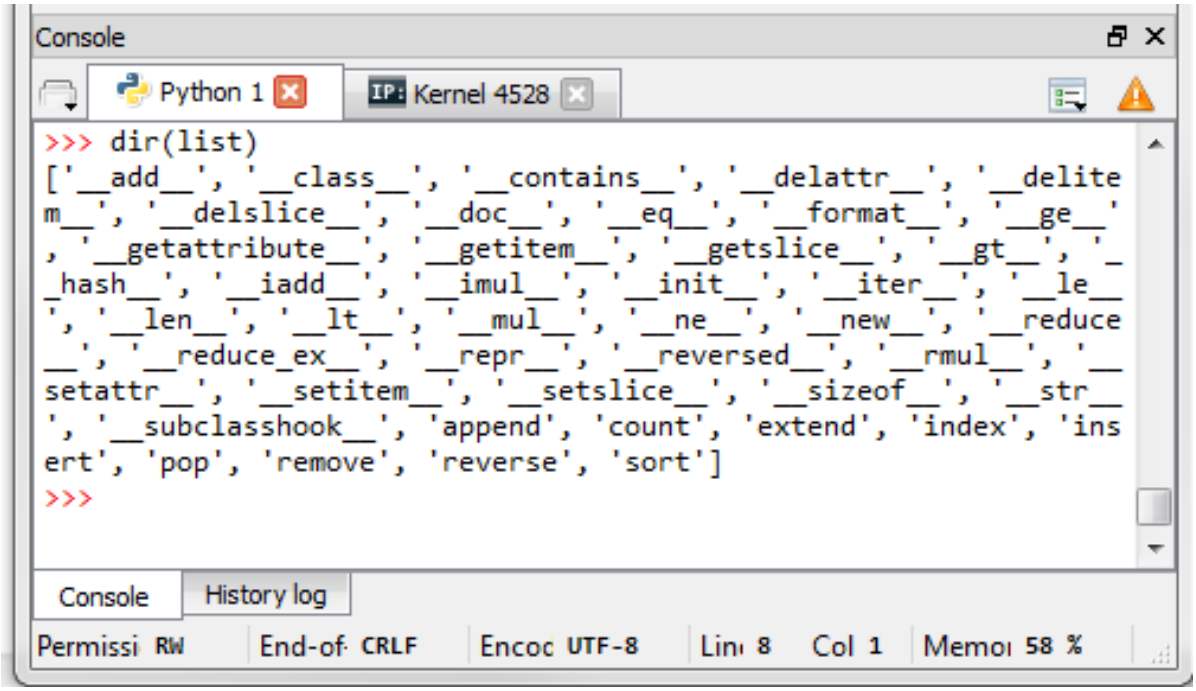
Lists: Methods

list Type Methods

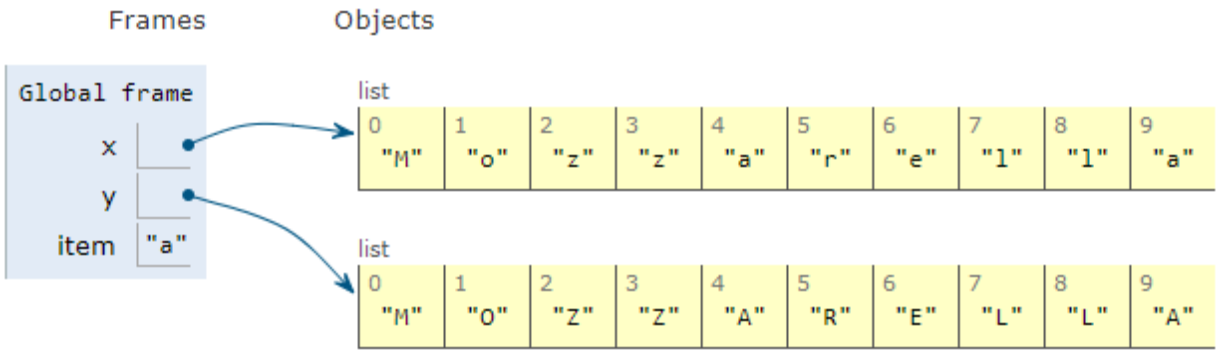
```
>>> dir(list)
```




```
>>> x = ['M','o','z','z','a','r','e','l','l','a']
>>> x.sort()
>>> print(x)
['M', 'a', 'a', 'e', 'l', 'l', 'o', 'r', 'z', 'z']
```

List Iteration

```
>>> # capitalize all letters in x
>>> x = list("Mozzarella")
>>> y = []   # y = list()
>>> for item in x:
>>>     y.append(item.upper())
```



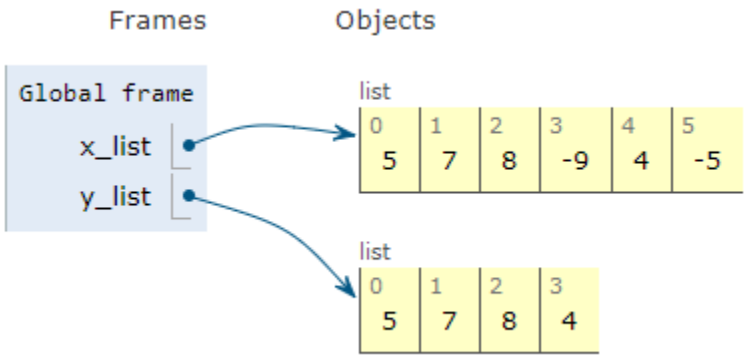
List Iteration Example: Remove Vowels

```
>>> x = list("Mozzarella")
>>> y = []
>>>  # same as y = list()
>>> VOWELS = list(“aeiou”)
>>> for item in x:
>>>     if item not in VOWELS:
>>>         y.append(item)
```

General Form of List Comprehension

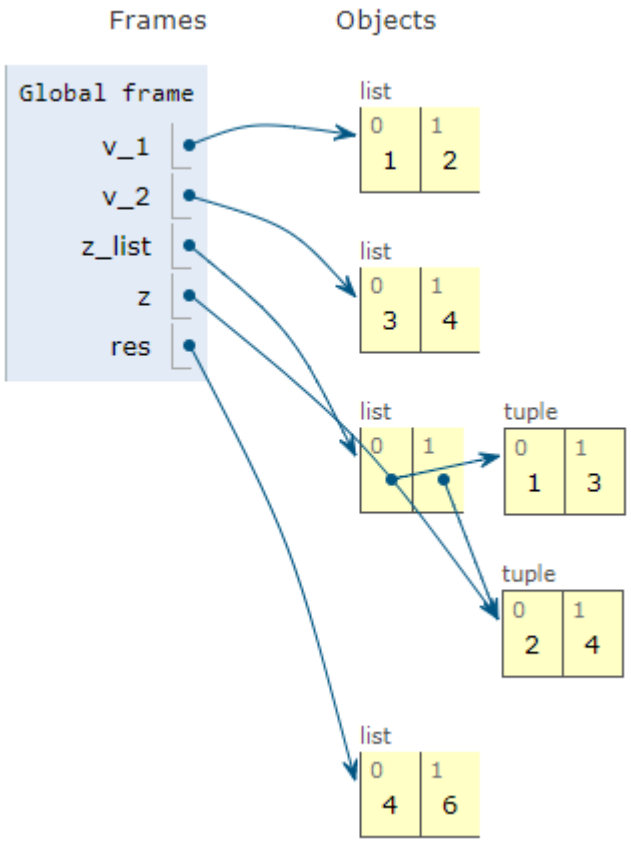
[expression for item in iterable
if condition]

```
>>> x_list = [5, 7, 8, −9, 4, −5]
>>> y_list = [x + 1 for x in x_list if x > 0]
>>> y_list
>>> [5, 7, 8, 4]
```



List Comprehension: Vector Addition

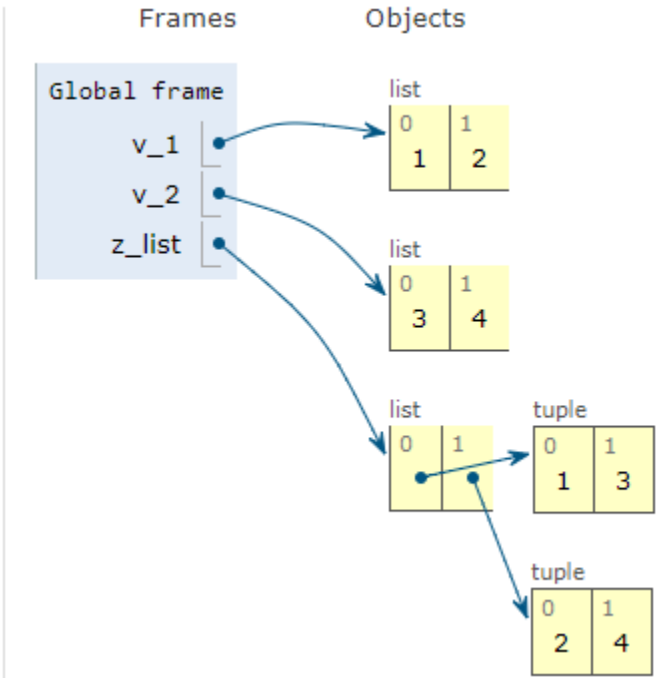
```
>>> v_1 = [1 ,2];  v_2 = [3, 4]
>>> z_list = zip(v_1, v_2)
```




```
>>> res = [z[0] + z[1] for z in z_list]
>>> res
[4, 6]
```

List Comprehension: Vector Dot Product

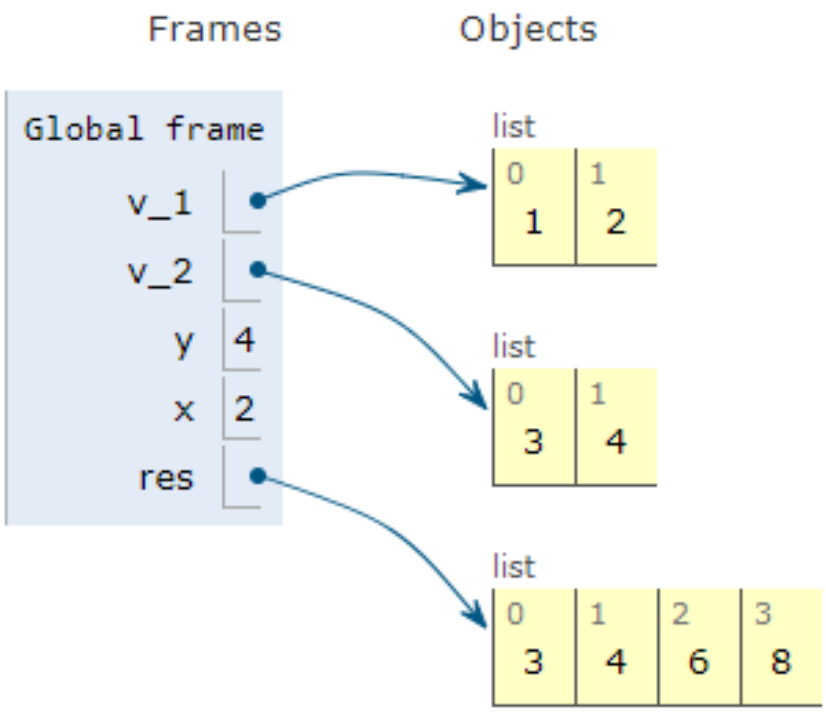
```
>>> v_1 = [1 ,2]
>>> v_2 = [3, 4]
>>> z_list = zip(v_1, v_2)
```




```
>>> res = sum(z[0] * z[1] for z in z_list)
>>> res
11
```

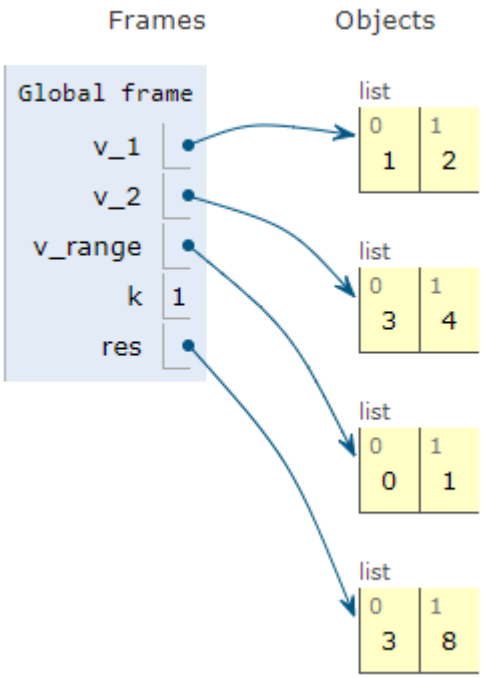
List Comprehension: Vector Cross Product

```
>>> v_1 = [1 ,2]
>>> v_2 = [3, 4]
>>> res = [x*y for x in v_1 for y in v_2]
>>> res
[3, 4, 6, 8]
```



List Comprehension: Hadamard Vector Product

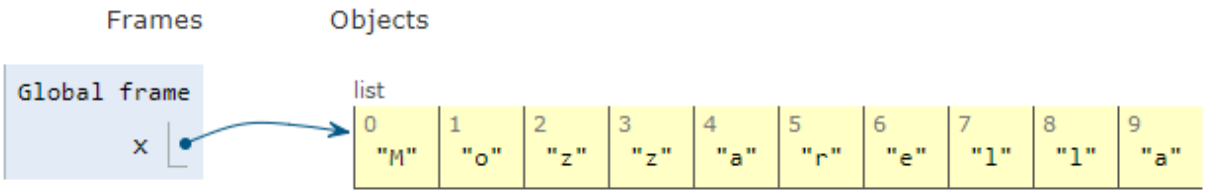
```
>>> # pointwise multiplication
>>> v_1 = [1 ,2];
>>> v_2 = [3, 4]
>>> v_range = range(len(v_1))
>>> res = [ v[k]+w[k]   for k in v_range ]
>>> res
[3, 8]
```



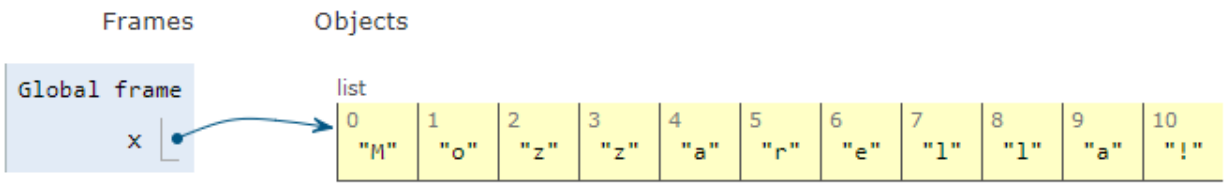
List Methods: *append*()

	0	1	2	3	4	5	6	7	8	9	
[	‘M’,	‘o’,	‘z’,	‘z’,	‘a’,	‘r’,	‘e’,	‘l’,	‘l’,	‘a’	]
	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	

```
>>> x = [‘M’,‘o’,‘z’,‘z’,‘a’,‘r’,‘e’,‘l’,‘l’,a’]
```



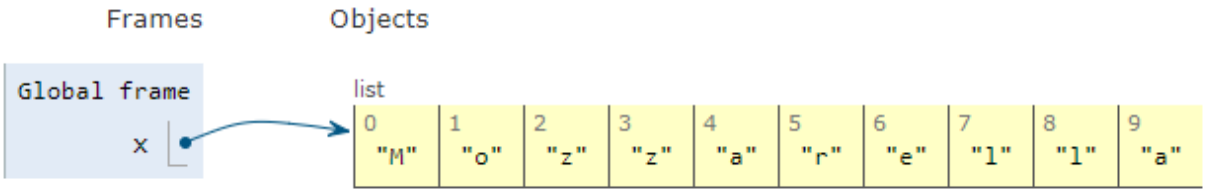

```
>>> # append ‘!’ to the end of list x
>>> x.append(‘!’)
>>> x
>>> [ ‘M’,‘o’,‘z’,‘z’,‘a’,‘r’,‘e’,‘l’,‘l’,‘a’,‘!’]
```



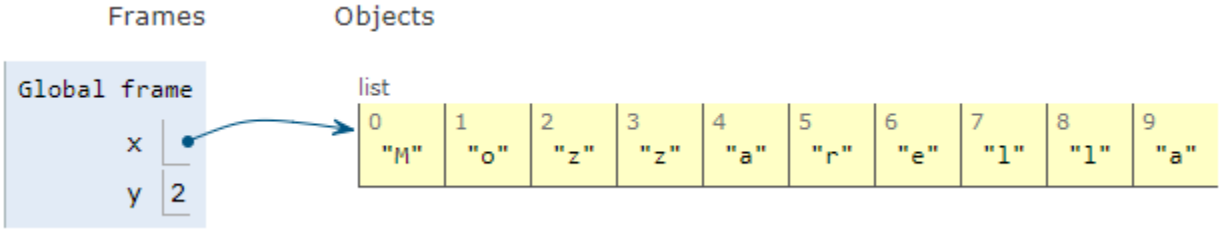
List Methods: *count*()

	0	1	2	3	4	5	6	7	8	9	
[	'M',	'o',	'z',	'z',	'a',	'r',	'e',	'l',	'l',	'a'	]
	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	

```
>>> x = ['M','o','z','z','a','r','e','l','l',a']
```



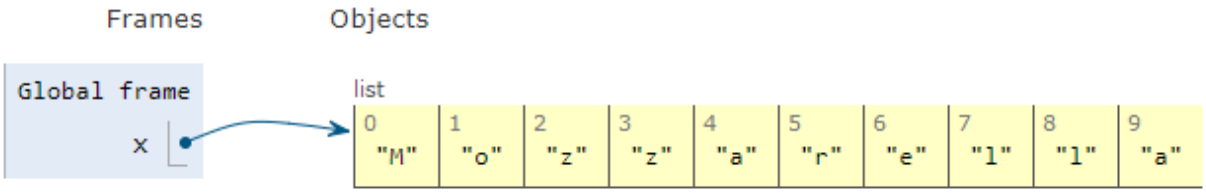

```
>>> # number of occurences of 'a' in x
>>> y = x.count('a')
>>> y
>>> 2
```



List Methods: *extend*()

	0	1	2	3	4	5	6	7	8	9	
[	'M',	'o',	'z',	'z',	'a',	'r',	'e',	'l',	'l',	'a'	]
	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	

>>> x = ['M','o','z','z','a','r','e','l','l',a']

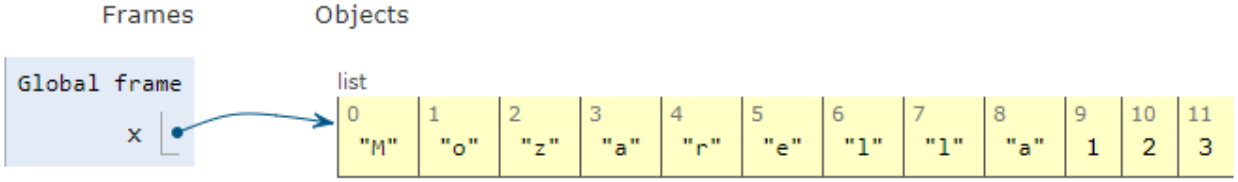


>>> append all items in list [1, 2, 3] to x

>>> x.*extend*([1, 2, 3])

