```
Ⅲ …
                                1 User Settings
                                                 percipio10_bytes_type.py X
       EXPLORER
                                       # percipio10 bytes type.py
     4 OPEN EDITORS
                                       # Percipio video: Data & Sequence Types; The Bytes Type in Python
         User Settings C:\Users\pc...
Q
         percipio10_bytes_type.p...
                                       # This shows 4 different ways to construct a byte object
                                       # The bytes class provides an immutable sequence (immutable = unchanging)
     ▲ PYTHON
89
                                       # Values must be integers from 0-255 to represent a byte
       ▶ Automate-Boring-Stuff
                                       # bytes literal object is storing the bytes for each of the characters in Unicode utf-8 format
       ▶ my_code
(%)
                                            by default
       ▲ Percipio_Python3-Course
                                       n1 = ' \ n'
        ▶ 01_Start
print(nl, 'Next')
        ■ 02_Data-Sequence Types
                                       bytes literal = b'Copyright \xc2\xa9' # similar to the 'r' prefix in a string, prints exactly
         percipio04_int_types.py
                                            provied contains only ASCII &/or escaped-hexidecimal characters
         percipio05_float_type.py
                                       print(nl, 'Next')
         percipio06_math_functio..
         percipio07_boolean_typ...
                                       # 1st way to create a byte object
                                  11
                                       # bytes literal.decode() decodes using Unicode utf-8 format by default if nothing is specified
         percipio08_Strings.py
                                       print('bytes_literal =', bytes_literal) #
         percipio09_float_type.py
                                       print('bytes literal.decode() ->',
         percipio10_bytes_type.py
                                            bytes literal.decode()) # default
         percipio11_bytearray_ty...
                                       print('bytes literal.decode("utf-8") ->',
         percipio12_list_type.py
                                            bytes literal.decode("utf-8")) # specified, but still befault format
         percipio13_tuple_type.py
                                       print('bytes literal.decode("utf-16") ->',
         percipio14_slice_type.py
                                            bytes_literal.decode("utf-16")) # encode & decode use different formats so error occurs
         percipio14a_list_copy_b...
                                                showing Asian characters
        ▶ 03_Collections-Mapping-L...
        ▶ 04 Modules-Functions
                                       # CAUTION: Must encode & decode using the same format or errors will occur
        ▶ 05 Classes
                                       print(nl, 'Next')
                                       # 2nd way to create a byts object
        ▶ 06_Working-with-Files
        ■ 07_Comprehensions
                                       str literal = 'Trademark ®' #
                                       bytes_encoded = str_literal.encode() # Creates a bytes object using the encode method encoding
         percipio45_list_compreh...
         percipio46_nested_com...
                                            the str literal variable which is Trademark ®
                                       print('bytes_encoded =', bytes_encoded) #
         percipio47_zip_function...
                                       print('bytes encoded.decode() ->',
         percipio48_set_compreh...
                                            bytes encoded.decode()) #
         percipio49 dictionary c...
                                       print('bytes(str literal) ->',
        ▶ 08 Iterables-and-Generat...
```

EXPLORER	{} User Sett	tings • percipio10_bytes_type.py ×	⊞ …
▲ OPEN EDITORS	28 p	rint('bytes(str_literal) ->',	
