Which of the following expressions corresponds to a dictionary with no elements?	Dictionaries	
detail  detail	ATEST SUBMISSION GRADE	
desct	. Which of the following expressions corresponds to a dictionary with no elements?	1/1 point
Comment	☑ 0	
Context an autisting dictionary formation, what Python statement adds the key "frait" to this dictionary with the corresponding water "blackberry?"		
Corect   Supplies a dictionary can be consequently asked by "Entit" to this dictionary with the corresponding values "Flacebarry?"	☑ dict()	
Content on ministing dictionary farmatisms, what Python statement adds the key "fruit" to this dictionary with the corresponding what "blackberry?"    farmatisms ("fruit" : "blackberry?"   farmatisms ("fruit" : "blackberry?"   farmatisms ("fruit" : "blackberry?"   farmatisms ("fruit" : "blackberry?"   farmatisms ("fruit" : "blackberry?"   devertisms ("fruit" : "blackberry?"   devertisms ("fruit" : "blackberry?"   devertisms ("fruit" : "blackberry?")   v	✓ Correct	
Common an entitle glectomary farm rivers, when Python oriented soft the keys "Firsts" to this decisionary unit the corresponding when "Manchinery?"		
Assertians ("Ereals": "blackboarry")   Exercise ("Ereals": "blackboarry")   Assertians ("Ereals": "bla	_ o	
Exercises "("frast" : "blackborry")   Exercise "("frast" : "blackborry")   Exercise "("frast" : "blackborry")   Exercise ("frast" : "blackborry")   Access   Exercise ("frast" : "blackborry")   Access	diven an existing dictionary favorites, what Python statement adds the key "fruit" to this dictionary with the corresponding value "blackborry"?	1/1 point
favoration = ("frost" : "blackborry")     favoration   frost the "related bray"       favoration   favorati		
Statistical ("Erects" : "Blackberry"]		
Which of the expressions before returns Trans when the dictionary sty_dictionary contains the key sty_key and false otherwise?    sty_key and false otherwise?   sty_dictionary has_key (sty_key)   sty_dictionary_has_key (sty_key)   style="total-style		
Which of the appressions below returns Trau when the dictionary ary_dictionary contains the key ary_key and false otherwise?    sy_key and false otherwise?   sy_key tim sy_dictionary.   wrong _ sy_dictionary. has_key (sy_key)   sy dictionary. h		
my_key in my_dictionary	Correct	
carect   cy_dictionary_has_kay (cy_key)   cy_dictionary_has_kay (cy_key)   cy_dictionary_kay (cy_key)	i. Which of the expressions below returns True when the dictionary $my\_dictionary$ contains the $ke$ $my\_key$ and False otherwise?	y 1/1 point
wy_dictionary_has_ley(wy_key)   wy_dictionary_key(wy_key)   wy_dictionary_key(wy_key)   wy_dictionary_for_key]   Stood   Stood   Correct   Booleans_are immutable.   dcc   wr   Correct   Indicatory_can have which of the following types?   1/1 paint   dcc   wr   Correct   Stoing   Stoing-Shall in Stoin key in the dictionary, Python raises a few force exception.   Python returns the value in the dictionary, Python raises a System exception.   Store-Shall in stot a key in the dictionary, Python raises a system error.   Correct   Write a function count_Letters(corrd_Libst) that takes a input a list of word is that are composed exception of stoiners are continued by the stoiner	my_key in my_dictionary	
my_dictionary_key(my_key)	✓ Correct	
wy.dictionary (ary, key)	_ my_dictionary.has_key(my_key)	
Agy in a dictionary can have which of the following types?   1/1 paint		
Ist   Dool	my_dictionary[my_key]	
bool   Carrect   Booleans are immutable.     det   of the     Carrect     fingers are immutable.     of the     carrect     interpretation   carrect     triggers are immutable.     triggers are immutable.     triggers are immutable.     string     Carrect     string     Carrect     det     Carrect     tuple     Carrect     tople     Carrect     tople     Carrect     tristructor_ratiops = {*Job** * "messors*, "Scott* : "mess*]     instructor_ratiops = {*Job** * "messors*, "Scott* : "mess*]     What happens when Python evaluates the expression instructor_ratings ("John*)?     Since *John** in not a key in the dictionary, Python ratios a Keyfirner exception.     Python returns the value Sions since *John* is not a key in the dictionary.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.     Since *John* in not a key in the dictionary, Python ratios a Seyfirner exception.	i. Keys in a dictionary can have which of the following types?	1/1 point
dict		
Booleans are immutable.    dict   mile   mile   carried   integers are immutable.   Values in a dictionary can have which of the following types?   1/1 paint   integers are immutable.   Values in a dictionary can have which of the following types?   1/1 paint   string   carried   carri		