>>> x

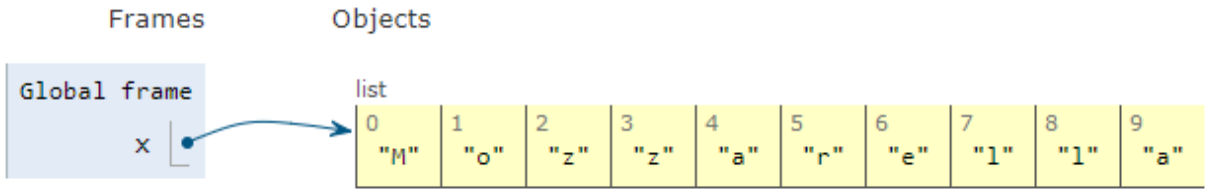
>>> ['M','o','z','z','a','r','e','l','l','a',1,2,3]



List Methods: *index*()

	0	1	2	3	4	5	6	7	8	9	
[	‘M’,	‘o’,	‘z’,	‘z’,	‘a’,	‘r’,	‘e’,	‘l’,	‘l’,	‘a’	]
	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	

>>> x = [‘M’,‘o’,‘z’,‘z’,‘a’,‘r’,‘e’,‘l’,‘l’,a’]

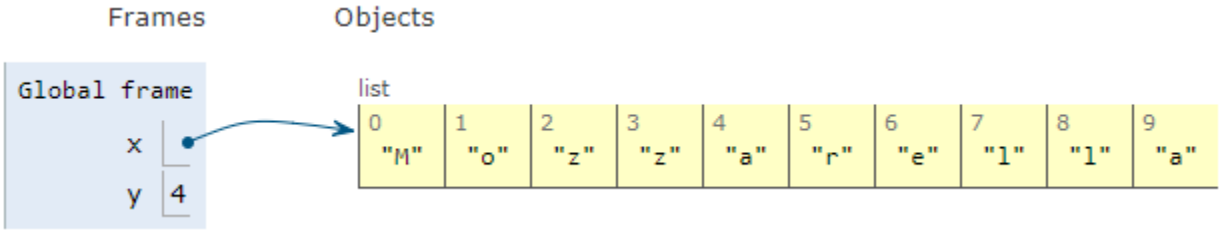


>>> # first occurrence of ‘a’ in list x

>>> y = x.*index*(‘a’)

>>> y

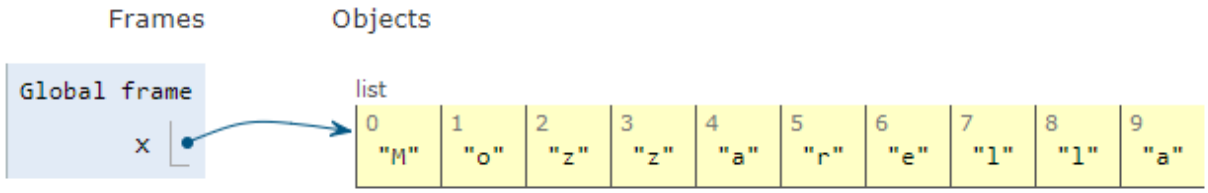
>>> 4



List Methods: *insert*()

	0	1	2	3	4	5	6	7	8	9	
[	‘M’,	‘o’,	‘z’,	‘z’,	‘a’,	‘r’,	‘e’,	‘l’,	‘l’,	‘a’	]
	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	

>>> x = [‘M’,‘o’,‘z’,‘z’,‘a’,‘r’,‘e’,‘l’,‘l’,a’]

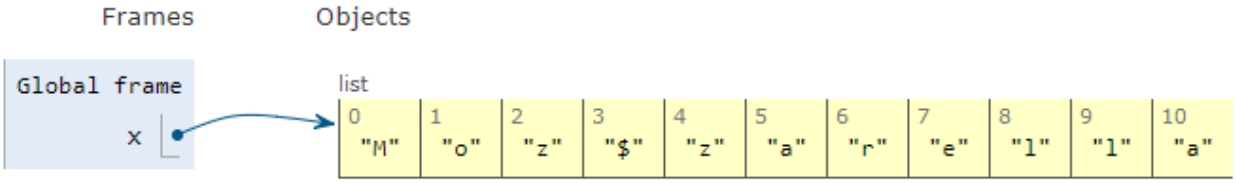


>>> # insert ‘$’ at index 3 in list x

>>> x.*insert*(3, ‘$’)

>>> x

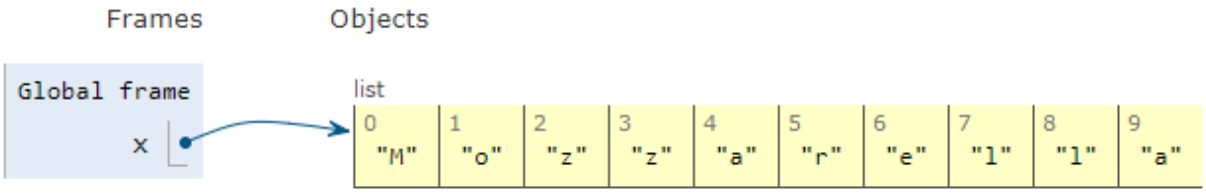
>>> [‘M’,‘o’,‘z’,’$’,’z’,‘a’,‘r’,‘e’,‘l’,‘l’,‘a’]



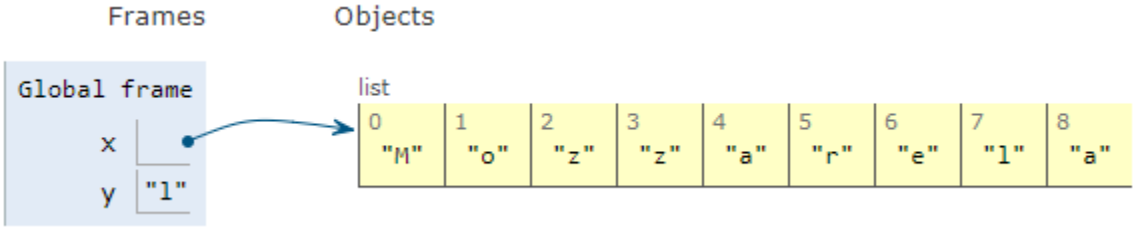
List Methods: *pop*(*i*=−1)

	0	1	2	3	4	5	6	7	8	9	
[	'M',	'o',	'z',	'z',	'a',	'r',	'e',	'l',	'l',	'a'	]
	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	

```
>>> x = ['M','o','z','z','a','r','e','l','l',a']
```



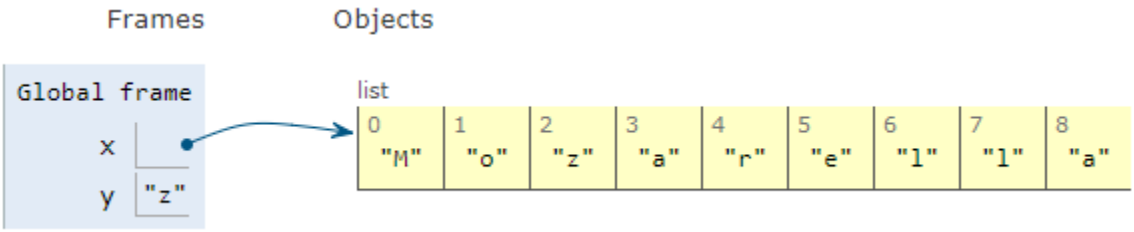

```
>>> # return the value at index i and
>>> # remove this value from the list
>>> y = x.pop(−3)
>>> x
>>> ['M','o','z','z','a','r','e','l','a' ]
```



List Element Removal

	0	1	2	3	4	5	6	7	8	9	
[	‘M’,	‘o’,	‘z’,	‘z’,	‘a’,	‘r’,	‘e’,	‘l’,	‘l’,	‘a’	]
	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	

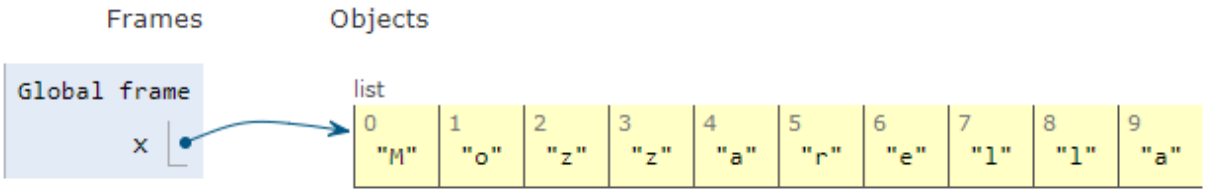
```
>>> x = [‘M’,‘o’,‘z’,‘z’,‘a’,‘r’,‘e’,‘l’,‘l’,a’]
>>> y = x.pop(2)
>>> y
>>> z
```



List Methods: *remove*()

	0	1	2	3	4	5	6	7	8	9	
[	‘M’,	‘o’,	‘z’,	‘z’,	‘a’,	‘r’,	‘e’,	‘l’,	‘l’,	‘a’	]
	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	

>>> x = [‘M’,‘o’,‘z’,‘z’,‘a’,‘r’,‘e’,‘l’,‘l’,a’]

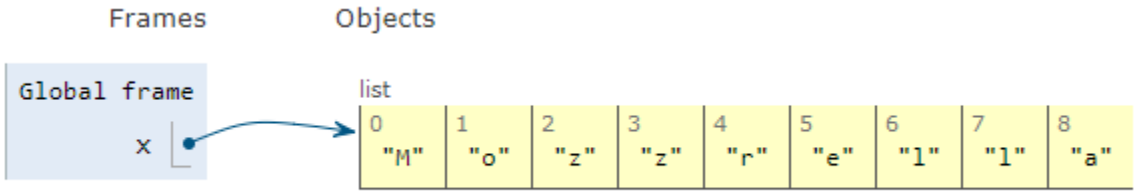


>>> # remove first ‘a’ from list x

>>> x.*remove*(‘a’)

>>> x

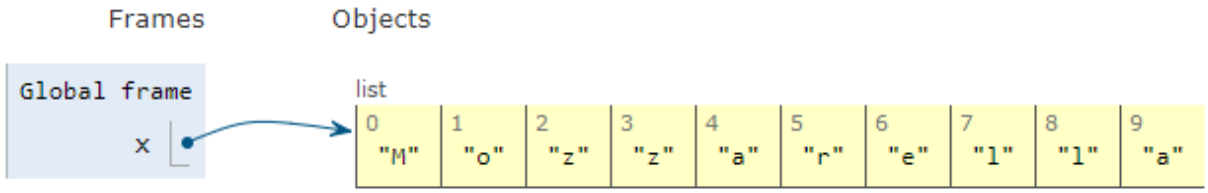
>>> [‘M’,‘o’,‘z’,’z’,‘r’,‘e’,‘l’,‘l’,‘a’]



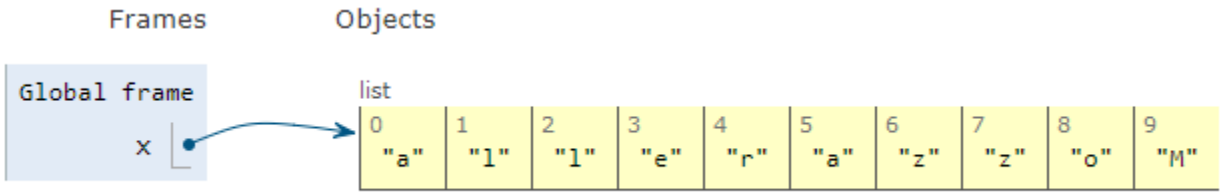
List Methods: *reverse*()

	0	1	2	3	4	5	6	7	8	9	
[	‘M’,	‘o’,	‘z’,	‘z’,	‘a’,	‘r’,	‘e’,	‘l’,	‘l’,	‘a’	]
	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	

```
>>> x = [‘M’,‘o’,‘z’,‘z’,‘a’,‘r’,‘e’,‘l’,‘l’,a’]
```



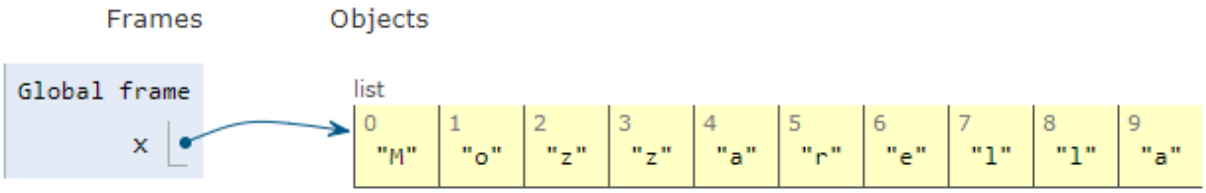

```
>>> # reverse list x in-place
>>> x.reverse()
>>> x
>>> [‘a,‘l’,‘l’,’e’,‘r’,‘a’,‘z’,‘z’,‘o’,‘M’]
```



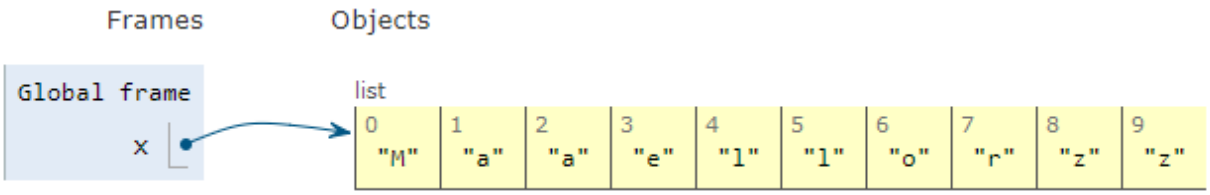
List Methods: *sort*()

	0	1	2	3	4	5	6	7	8	9	
[	'M',	'o',	'z',	'z',	'a',	'r',	'e',	'l',	'l',	'a'	]
	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1	

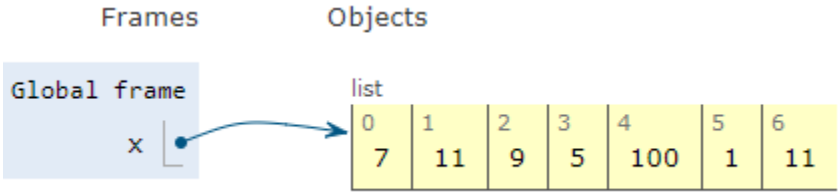
```
>>> x = ['M','o','z','z','a','r','e','l','l',a']
```



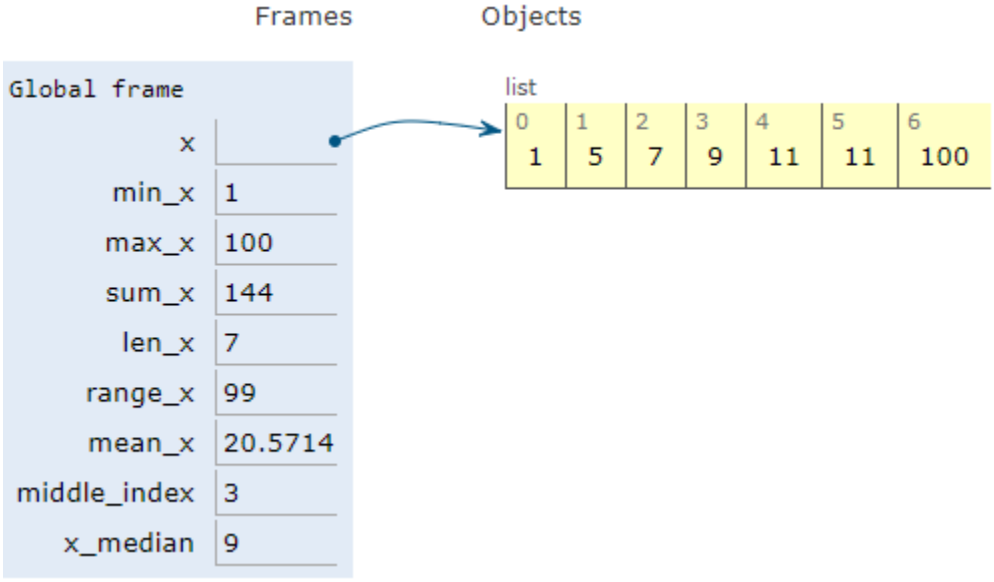

```
>>> # sort list x in-place
>>> x.sort()
>>> x
>>> ['M','a,'a','e','l','l','o','r','r','z']
```



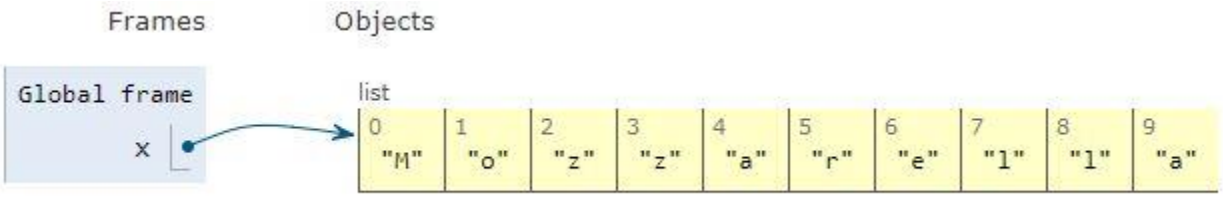
min(), *max*(), *sum*()



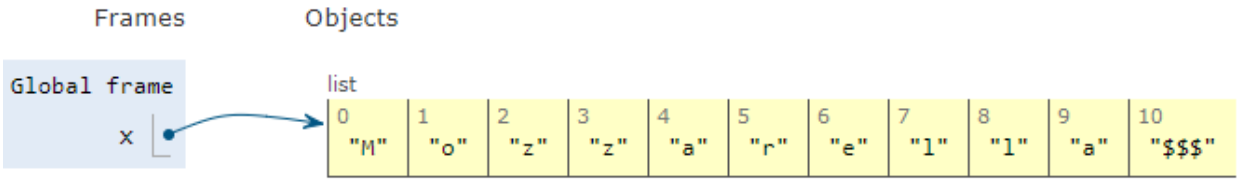

```
>>> x.sort()
>>> min_x = min(x); max_x = max(x)
>>> sum_x = sum(x); len_x = len(x)
>>> range_x = max_x – min_x
>>> mean_x = float(sum_x)/len_x
>>> middle_index = len_x // 2
>>> x_median = x[middle_index]
```



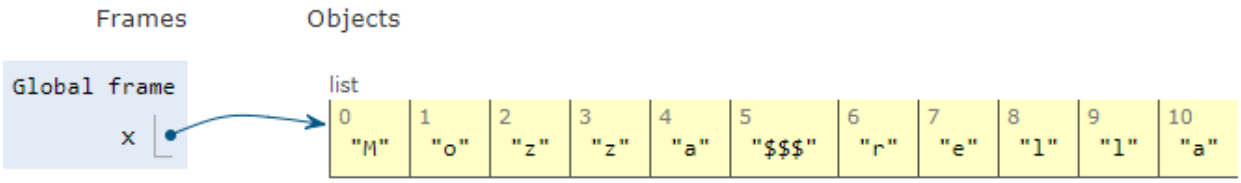
Common List Methods



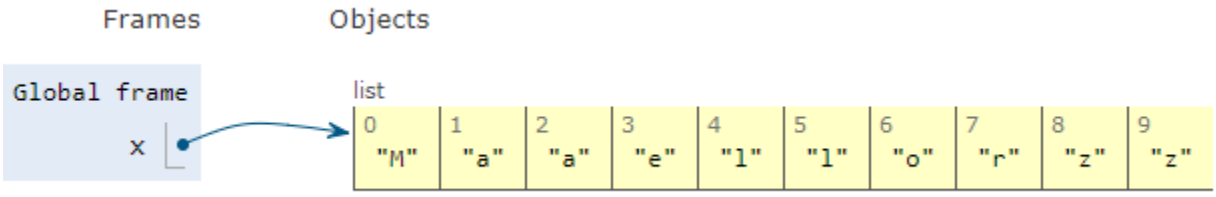

```
>>> x.append('$')
```



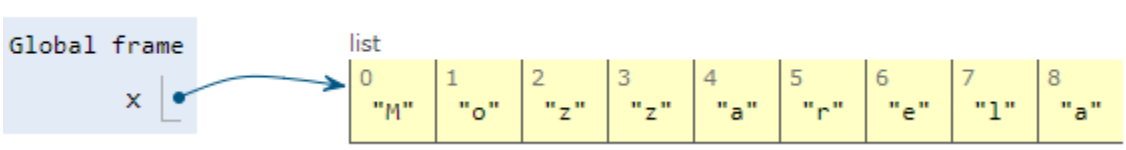

```
>>> x.insert(5, '$')
```




```
>>> x.sort()
```




```
>>> x.remove('l')
```



