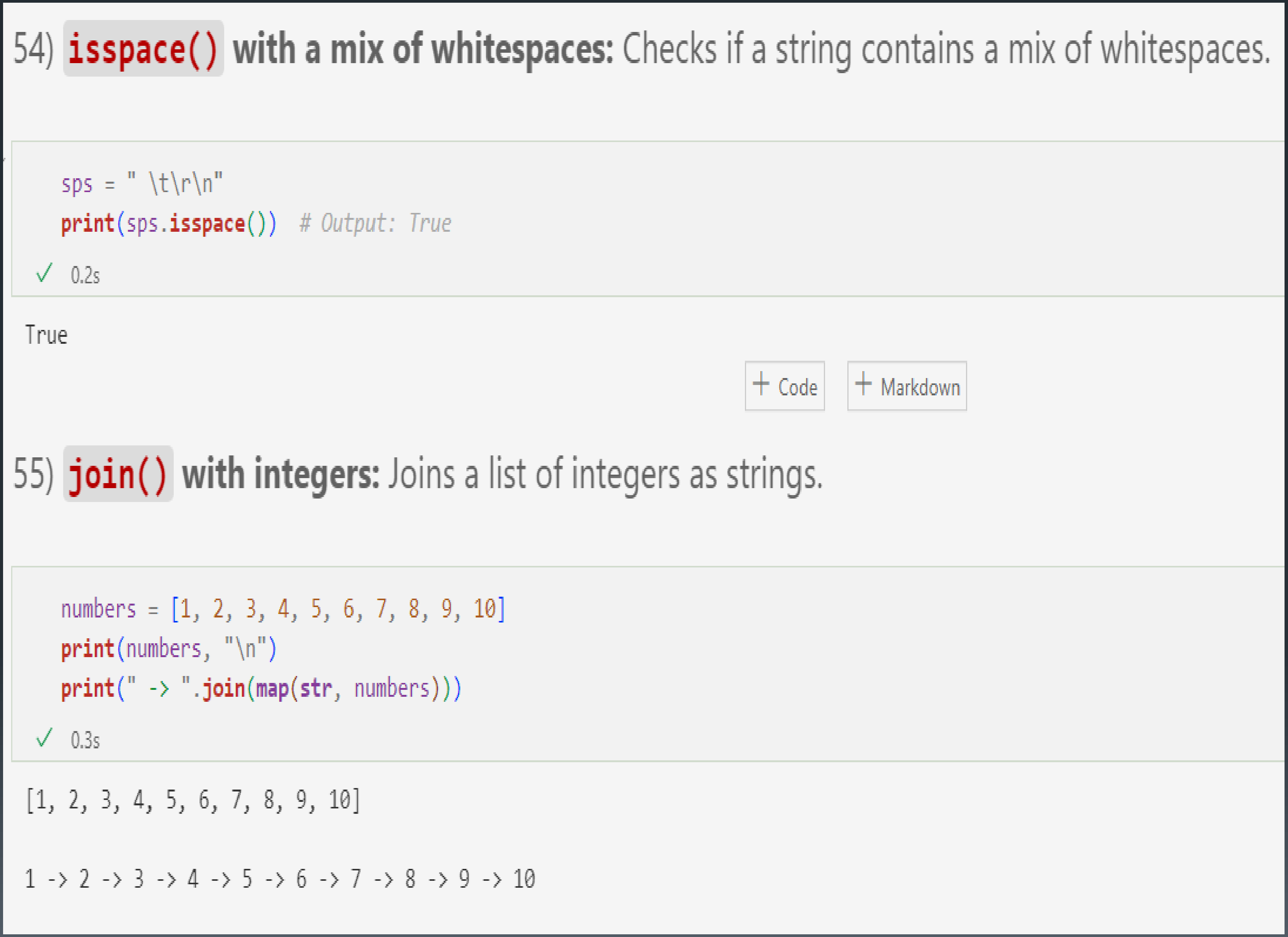
```
variable_name1

True
```

53) `isdecimal()` **with superscript numbers:** Handles superscript numbers as decimals.

```python
number = "²³"
print(number, "\n")
print(number.isdecimal())  # Output: True
```

✓ 0.2s

```
²³

False
```



54) `isspace()` **with a mix of whitespaces:** Checks if a string contains a mix of whitespaces.

```
sps = " \t\r\n"
print(sps.isspace())  # Output: True
```

✓ 0.2s

```
True
```

55) `join()` **with integers:** Joins a list of integers as strings.

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
print(numbers, "\n")
print(" -> ".join(map(str, numbers)))
```

✓ 0.3s

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10
```

SOURCE CODE LINK:

- https://github.com/IrtizaZaidi356/Practice_Python_DS_ML_CV_AI/blob/main/Basic_Python/01_String_Manipulation.ipynb

GITHUB LINK:

- https://github.com/IrtizaZaidi356