Volues in a dictionary can have which of the following types?   1/1 point		
\times a function count_letters (word_list) that takes as input a first of words that are composed entirely of lower case letter that dictionary, Python raises a lyrighter exception.    Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a lyrighter exception.   Since "Joha" is not a key in the dictionary, Python raises a linguist of words that are composed entirely of lower case letters that are composed entirely follower case letters and whose values are the corresponding number of socurrences of exception is not a key in the dictionary is a list of words that are composed entirely follower case letters and whose values are the corresponding number of socurrences of exception to the strings in word_list.    Mine the single letter in the letter cont_lists is a list of words that are continued as a		
Integers are immutable.  **Values in a dictionary can have which of the following types?*  **Values in a dictionary can have which of the following types?*  **Values in a dictionary can have which of the following types?*  **Values in a dictionary can be considered by the considere		
dct  √ Currect  type  1 instructor_rettings = {*loo* : "awasone*, "scot*": "home"}  1 instructor_rettings = {*loo* : "awasone*, "scot*": "home"}  What happens when Python evaluates the expression instructor_ratings (*John*)?  © Since "John* in not a levy in the dictionary, Python raises a Keyfrore exception.  Python returns the value list one since "John* is not as key in the dictionary.  Since "John* in not a key in the dictionary, Python raises a Keyfrore exception.  Since "John* in not a key in the dictionary, Python raises a keyfrore exception.  Since "John* in not a key in the dictionary, Python raises a syntax error.  √ Currect  Write a function count_latters(word_list) that takes as input a list of words that are composed entirely of lower case letters. This functions follow from the lower case letter that expeater most Reposed (youth lamber of currence) and currence(a) in the value in word_list. (but case of taxe, return the earliest letter in alphabetical order case letter and whose values are the curresponding number of occurrence and certain test in size in alphabetical order.  1 if eff count_latters(ord,list):  2 if of count_latters(ord,list):  2 if of care count - (list):  3 if of care count - (list):  5 if of latters in ARMSET:  6 latters count - (list):  7 for latters in ARMSET:  8 latters count - (list):  8 latters count - (list):  9 for latters in alphabetical order case letter return by count_latters(count_latters(count_latters):  1 most, word = spite of government in general currence and in the strings in order in your code, compute the lower case letter return by count_latters(count_latters	i. Values in a dictionary can have which of the following types?	1/1 point
Consider the following dictionary:	☑ string	
tuple  ✓ Correct  Lorenteder the following dictionary:  1 Instructor_retings = {**Joe** : "ownsome*, "scott* : "home**]  1 Instructor_retings = {**Joe** : "ownsome*, "scott* : "home**]  What happens when Pythen evaluates the expression instructor_retings {**John**}?  Since **John** is not a key in the dictionary, Python raises a Myeferor exception.  Python returns the value lisue since **John* is not a key in the dictionary.  Since **John** is not a key in the dictionary, Python raises a Syeferor exception.  Since **John** is not a key in the dictionary, Python raises a Syeferor exception.  Since **John* is not a key in the dictionary, Python raises a syntax error.  ✓ Correct  Write a function countletters(word_list) that takes as input a list of words that are composed entirely of lower case letters. This functions should return the lower case letter that groups are most requestly total number of correctness of each letter in the strings in word_list. (In the case of ties, return the earliest letter in alphabetical order.)  The Python code repetition is a struct a implementing countletter using a dictionary latter; count these keys are the lower case letters and whose values are the corresponding number of correctness of each letter in the strings in word_list.  1 def count_letters(count_tist):  2 def count_letters(count_tist):  3 def count_letters(count_tist):  4 def count_letters(count_tist):  5 def letter_count = {1 def count_letters(count_tist):  5 def letter_count = {1 def count_letters} = 0  2 def count_letters(count_tist):  5 def count_letters(count_tist):  6 def count_letters(count_tist):  7 def count_letters(count_tist):  8 def count_letters(count_tist):  9 def count_letters(count_tist):  9 def count_letters(count_tist):  1 def count_letters(count_tist):  1 def count_letters(count_tist):  1 def count_letters(count_tist):  2 def count_letters(count_tist):  3 def count_letters(count_tist):  4 def count_letters(count_tist):  5 def count_letters(count_tist):  6 def count_letters(count_tist):  9	✓ Correct	