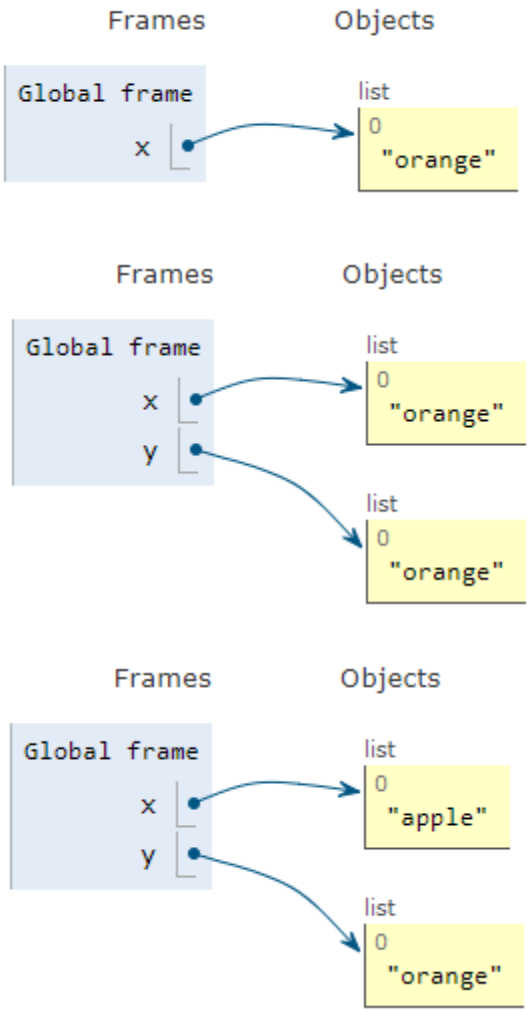
List Copy: Quiz

- what is the result of last print statement?

```
>>> x = [‘orange’]
>>> print(len(x))
>>> 1
>>> print(x[0])
>>> orange
>>> y = x
>>> x[0] = ‘apple’
>>> print(x[0], y[0])
```

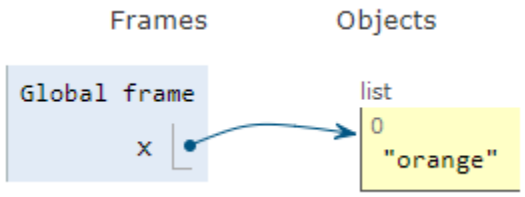
List Copy: Quiz (cont'd)

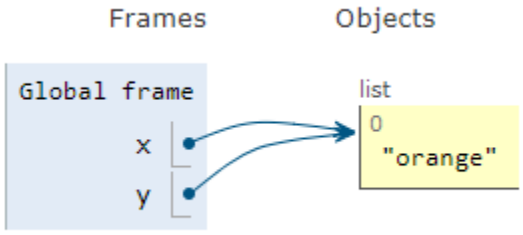
```
>>> x = ['orange']
>>> y = x
>>> x[0] = 'apple'
>>> print(x[0], y[0])
>>> ('apple', 'orange')
```

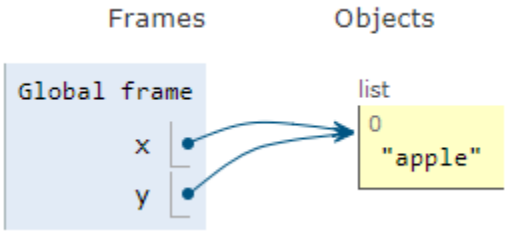


List Copy: Quiz (cont'd)

```
>>> x = ['orange']
>>> y = x
>>> x[0] = 'apple'
>>> print(x[0], y[0])
>>> ('apple', 'apple')
```







Review Problems

Interview Problem

- explain the difference between list *append*() and list *extend*() methods

Interview Problem

- what is *list comprehension*?

Interview Problem

- given the list x = [“One”, “Two”, “Three”]
- convert x to string “One, Two, Three”
- show two different ways to do so

Interview Problem

- how to count the occurrences of each item present in the list without explicitly mentioning them?

Interview Problem

- sort integers in list x

```
>>> x = [6, 9, 3, 91, 54, 122, 3]
```

Interview Problem

- given the list with odd and even numbers

```
>>> x = [1,3,8,3,6,8,11,19, 22]
```

- create two lists:
  (1) x_even with even numbers from x
  (2) x_odd with odd numbers from x

Interview Problem

- remove duplicates from a list

Interview Problem

- the following code is supposed to remove numbers less than 5 from list n, but there is a bug. Fix the bug.

```
n = [1,2,5,10,3,100,9,24]

for e in n:
if e < 5:
    n.remove(e)
print n
```

Interview Problem

- remove duplicates from a list

Interview Problem

- given the list
  >>> x = [‘1’, ‘13’, ‘61’, ‘99’, ‘17’]
- convert to list of numbers:
  y = [1, 13, 61, 99, 17]
- show two different ways to do so (including list comprehension)

Interview Problem

- how to generate random numbers?

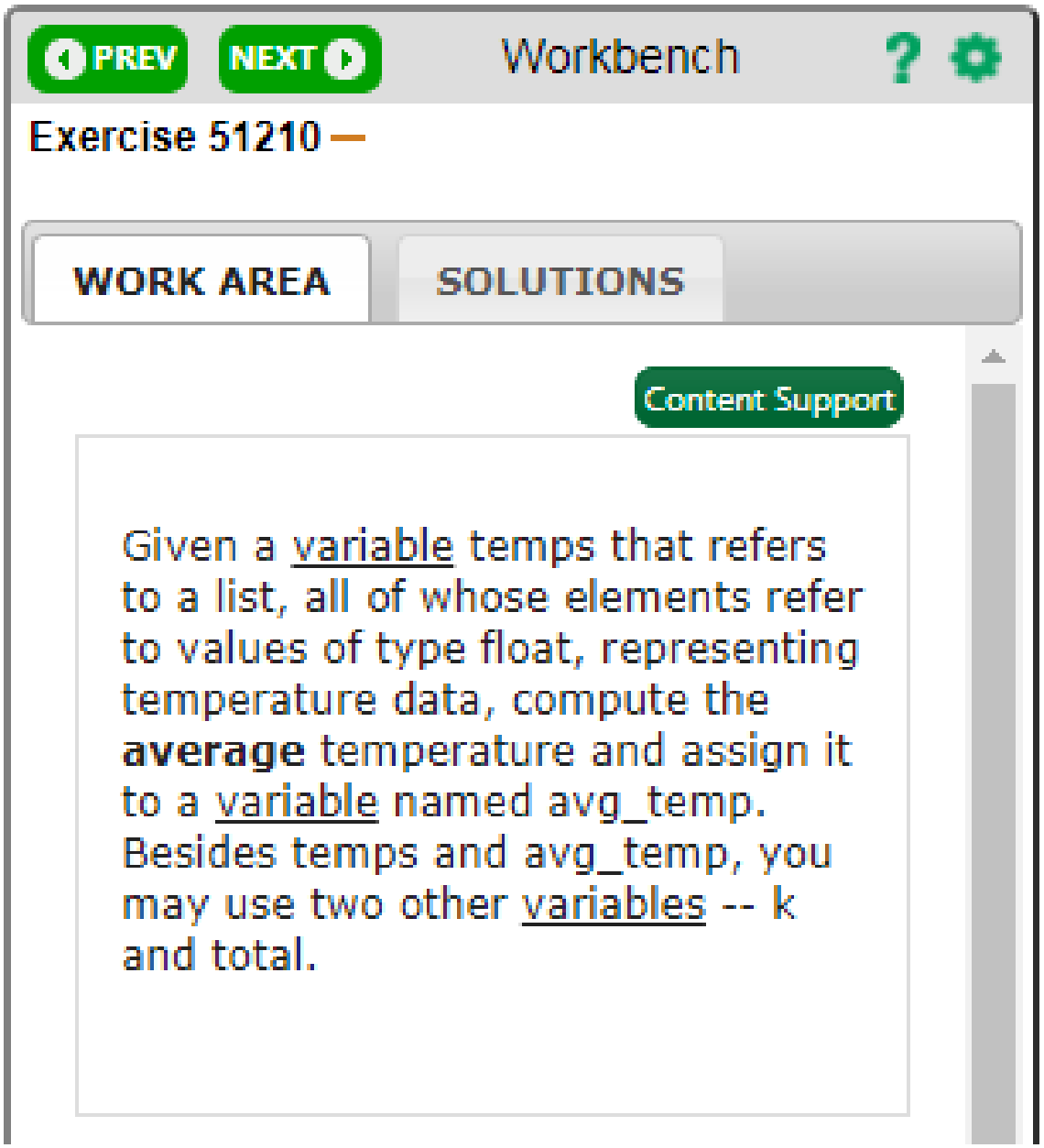
Interview Problem

- what built-in function can be used to iterate over a number sequence?

Interview Problem

- what is the difference between *range*() and *xrange*() functions?

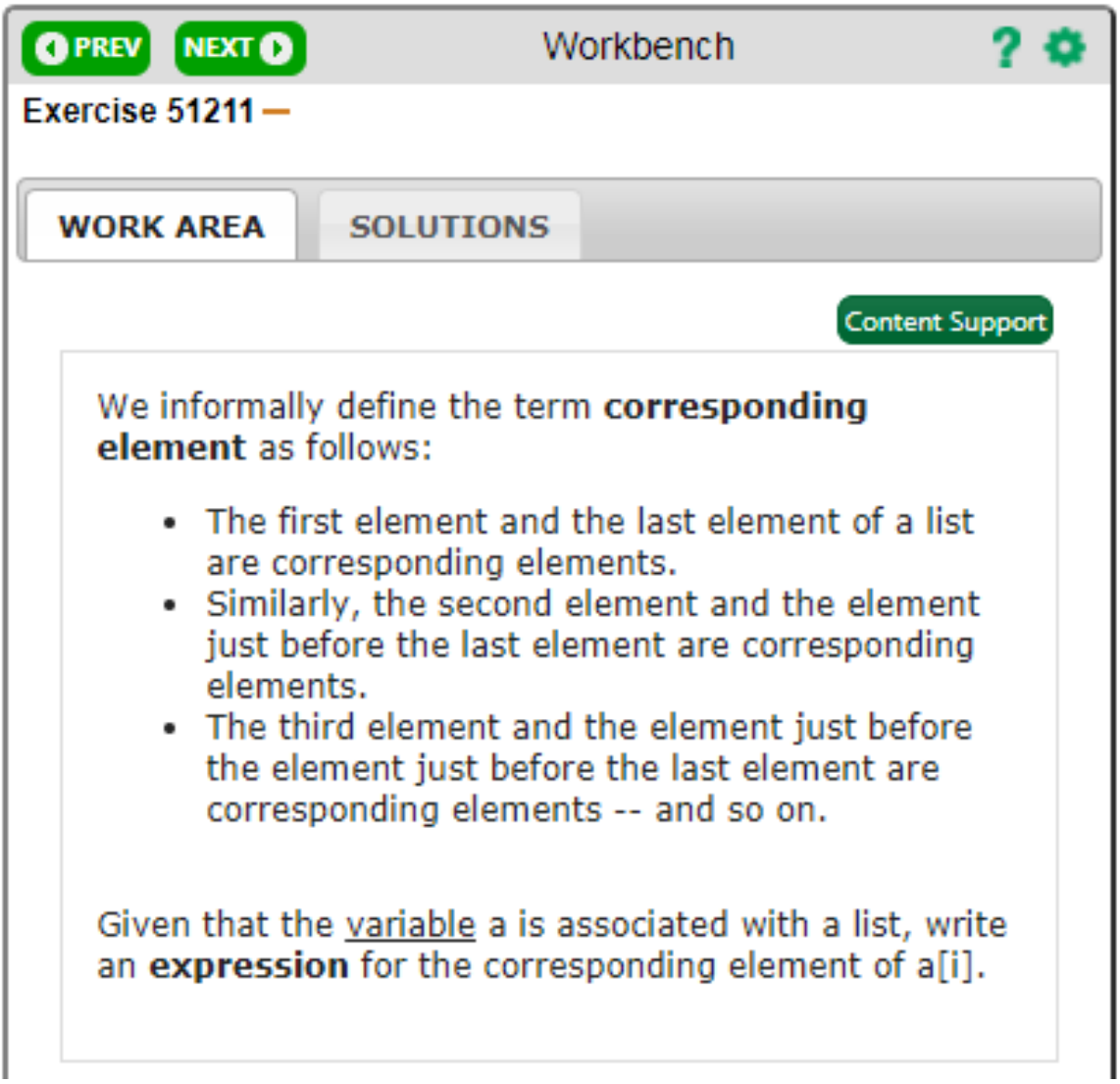
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

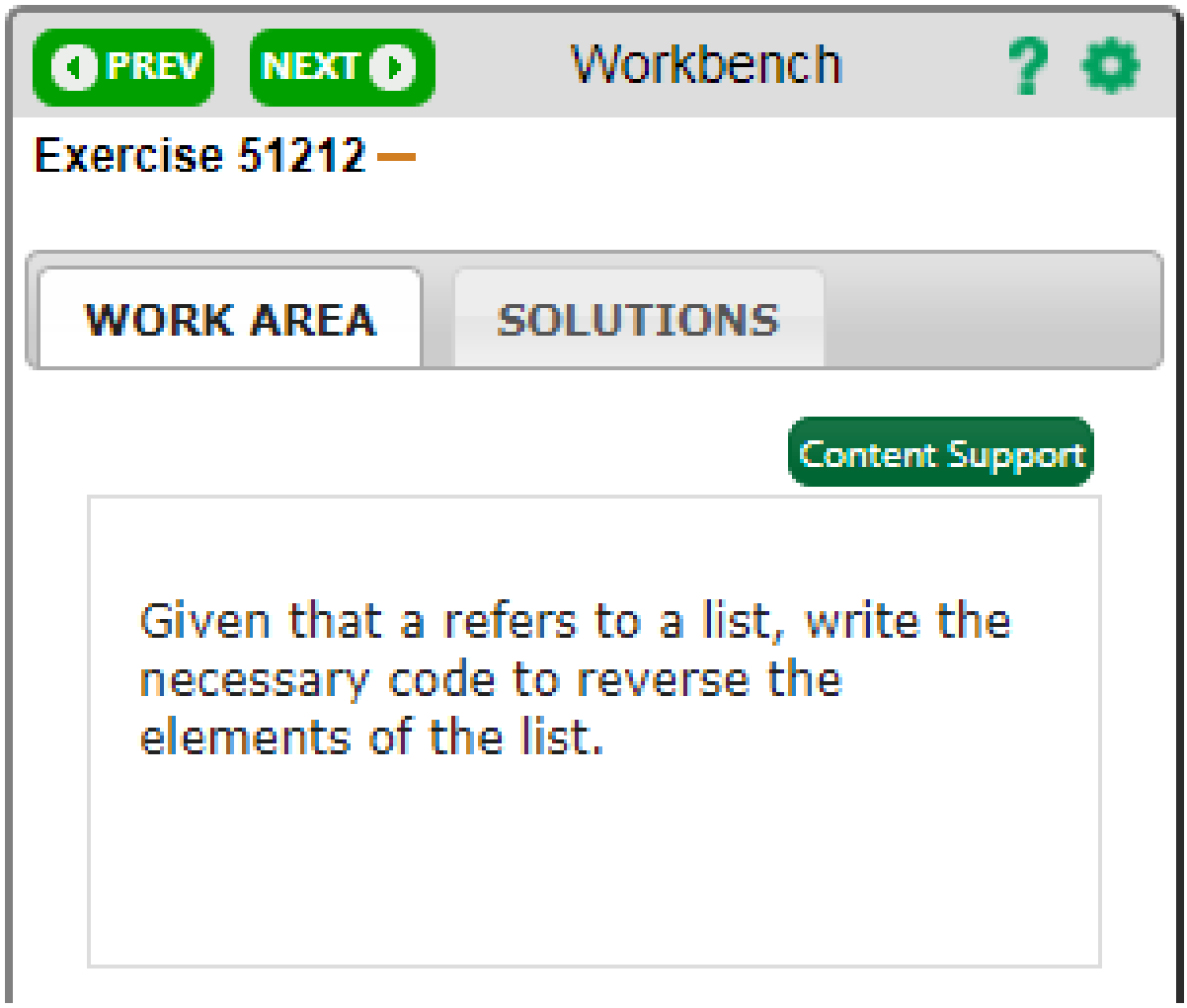
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

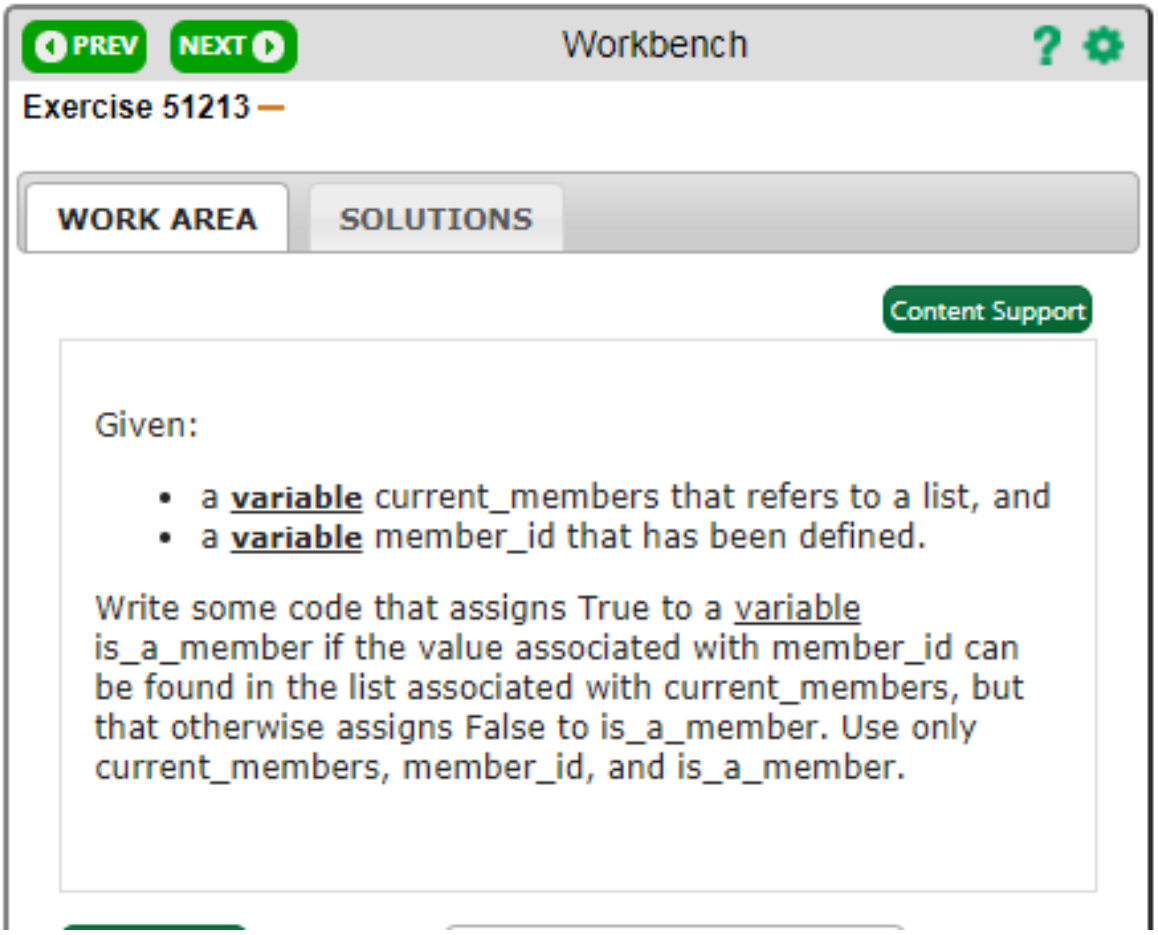
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