User Settings C:\Users\pc	29	<pre>bytes(str_literal, 'utf-8')) #</pre>	
percipio10_bytes_type.p	30 p	rint(nl, 'Next')	
▲ PYTHON	31 #	3rd way to create a bytes object	
Automate-Boring-Stuff	32 b	ytes_construct = bytes(str_literal, 'utf-8') # bytes_constructor when applied to a str_literal	Emerica- Escapea Escapea
my_code		will yeild a byte-literal-string	
▲ Percipio_Python3-Course	33 p	rint('bytes_construct.decode() ->',	
▶ 01_Start	34	<pre>bytes_construct.decode()) #</pre>	
 02_Data-Sequence Types 	35 p	rint(nl, 'Next')	
percipio04_int_types.py	36 #	4th way to create a bytes object	
percipio05_float_type.py	37 b	ytes_from_hex = bytes.fromhex('54 72 61 64 65 6d 61 72 6b 20 c2 ae') #	
🅏 percipio06_math_functio	38 p	rint('bytes_from_hex.decode() ->',	
🅏 percipio07_boolean_typ	39	<pre>bytes_from_hex.decode()) #</pre>	
percipio08_Strings.py	40 #	A bytes sequence behaves similar to a string	
percipio09_float_type.py	41 p	rint('str_literal.count("T") ->',	
percipio10_bytes_type.py	42	str_literal.count('T')) # counts number of T's in a string	
percipio11_bytearray_ty	43 p	rint('str_literal.index("T") ->',	
percipio12_list_type.py	44	<pre>str_literal.index('T')) # indexes position number of T's in a string</pre>	
percipio13_tuple_type.py	45 #	This performes the same function as above except uses byte values instead of string values	
percipio14_slice_type.py	46 p	rint('bytes_encoded.count(0x54) ->',	
percipio14a_list_copy_b	47	bytes_encoded.count(0x54)) #	
03_Collections-Mapping-L	48 p	rint('bytes_encoded.index(0x54) ->',	
04_Modules-Functions	49	bytes_encoded.index(0x54)) #	
▶ 05_Classes	50		
06_Working-with-Files	51 R	ESULT:	
07_Comprehensions	52 N	lext	
	✓ OPEN EDITORS ① User Settings C:\Users\pc ② percipio10_bytes_type.p ✓ PYTHON Automate-Boring-Stuff Decode ✓ Percipio_Python3-Course Decode ✓ Percipio_Python3-Course Decomposite of the percipio04_int_types.py ② percipio04_int_types.py ② percipio05_float_type.py ② percipio06_math_functio ② percipio08_Strings.py ② percipio09_float_type.py ② percipio10_bytes_type.py ② percipio11_bytearray_ty ② percipio12_list_type.py ② percipio13_tuple_type.py ② percipio14_slice_type.py ② percipio14a_list_copy_b Decomposite of the percipions of the	✓ OPEN EDITORS 28 () User Settings C:\Users\pc 29 ♠ percipio 10_bytes_type.p 30 pp ♠ PYTHON 31 # ♠ Automate-Boring-Stuff 32 b ♠ my_code 4 Percipio_Python3-Course 33 pp ♠ Percipio_Python3-Course 33 pp ♠ O1_Start 34 34 34 ♠ O2_Data-Sequence Types 35 pp ♠ percipio04_int_types.py 36 # ♠ percipio05_float_type.py 37 b ♠ percipio05_float_type.py 39 # ♠ percipio07_boolean_typ 39 # ♠ percipio08_Strings.py 40 # ♠ percipio09_float_type.py 41 pp ♠ percipio10_bytes_type.py 42 pp ♠ percipio12_list_type.py 43 pp ♠ percipio14_slice_type.py 45 # ♠ percipio14a_list_copy_b 47 P ♠ O3_Collections-Mapping-L 48 P ♠ O4_Modules-Functions 49 P ♠ O6_Working-w	O'DEN EDITIONS 1 User Settings C Albershop. 29

6	EXPLORER	User Settings	Ⅲ …
	▲ OPEN EDITORS	51 RESULT:	
2	{} User Settings C:\Users\pc	52 Next	
	percipio10_bytes_type.p	53	September 1
99	▲ PYTHON	54 Next	
>	Automate-Boring-Stuff	55 bytes_literal = b'Copyright \xc2\xa9'	Emerica- Emerica- Emerica- Posts
(▶ my_code	56 bytes_literal.decode() -> Copyright ©	
8	◆ Percipio_Python3-Course	bytes literal decode("utf-8") -> Convright @	
	▶ 01_Start	58 bytes_literal.decode("utf-16") ->	
	■ 02_Data-Sequence Types	59	
	percipio04_int_types.py	60 Next	
	percipio05_float_type.py	61 bytes_encoded = b'Trademark \xc2\xae'	
	percipio06_math_functio	62 bytes_encoded.decode() -> Trademark ®	
	percipio07_boolean_typ	63 bytes(str_literal) -> b'Trademark \xc2\xae'	
	percipio08_Strings.py	64	
	percipio09_float_type.py	65 Next	
	percipio10_bytes_type.py	66 bytes_construct.decode() -> Trademark ®	
	percipio11_bytearray_ty	67	
	percipio12_list_type.py	68 Next	
	percipio13_tuple_type.py	69 bytes_from_hex.decode() -> Trademark ®	
	percipio14_slice_type.py	70 str_literal.count("T") -> 1	
	percipio14a_list_copy_b	71 str_literal.index("T") -> 0	
	03_Collections-Mapping-L	72 bytes_encoded.count(0x54) -> 1	
	▶ 04_Modules-Functions	73 bytes_encoded.index(0x54) -> 0	
	> 05_Classes	74	