tuple  / Carriect  Consider the following dictionary:  1 testinctor_retings = {"Joe" : "emesore", "Scott" : "Imme"]  1 testinctor_retings = {"Joe" : "emesore", "Scott" : "Imme"]  What happens when Python evaluates the expression instructor_ratings {"John"}?  Since "John" is not a key in the dictionary, Python raises a Keyferor exception.  Since "John" is not a key in the dictionary, Python raises a Keyferor exception.  Since "John" in not a key in the dictionary, Python raises a Keyferor exception.  Since "John" in not a key in the dictionary, Python raises a keyferor exception.  Since "John" in not a key in the dictionary, Python raises a syntax error.  / Carriect  Write a function count_letters(word_list) that takes as input a list of words that are composed entirely of flower case letters. This function should return the lower case letter that appears most frequently (testin number Georemics) in the content in word_list.  Write a function count_letters(word_list) that takes as input a list of words that are composed entirely follower has been declared in the strings in word_list.  The Python code supple below represents a start at implementing count_letters using a dictionary latter, count whose keys are the lower case letter and whose values are the corresponding number of occurrences of each letter in the strings in word_list.	☑ dict	
Consider the following dictionary:  1/1 point    Instructor_ratings = {"Joe" : "mesone", "scott" : "meso"    Since "Joba" is not a key in the dictionary, Python raises a Keyforre exception.   Since "Joba" is not a key in the dictionary, Python raises a Keyforre exception.   Since "Joba" is not a key in the dictionary, Python raises a Keyforre exception.   Since "Joba" is not a key in the dictionary, Python raises a keyforre exception.   Since "Joba" is not a key in the dictionary, Python raises a keyforre exception.   Since "Joba" is not a key in the dictionary, Python raises a keyforre exception.   Since "Joba" is not a key in the dictionary, Python raises a keyforre exception.   Since "Joba" is not a key in the dictionary, Python raises a keyforre exception.   Since "Joba" is not a key in the dictionary, Python raises a keyforre exception.   Since "Joba" is not a key in the dictionary, Python raises a keyforre exception.   Since "Joba" is not a key in the dictionary, Python raises a keyforre exception.   The Python code exception to the sphale-decidated roter.   The Python code is negligated by the sphale exception to the sphale-decidated roter.   The Python code is capacitated of course. I have a sphale exception to the sphale-decidated roter.   The Python code is capacitated of course. I have a sphale exception to the spha	✓ Correct	
Consider the following dictionary:  1/1 paint    1   instructor_retings = (**Joe* : "eacoses", "Scott* : "Penes")    1   instructor_retings = (**Joe* : "eacoses", "Scott* : "Penes")    1   instructor_retings = (**Joe* : "eacoses", "Scott* : "Penes")    2   instructor_retings = (**Joe* : "eacoses", "Scott* : "Penes")    3   instructor_retings = (**Joe* : "eacoses", "Scott* : "Penes")    4   instructor_retings = (**Joe* : "eacoses = (**Joe* : "eacose = (**Joe*	☑ tuple	
Consider the following dictionary:  1 instructor_ratings = {*"Joh** : "awazosa*, "Scott* : "home,"}  1 instructor_ratings = {*"Joh** : "awazosa*, "Scott* : "home,"}  What happens when Python evaluates the expression instructor_ratings["John*]?  © ince "John** in not a key in the dictionary, Python raises a Keyferre exception.  Python returns the value Biosa since "John* in not a key in the dictionary.  For a return of the same because the dictionary python raises a keyferre exception.  Since "John* in not a key in the dictionary, Python raises a logical six of words that are composed entirely of lower case letters. This function should return the lower case letter that appears most frequently intelled number of occurrences] in the words in word_list. (In the case of text, return the active letter in alphabetical orders.)  The Python code snippet below represents a start at lamplementing count_letters using a dictionary latter. count whose keys are the lower case letters and whose values are the corresponding number of occurrences deal texts in the strings in word_list.  1 def count_letters(ont_list): 2	✓ Correct	