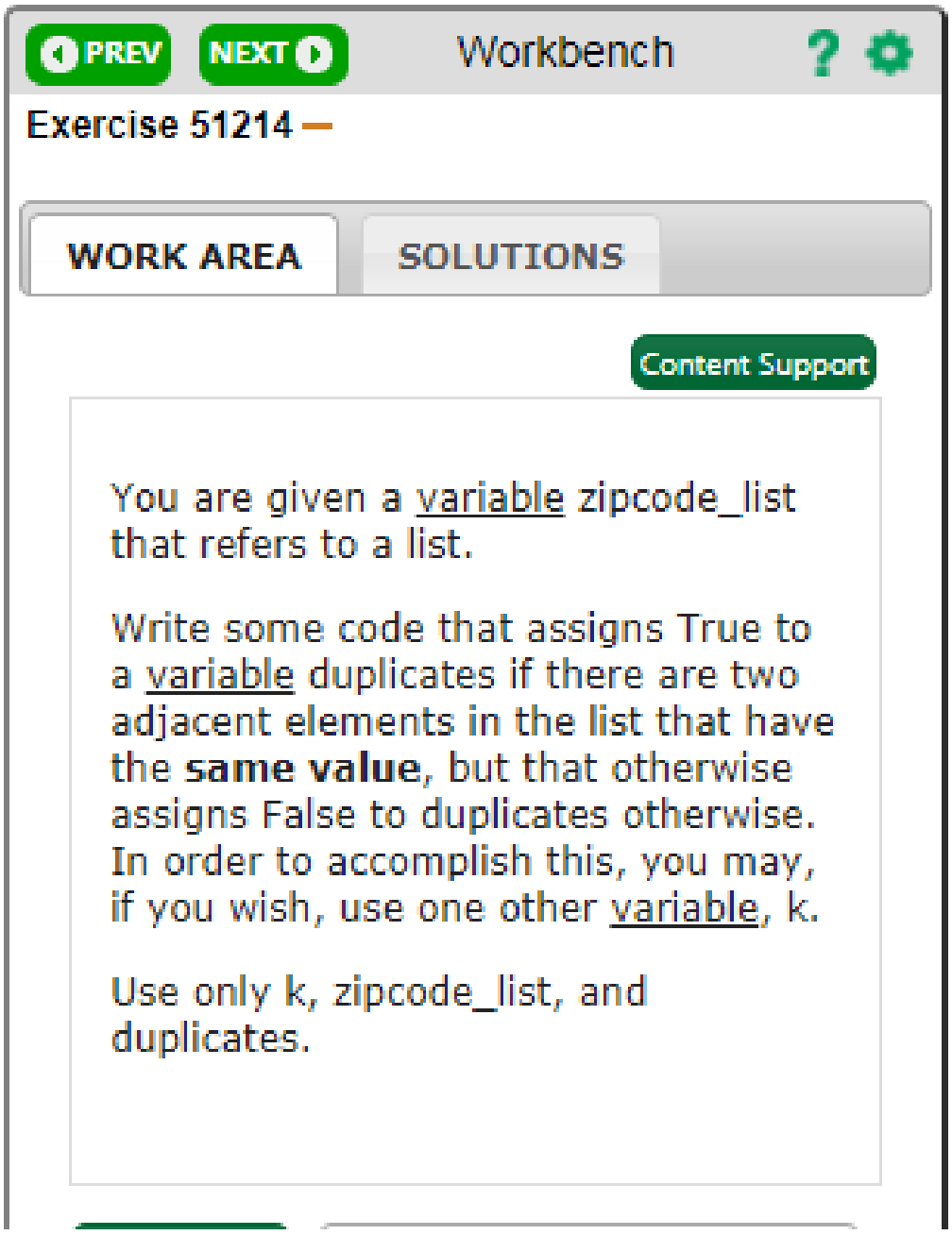
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

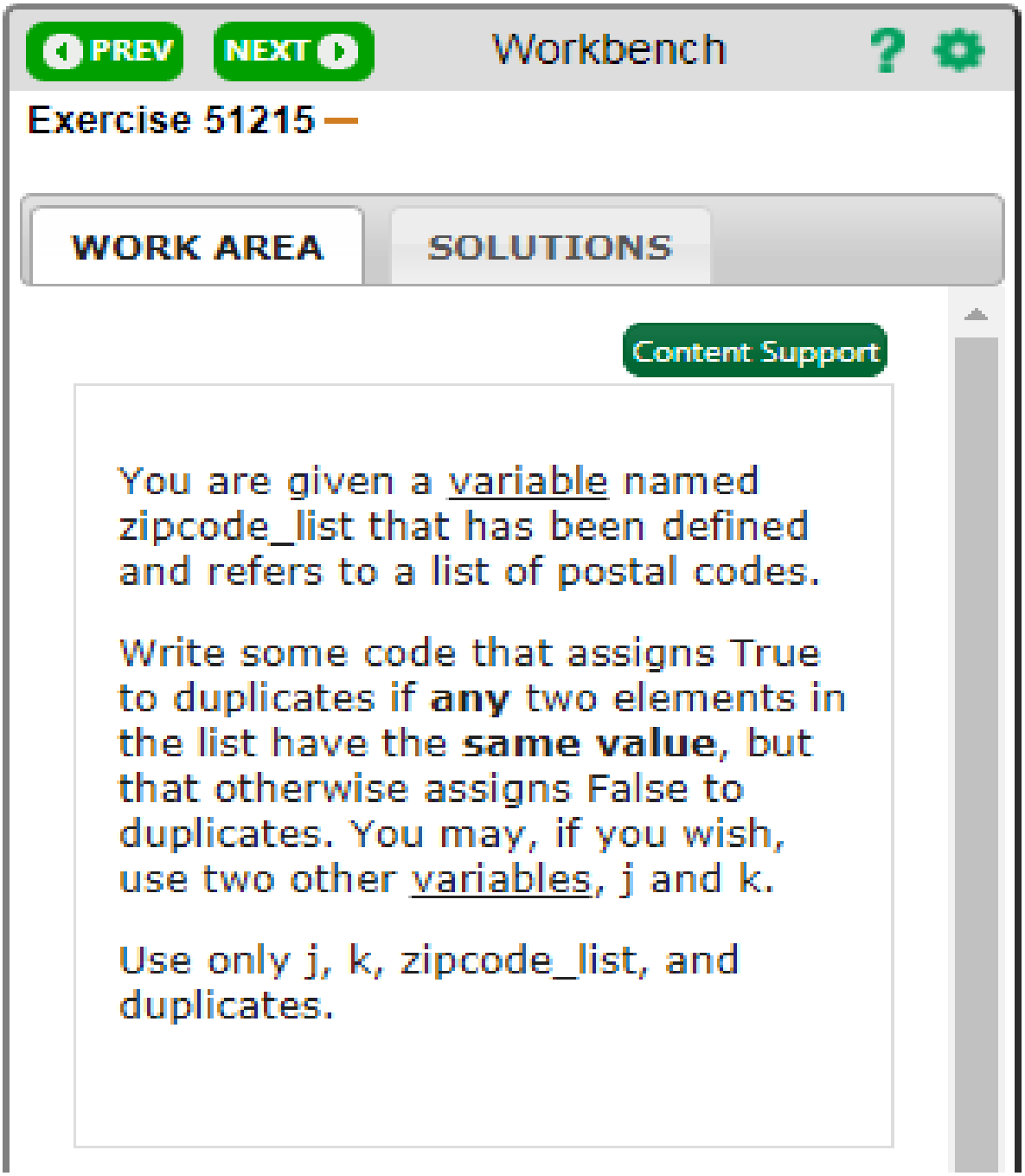
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

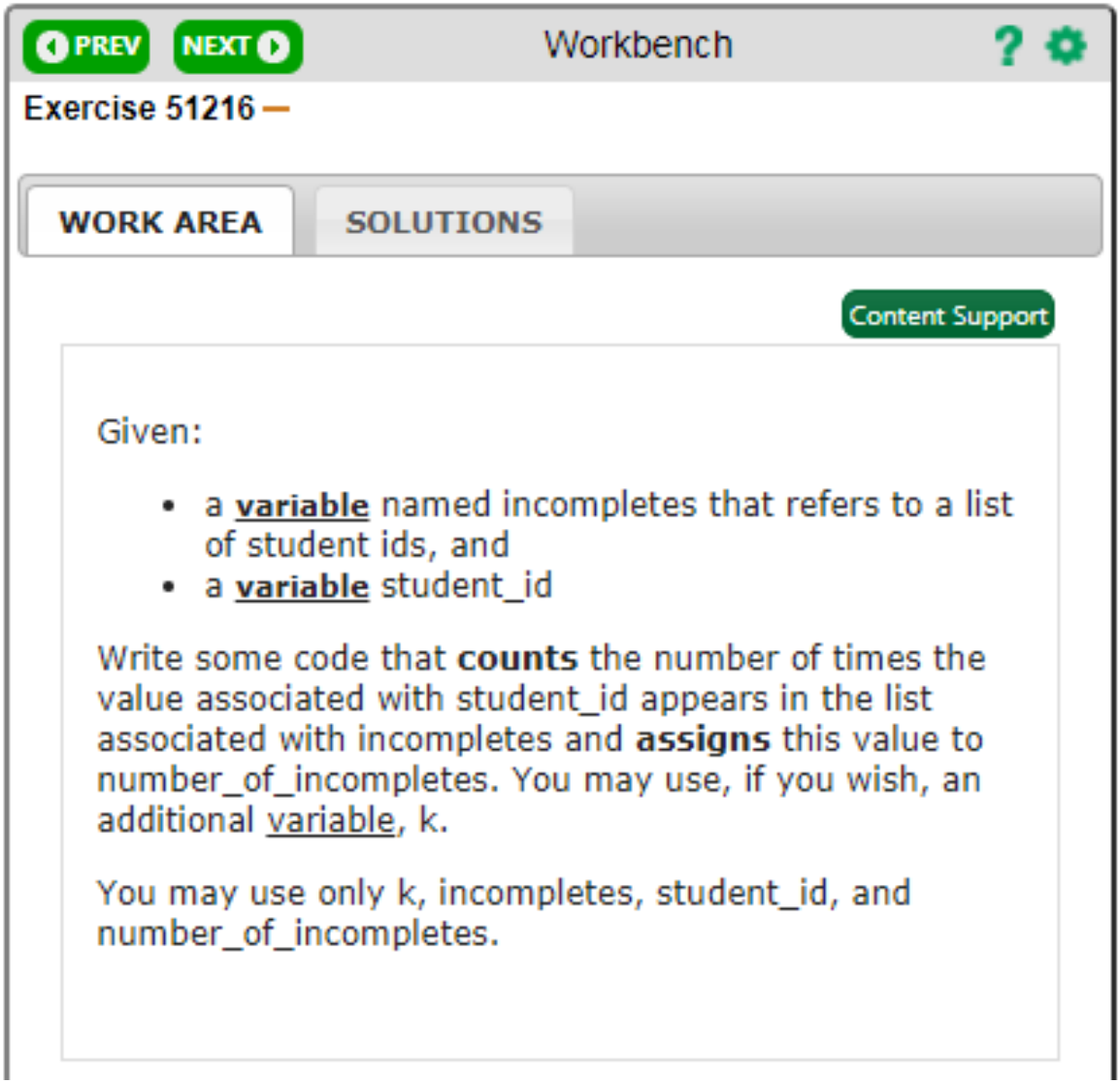
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

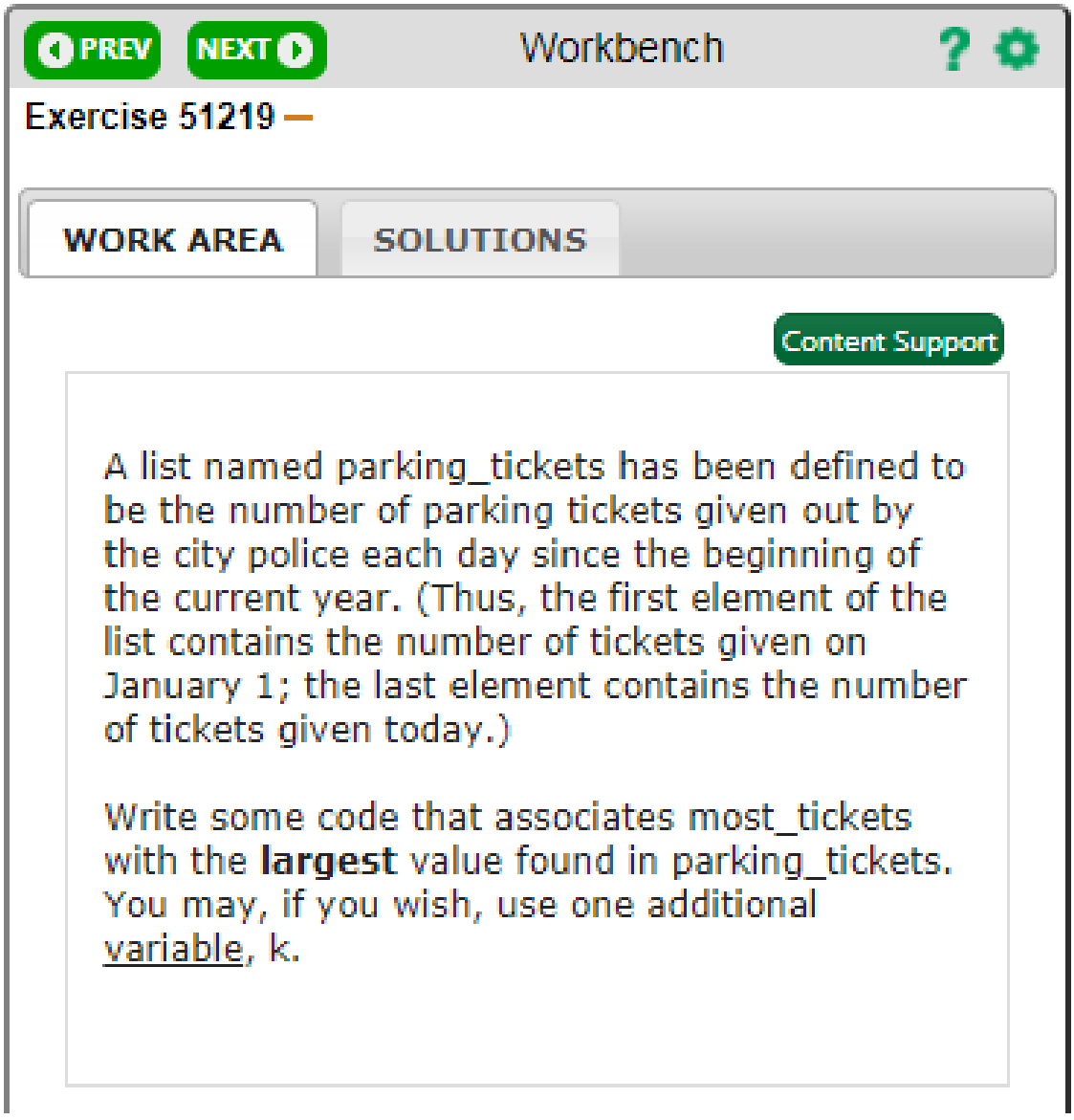
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

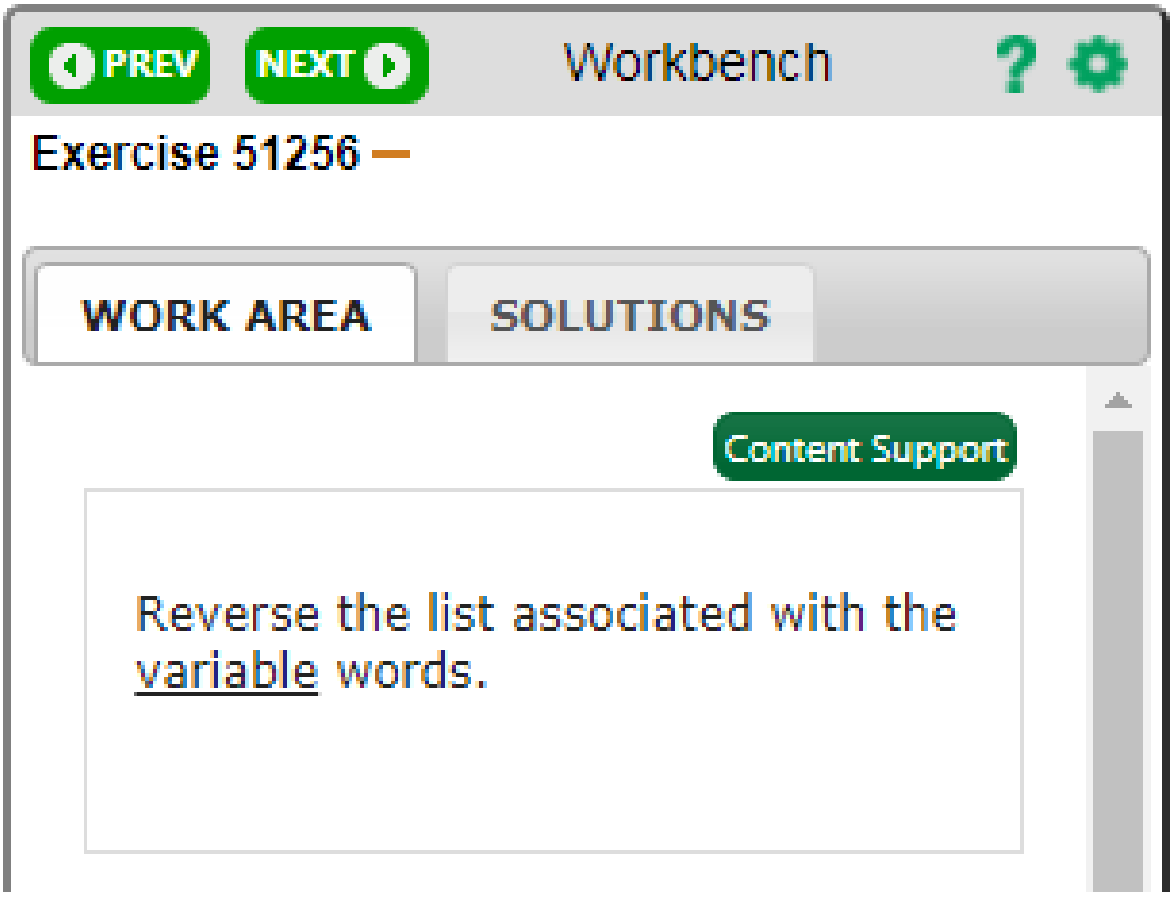
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

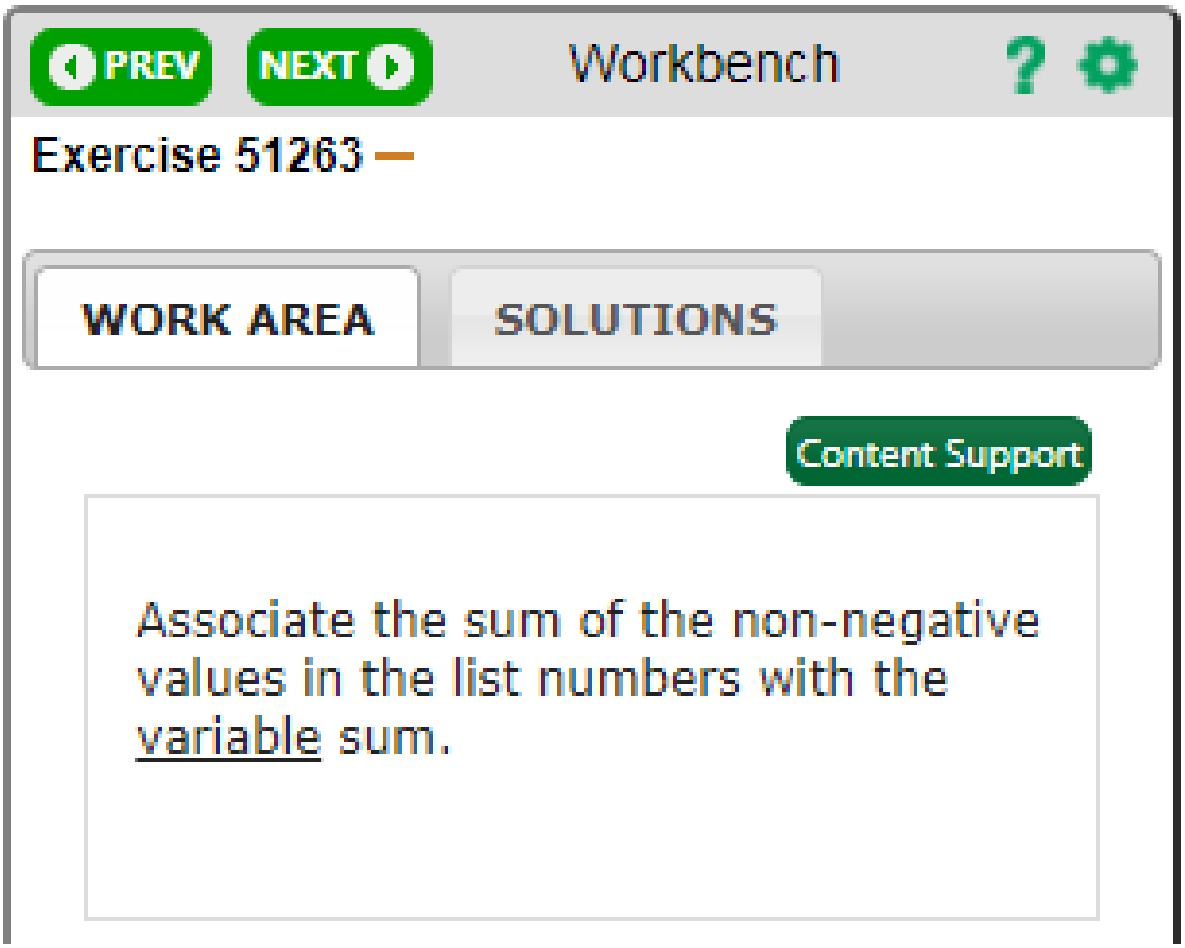
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

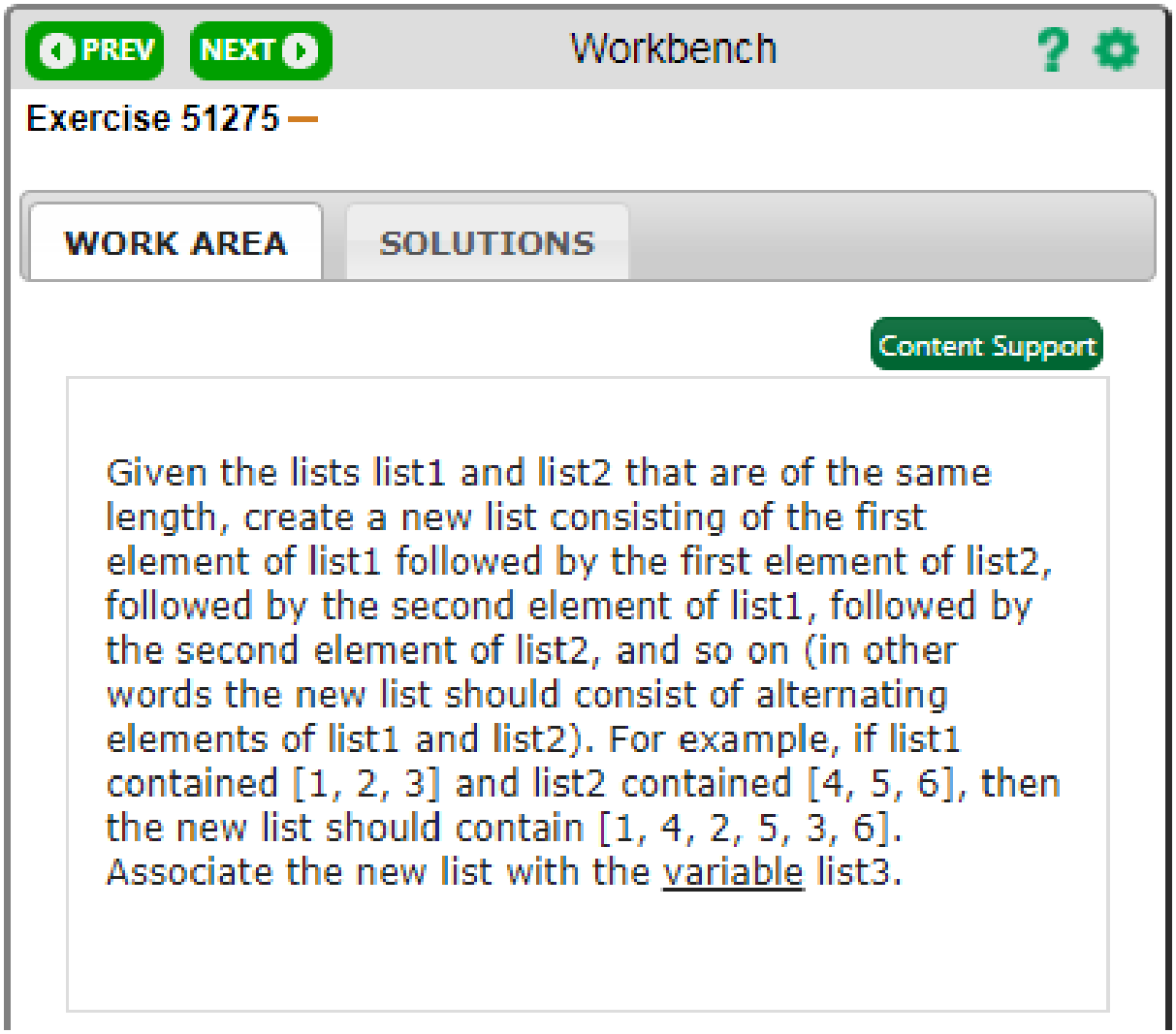
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

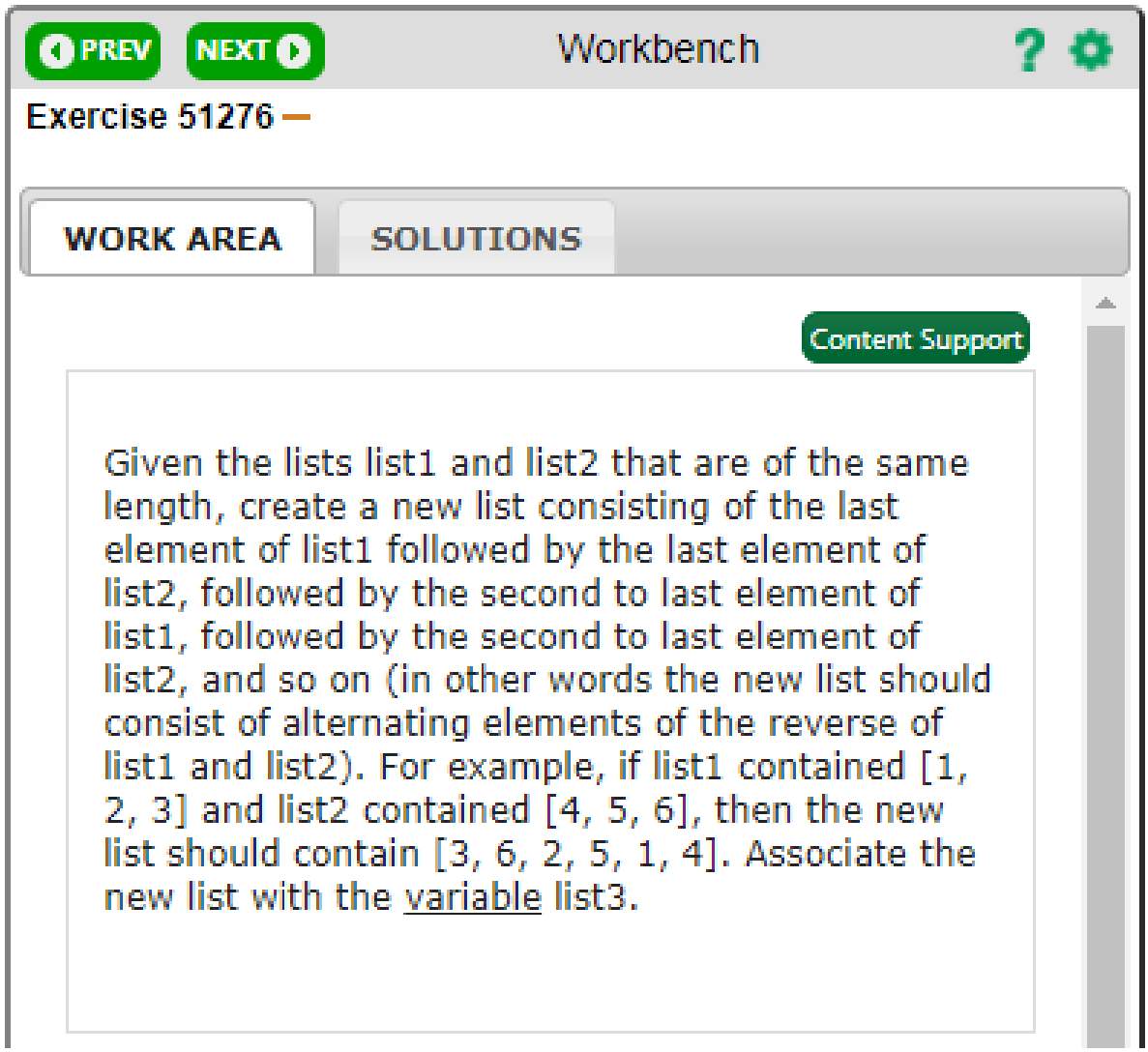
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

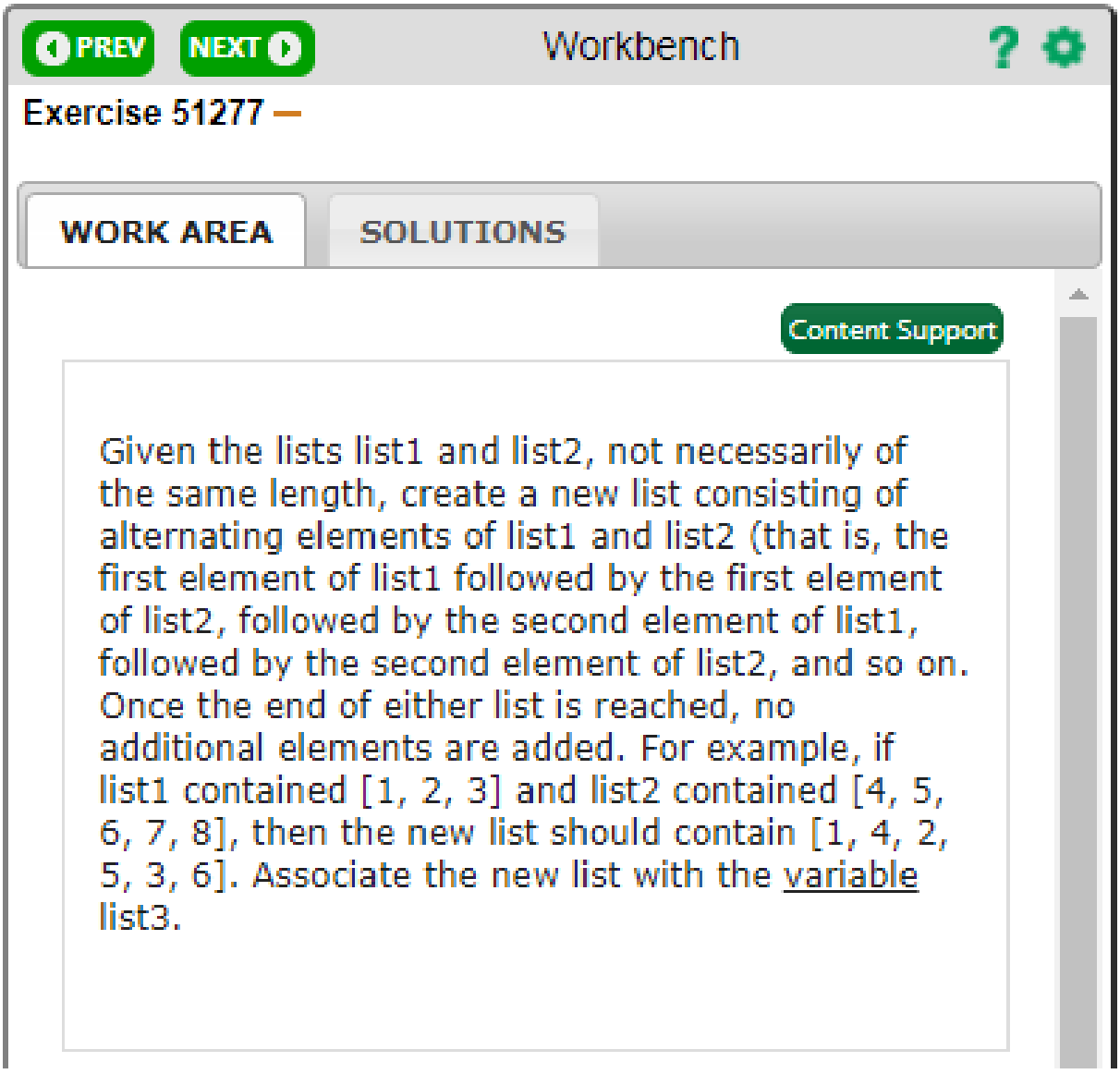
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

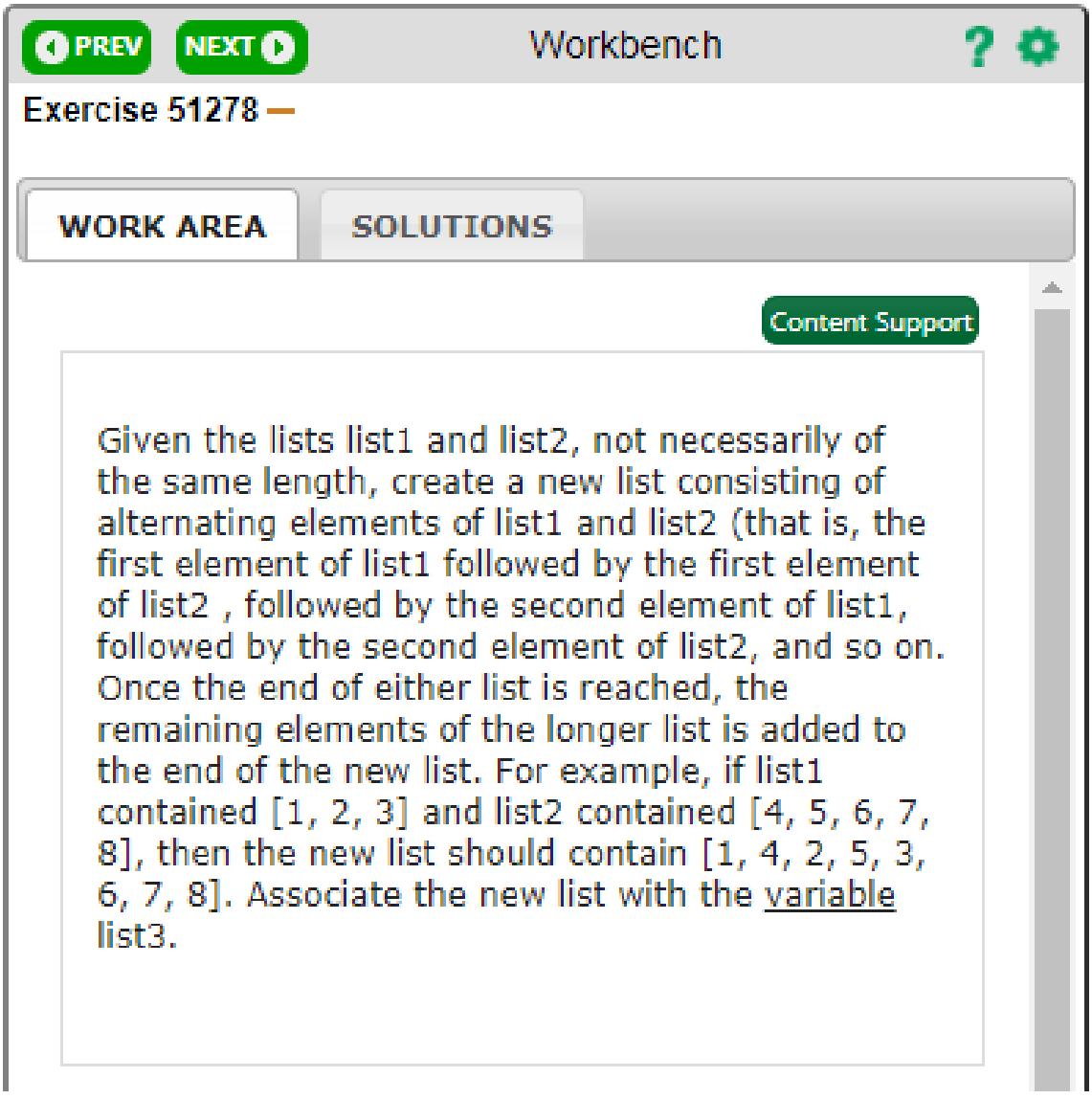
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