Consider the following dictionary:   1	☑ bool	
I instructor_ratings = ("Joe" : "awasons", "Scott" : "Nemm")  What happens when Python evaluates the expression instructor_ratings ("John")?  ⑤ ince "John" is not a key in the dictionary, Python raises a Keyferre exception.  Python returns the value Nose since "John" is not a key in the dictionary.  Since "John" in and value in the dictionary. Python raises a Keyferre exception.  Since "John" in an a value in the dictionary. Python raises a Keyferre exception.  Since "John" in an a value in the dictionary. Python raises a long-time of the very district that are unusually as the python raises a syntax error.  ✓ Cornect  With a function countLetters("cord_list") that takes as input a fixed words that are unusually of lower case letters. This function should return the lower case letters that appears most frequently floated number of occurrences] in the words in word_list. (In the case of tax, return the actine letter in alphabetical order.)  The Python code snippet below represents a start at lamplementing count_letters using a dictionary. Jetter_count values are the lower case letter and whose values are the corresponding number of occurrences deal texts in the strings in word_list.  1 def count_letters(cont_list):  2	✓ Correct	
What happens when Python evaluates the expression instructor_ratings[*John*]?  Since *John* is not a key in the dictionary, Python raises a Keyferor exception.  Python returns the value Ricea since *John* is not a key in the dictionary.  Since *John* is not a key in the dictionary, Python raises a Keyferor exception.  Since *John* is not a key in the dictionary, Python raises a keyferor exception.  Since *John* is not a key in the dictionary, Python raises a syntax error.  */ Curriet  Write a function count_letters(word_list) that takes as input a list of words that are composed entirely of lower case letters. This function should return the lower case letter that appears most requestly fotals number of courrences in New Just vote in word_list. Use case of tax, return the earlies letter in alphabetical order)  The Python ocal supplex below represents a start at implementing count_letters using a dictionary latter_count whose keys are the lower case letters and whose values are the corresponding number of occurrences of each letter in the strings in word_list.	i. Consider the following dictionary:	1/1 point
Since *Joba* is not a key in the dictionary, Python raises a Keyferor exception.  □ Python returns the value floos aims *Joha* is not a key in the dictionary.  □ Since *Joba* is not a value in the dictionary, Python raises a Keyferor exception.  □ Since *Joba* is not a key in the dictionary, Python raises a Keyferor exception.  □ Since *Joba* is not a key in the dictionary, Python raises a syntax error.  ✓ Carrect  Write a function count_letters (word_list) that takes as input a list of words that are composed entirely of flower case letter. This function should return the lower case letter that appears most frequently (total number of courtences) in the vords in word_list. (but case of ties, return the earliest letter in alphabetical order.)  The Python code imples below represents a star at implementing count_letters using a dictionary latter, count whose keys are the lower case letter and whose values are the corresponding number of occurrences of each letter in the strings in word_list.    dictionary_latter_count_list() =	1 instructor_ratings = {"Joe" : "awesome", "Scott" : "hemm"}	
Since *Joba* is not a key in the dictionary, Python raises a Keyferor exception.  □ Python returns the value floos aims *Joha* is not a key in the dictionary.  □ Since *Joba* is not a value in the dictionary, Python raises a Keyferor exception.  □ Since *Joba* is not a key in the dictionary, Python raises a Keyferor exception.  □ Since *Joba* is not a key in the dictionary, Python raises a syntax error.  ✓ Carrect  Write a function count_letters (word_list) that takes as input a list of words that are composed entirely of flower case letter. This function should return the lower case letter that appears most frequently (total number of courtences) in the vords in word_list. (but case of ties, return the earliest letter in alphabetical order.)  The Python code imples below represents a star at implementing count_letters using a dictionary latter, count whose keys are the lower case letter and whose values are the corresponding number of occurrences of each letter in the strings in word_list.    dictionary_latter_count_list() =	What hungar when Buthon avaluate the expression instructor, various (* John*12	