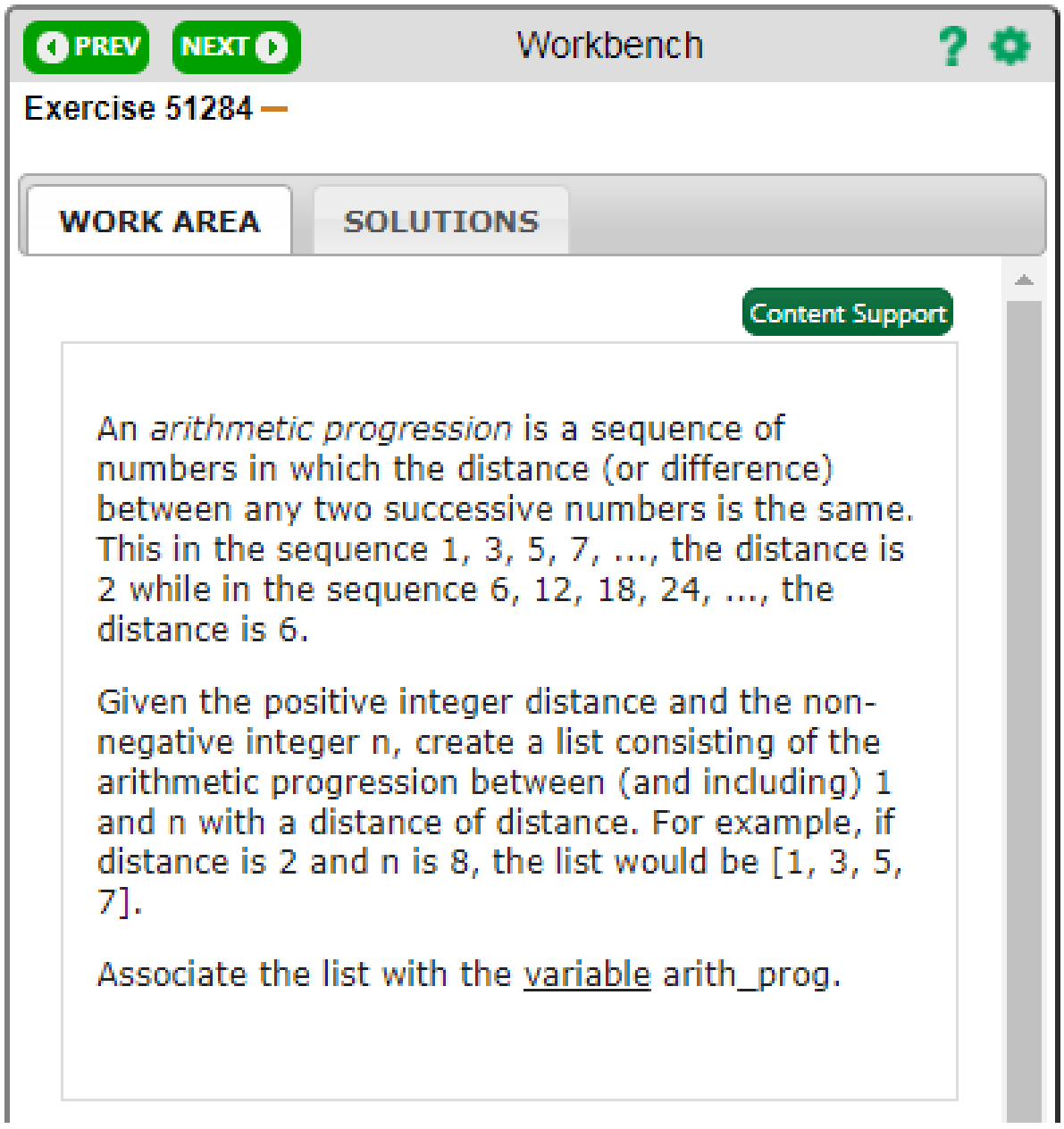
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

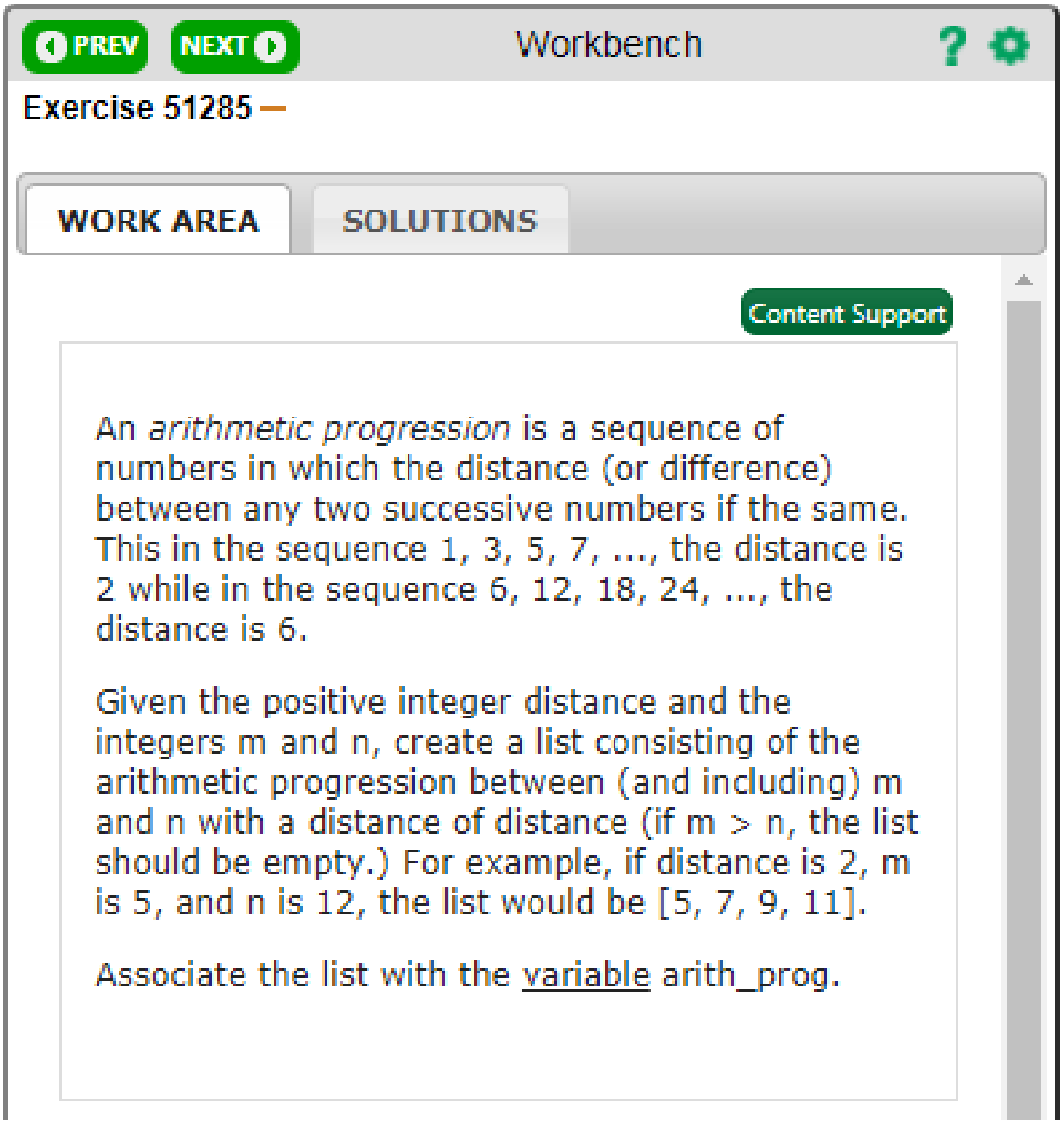
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

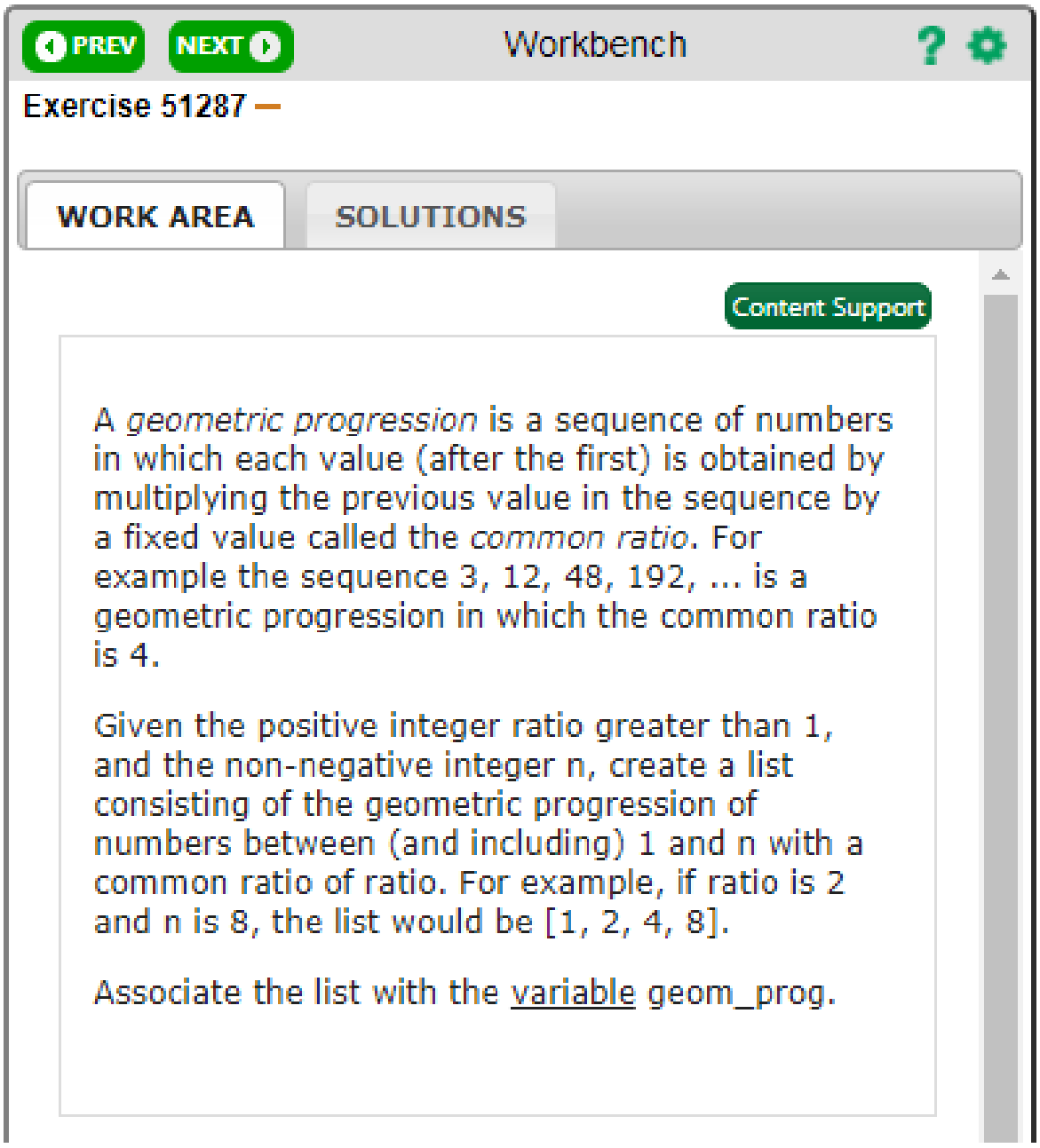
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

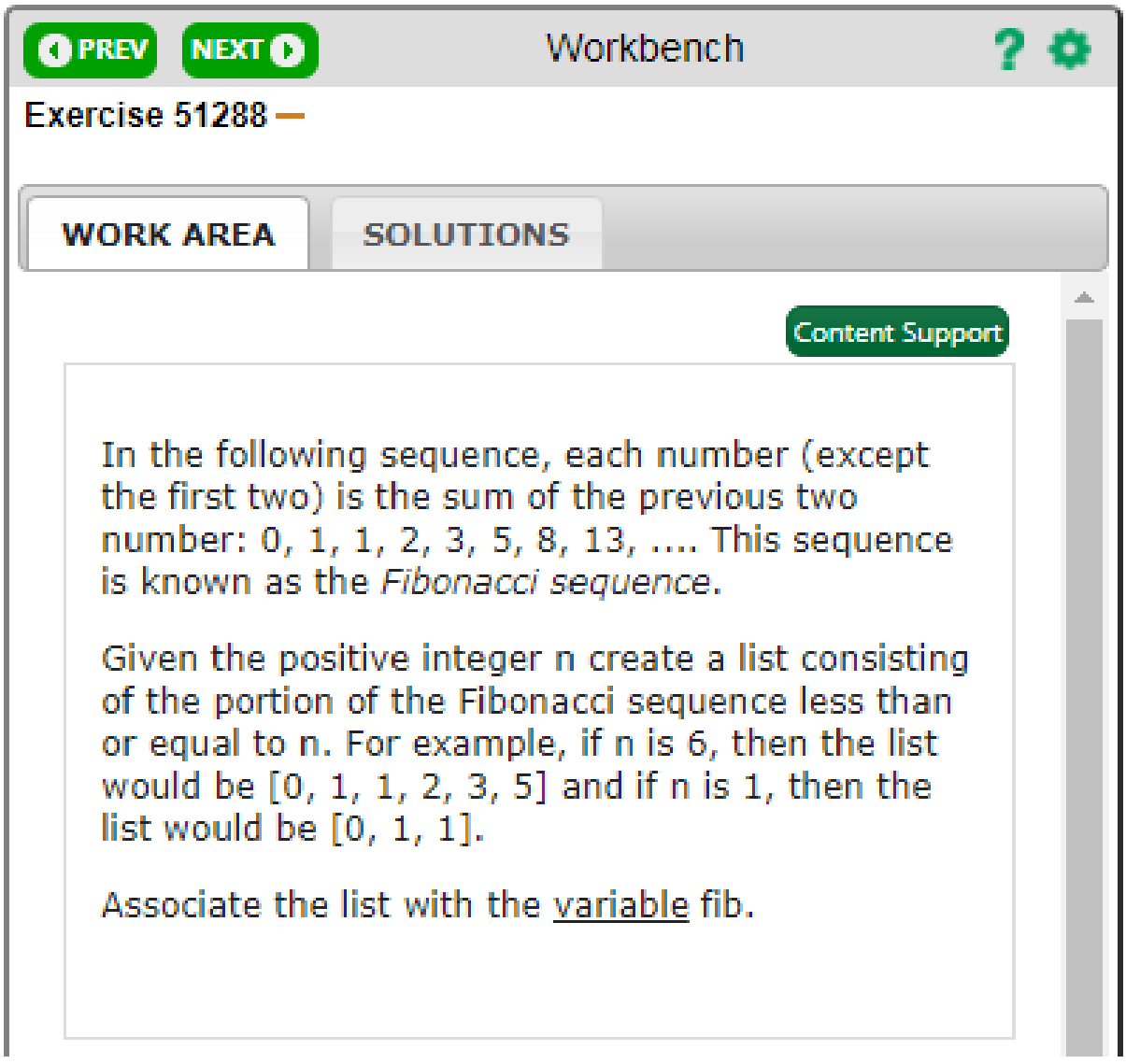
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

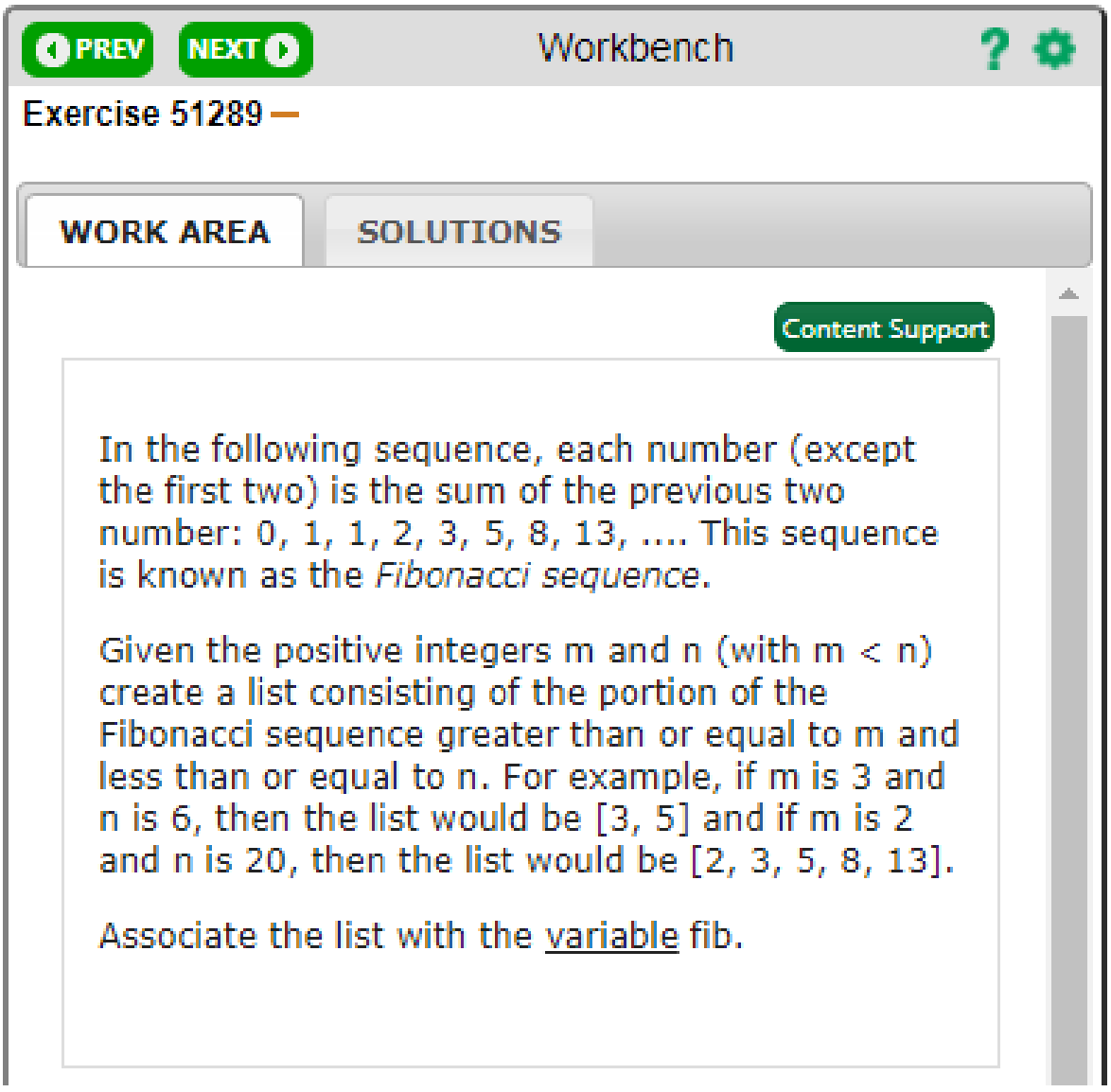
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