Since *Joba* in not a value in the dictionary, Python raises a Knyferor exception.  Since *Joba* in not a key in the dictionary, Python raises a syrtax error.  // Currict  Write a function count_letters(word_list) that takes as input a list of words that are composed entirely of lower case letters. This function should return the lower case letter that appears most frequently (total number of occurrences) in the words in word_list. (In the case of take, return the earlies letter in alphabetical orders)  The Python code snippet below represents a start at implementing cosmlatts are using a dictionary latter, count whose keys are the lower case letters and whose values are the corresponding number of occurrences of each letter in the strings in word_list.		
Since "Joha" is not a key in the dictionary, Python raises a syntax error.  Views function count_letters(ourd_list) that takes a input a fixed words that are composed entirely of lower case letters. This function should return the lower case letter that appears most frequently floated number of occurrences) in the words in word_list. (In the case of take, return the case letter should return the lower case letter that appears most frequently floated number of occurrences) in the words in word_list. (In the case of take, return the case letter is set of the case of take, return the case letter is an expensive state of the contraction		
Write a function count_lattere(word_list) that takes as input a list of words that are composed entirely of lower case letters. This function should return the lower case letter that appears most frequently total number of occurrences) in the words in word_list. (In the case of tex. return the case letter has appear most frequently count number of occurrences) in the words in word_list. (In the case of tex. return the case letter has a set of the lower case letter using a dictionary latter_count where the lower case letter and whose values are the corresponding number of occurrence of each letter in the using in word_list.    1		
composed entirely of flower case letters. This function should return the lower case letter that appears most frequently (total number of courtences) in the votal most all to the case of ties, return the earliest letter in alphabetical order.)  The Python code impose below represents a start at implementing count_letters using a dictionary latter_count whose keys are the lower case letter and whose values are the corresponding number of occurrences of each letter in the strings in word_list.    def count_letter_count_c	✓ Correct	
composed entirely of flower case letters. This function should return the lower case letter that appears most frequently (total number of courtences) in the votal most all to the case of ties, return the earliest letter in alphabetical order.)  The Python code impose below represents a start at implementing count_letters using a dictionary latter_count whose keys are the lower case letter and whose values are the corresponding number of occurrences of each letter in the strings in word_list.    def count_letter_count_c	Write a function count. Letters (word list) that takes as input a list of words that are	1/1 point
The Python code snippet below represents a start at Implementing count_letters using a dictionary letter_count whose keys are the lower case letters and whose values are the corresponding number of occurrence of sech letter in the strings in word_list.  1 of for count_letters/cond_lists). 2 *** "See quantion description *** 4 APPMETF - "doctedinglistimogystawways" 5 ** Interecount - () ** 5 ** Interecount - () ** 7 ** for letter in APPMETF: 9 ** Interecount - () ** 9 ** 9 ** Interecount - () ** 9 ** 9 ** 9 ** 9 ** 9 ** 9 ** 9 **	composed entirely of lower case letters. This function should return the lower case letter that appears most frequently (total number of occurrences) in the words in word_list. (In the case of	
corresponding number of excurrences of each letter in the strings in word_list.  2	The Python code snippet below represents a start at implementing count_letters using a	
A RIMBET - "dockef git lytimogratuwoyz"  6	corresponding number of occurrences of each letter in the strings in word_list.	
Complex pour implementation of counts_lessers based on this snippet. As a text.  counts_lessers(_hallon'_ "verlet') should return the letter '1' save '1' appears 3 times  total in the entireg. "ballo" and "verlet'.  When you are confident in your code, compute the lower case letter return by  counts_lesters conts_vercefs where entry, verde is defined as shown.  1 sonty_opte = "listen strongs aware lying in prote distributing arooms is no  basis for a system of governess begree executive governer drives from a  monitor from the masses not from some forcical apartic ceremony"  2 monty_work = monty_opte.a.plitt(" ")  Enter this single letter in the text box below. Do not include any spaces or enclosing quotes around  the letter.	2 "" See question description ""	
Complex pour implementation of counts_lessers based on this snippet. As a text.  counts_lessers(_hallon'_ "verlet') should return the letter '1' save '1' appears 3 times  total in the entireg. "ballo" and "verlet'.  When you are confident in your code, compute the lower case letter return by  counts_lesters conts_vercefs where entry, verde is defined as shown.  1 sonty_opte = "listen strongs aware lying in prote distributing arooms is no  basis for a system of governess begree executive governer drives from a  monitor from the masses not from some forcical apartic ceremony"  2 monty_work = monty_opte.a.plitt(" ")  Enter this single letter in the text box below. Do not include any spaces or enclosing quotes around  the letter.	5 6 letter_count = {} 7 for letter in ALPHABET:	
count_letters(["hello", "world") should return the letter 'l' since 'l' appears it times total in the strings "hello" and "voil".  When you are confident in your code, compute the lower case letter return by count_letters(monty_world) where monty_world is defined as shown.  Jesus_quate - "litten strongs world by light point distribution genetic to bosts for a system of government supreme exactive poser derives from a monitor from the massis on from some forcical capacitic ceremony"  monty_monds = monty_quate.apite(" ')  Enter this single letter in the text box below. Do not include any spaces or enclosing quotes around the letter.		
When you are confident in your code, compute the lower case letter return by count. Letters (monty_words) where souty_words idefined as shown.    Senty_quete = "Listen arrange wash lysis proofs (stricthating souths is no boats for a system of government sources executive power derivative from a soundost from the assists not from some for/cital apartic ceremony?    Senty_words = mosty_quete.split(" ")    Enter this single letter in the text box below. Do not include any spaces or enclosing quotes around the letter.	Complete your implementation of count_letters based on this snippet. As a test, count_letters(["hello", "world"]) should return the letter 'l' since 'l' appears 3 times	
count_letters(monty_words) where monty_words idefined as shown.  1 monty_quote = 'listen stronge mone bying in ponds distributing seords is no monthly	When you are confident in your code, compute the lower case letter return by	
sorty_words = nosty_quate.split(" ')   Enter this single letter in the text box below. Do not include any spaces or enclosing quotes around the letter.	count_letters(monty_words) where monty_words is defined as shown.	
Enter this single letter in the text box below. Do not include any spaces or enclosing quotes around the letter.	basis for a system of government supreme executive power derives from a mandate from the masses not from some farcical aquatic ceremony.  3 monty_monds = monty_quote.spltif* "]	
the letter.		
	the letter.	