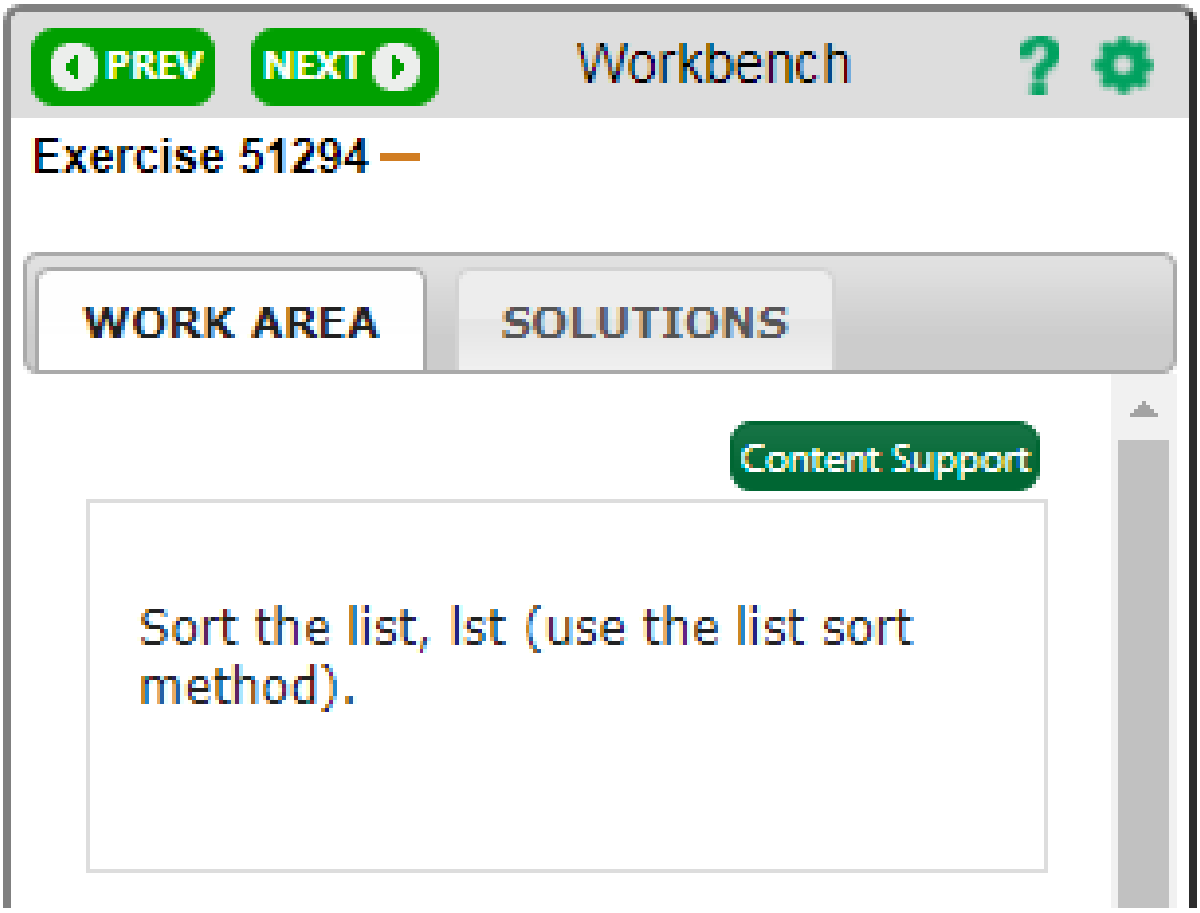
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

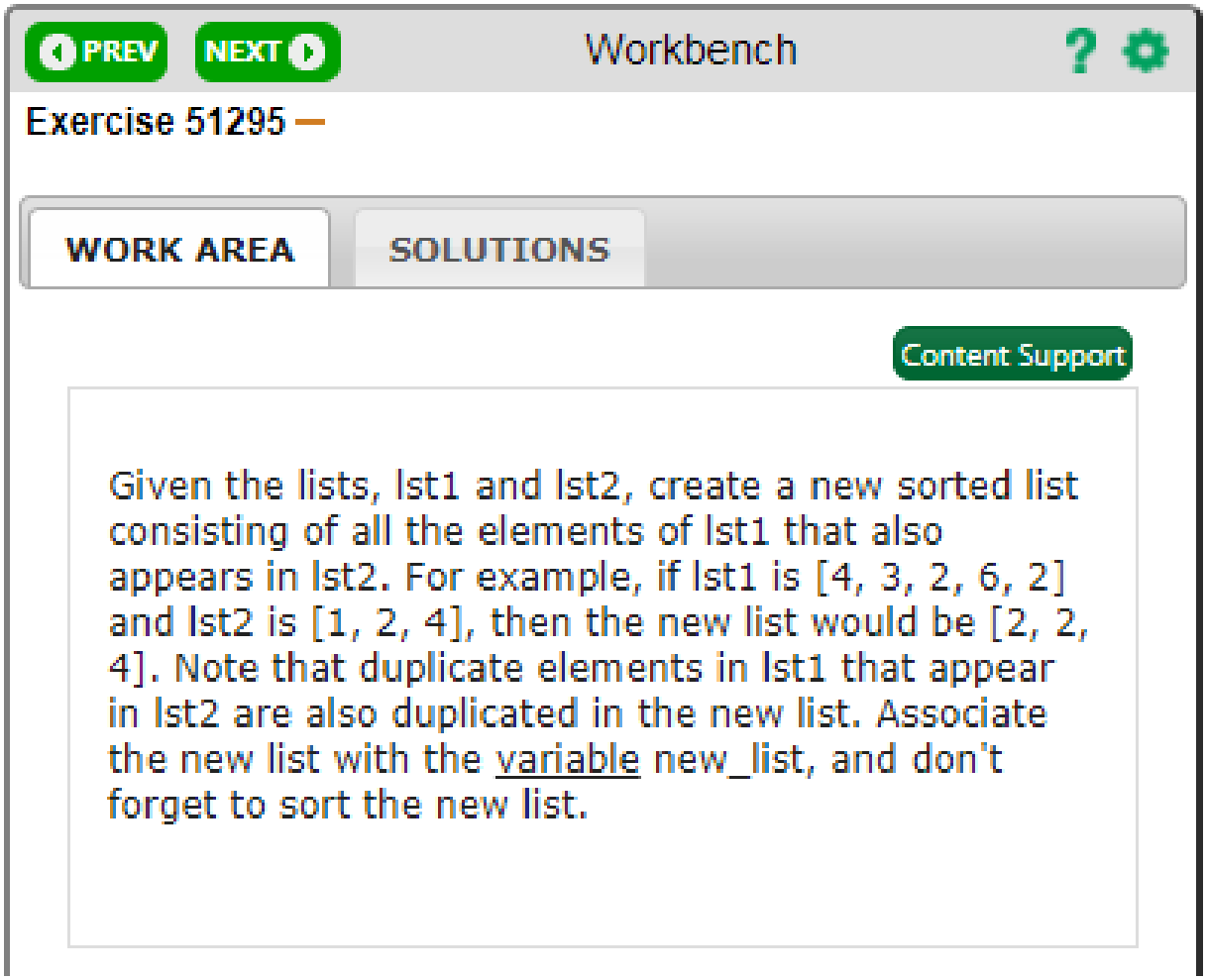
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

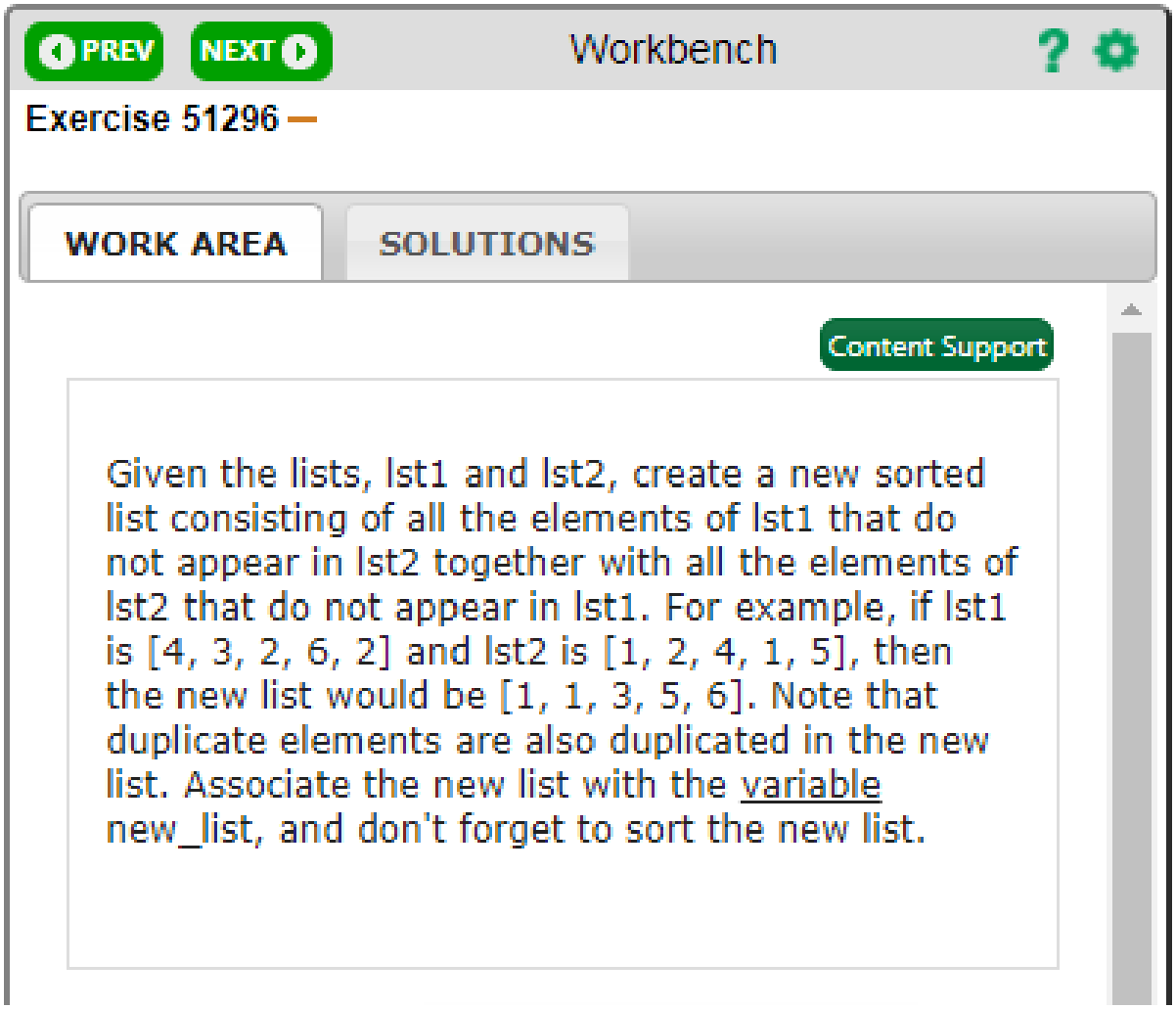
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

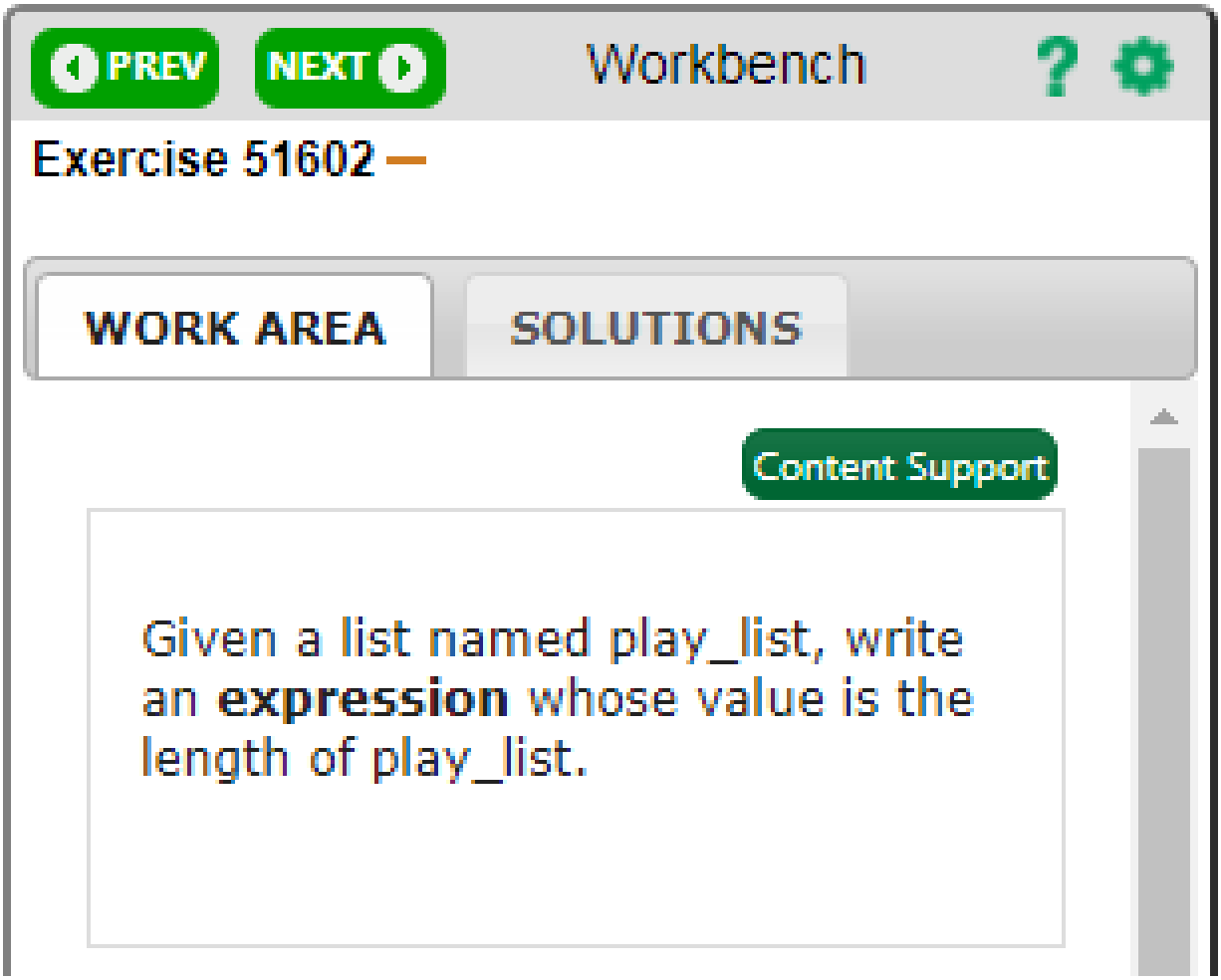
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

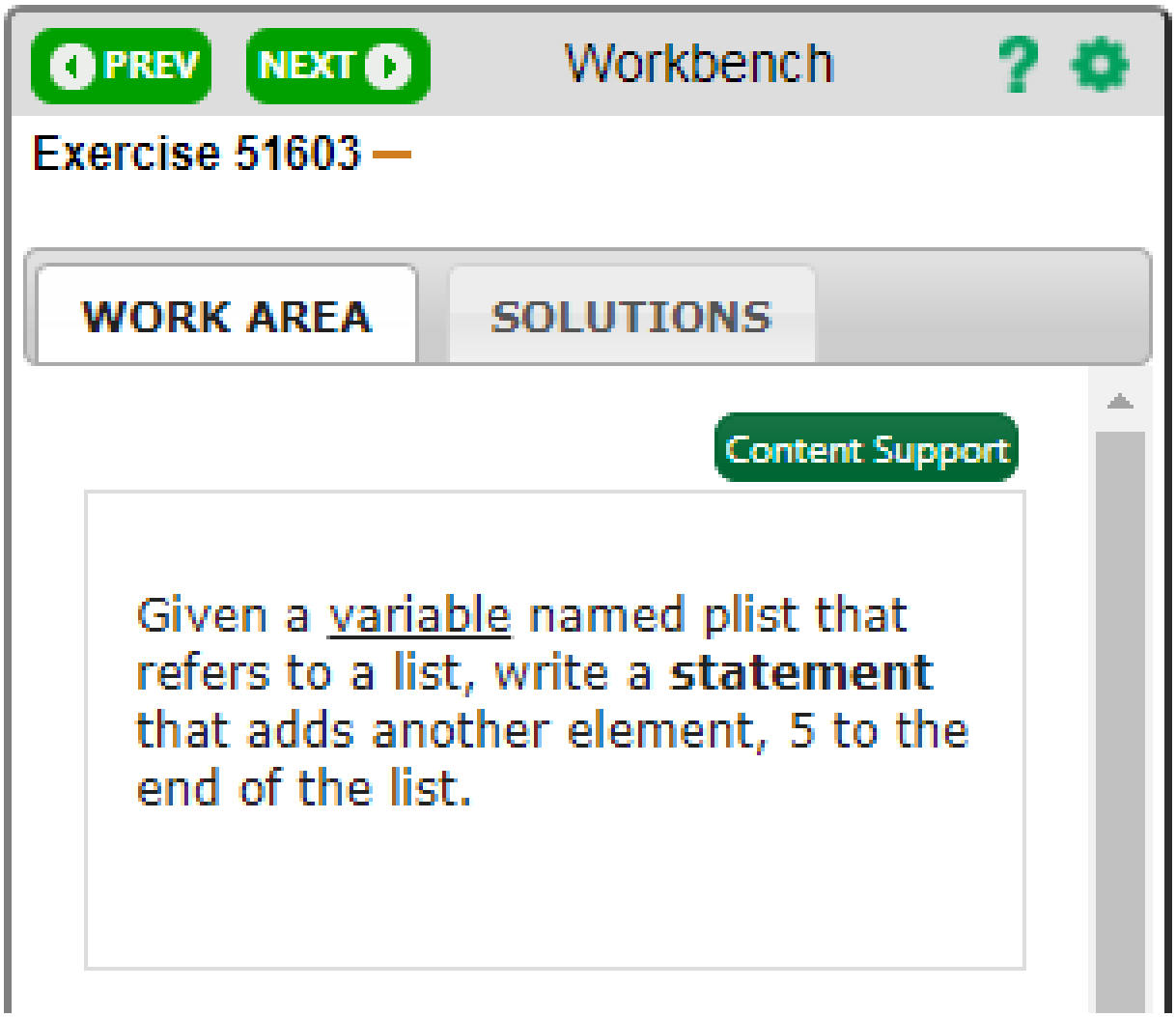
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

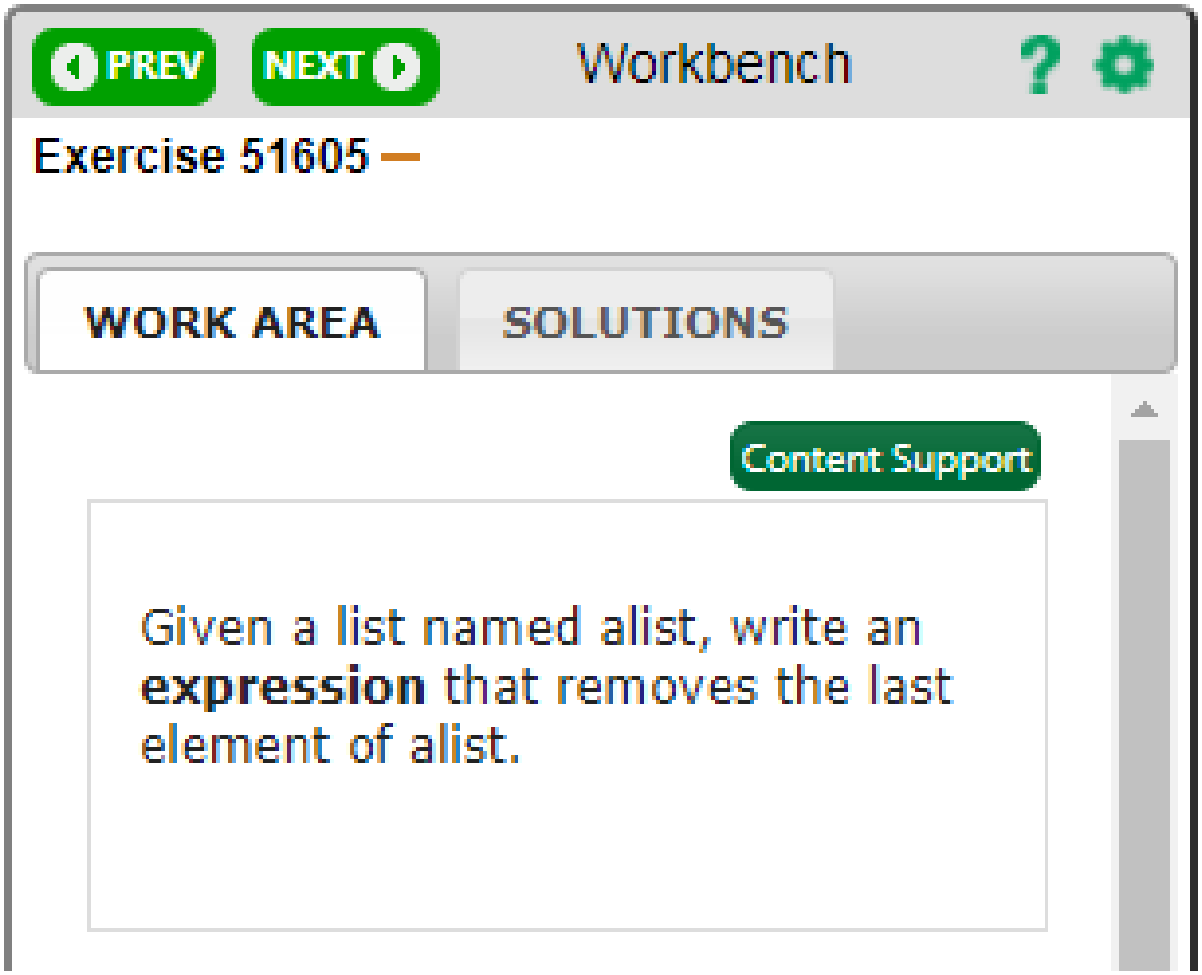
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

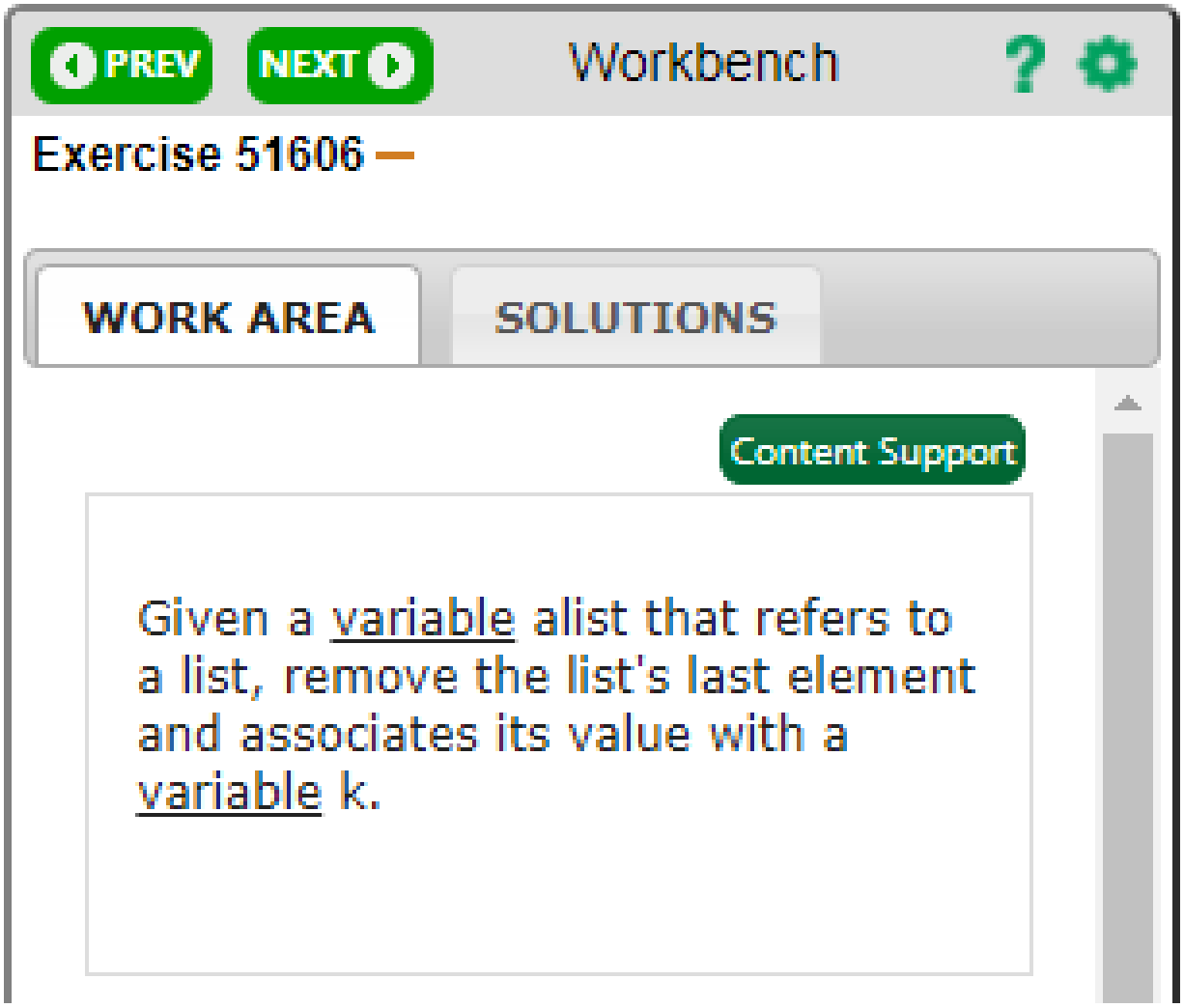
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

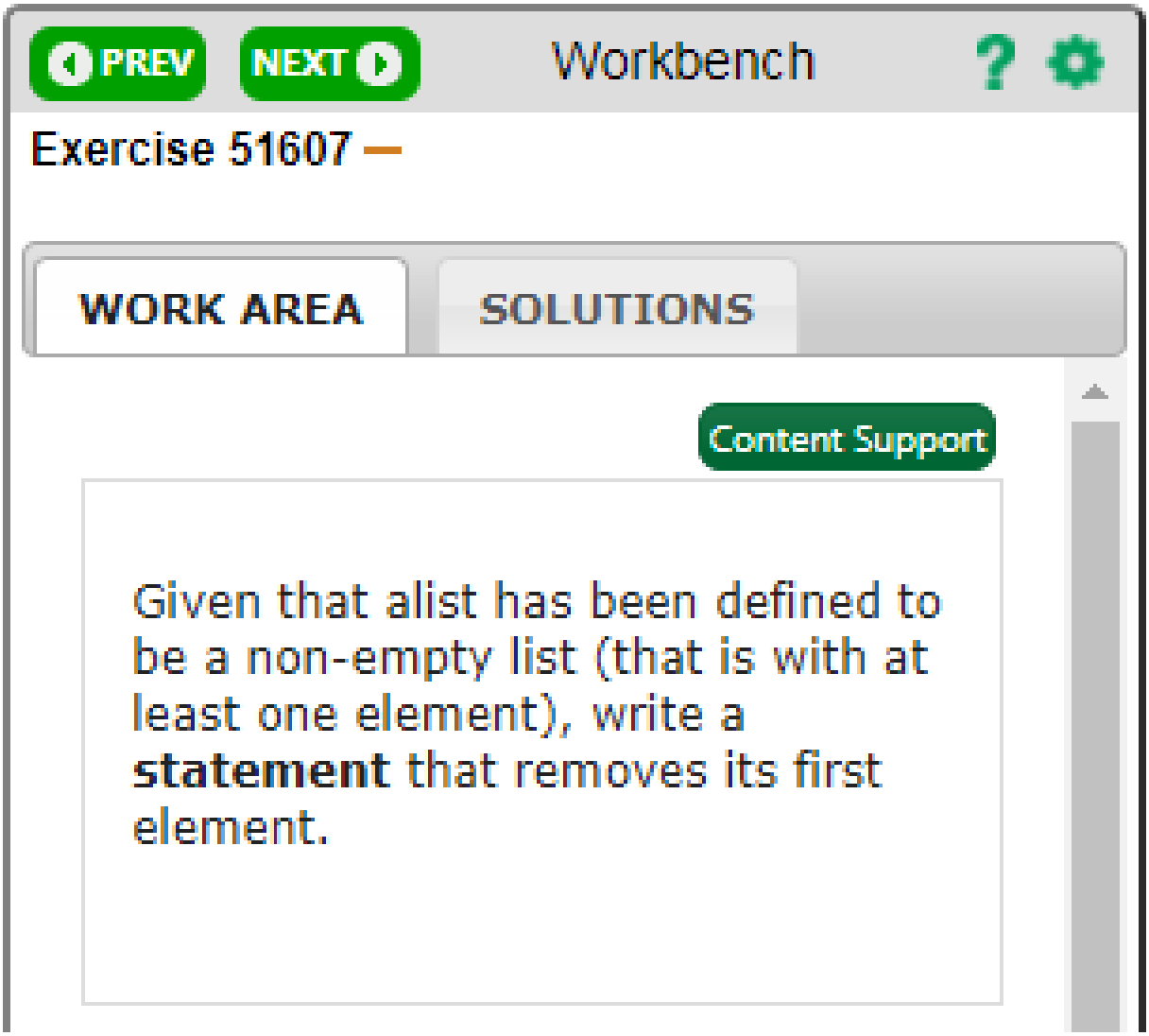
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

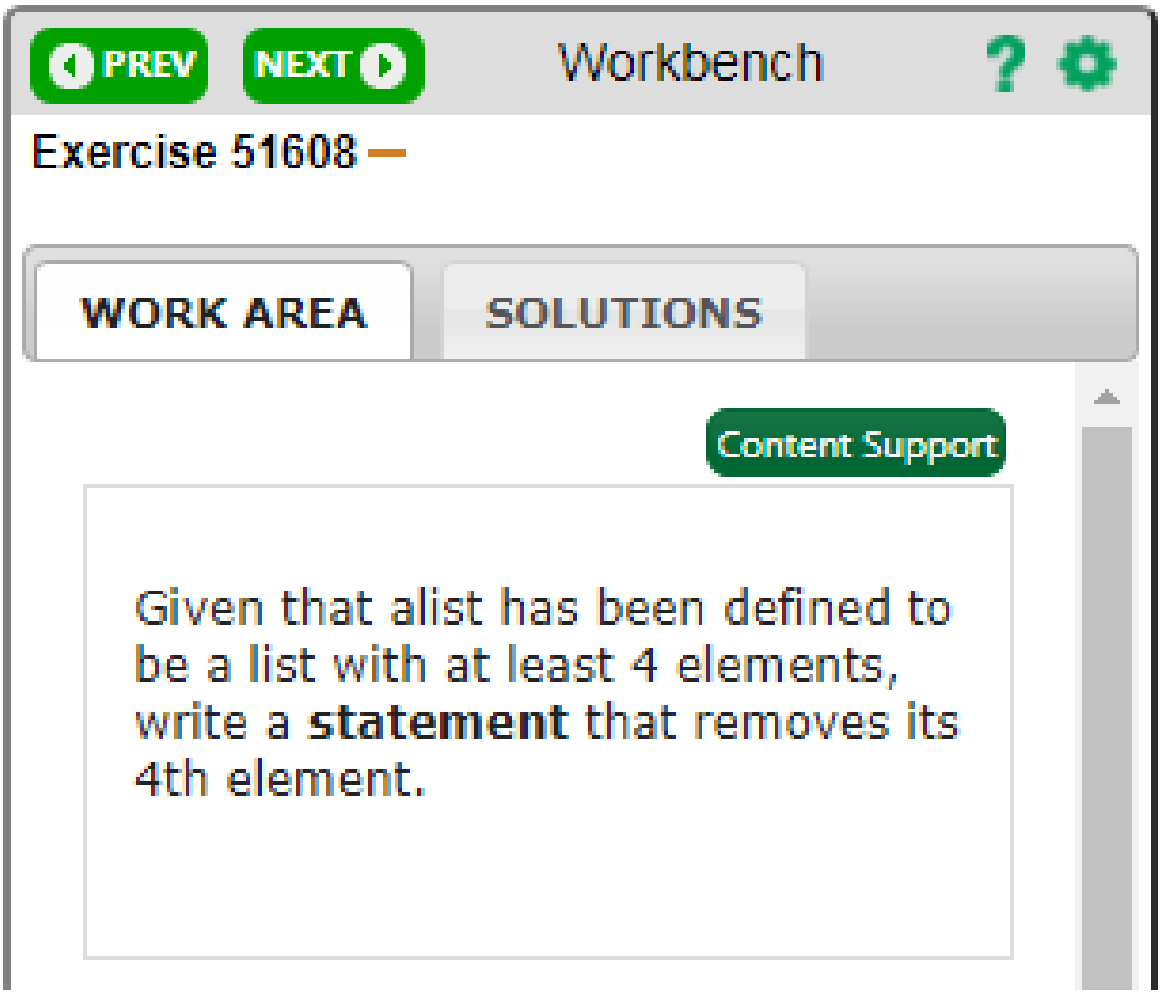
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

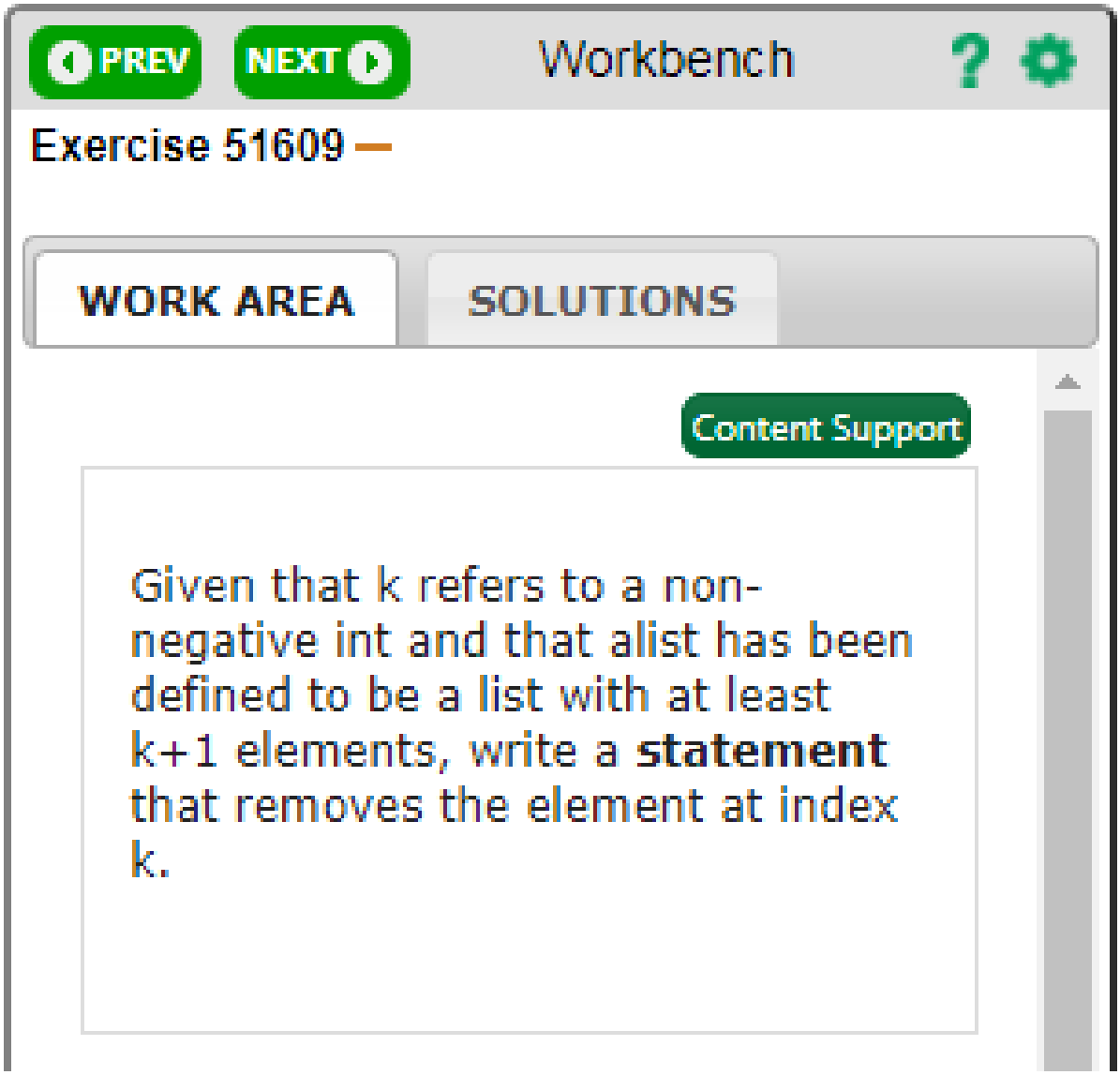
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

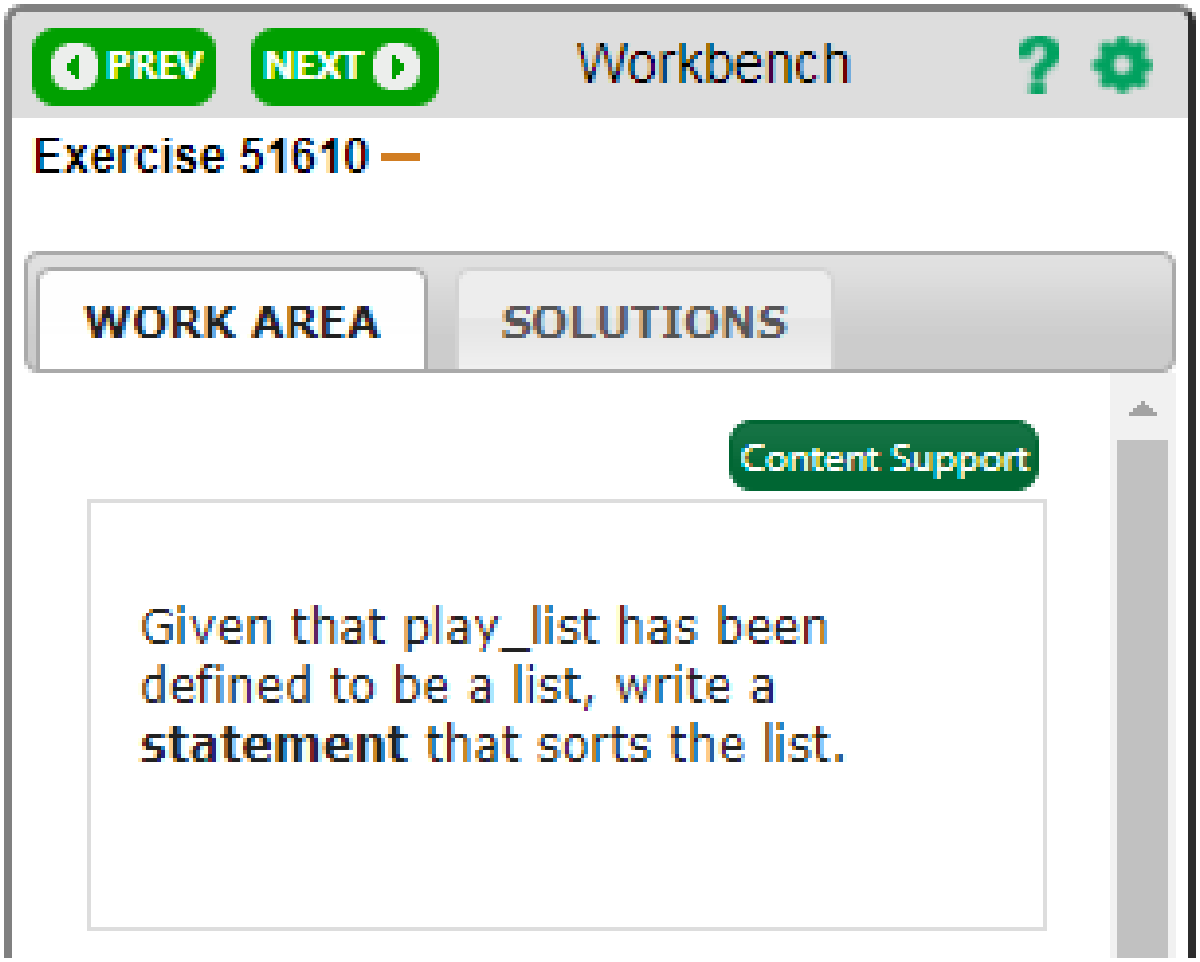
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

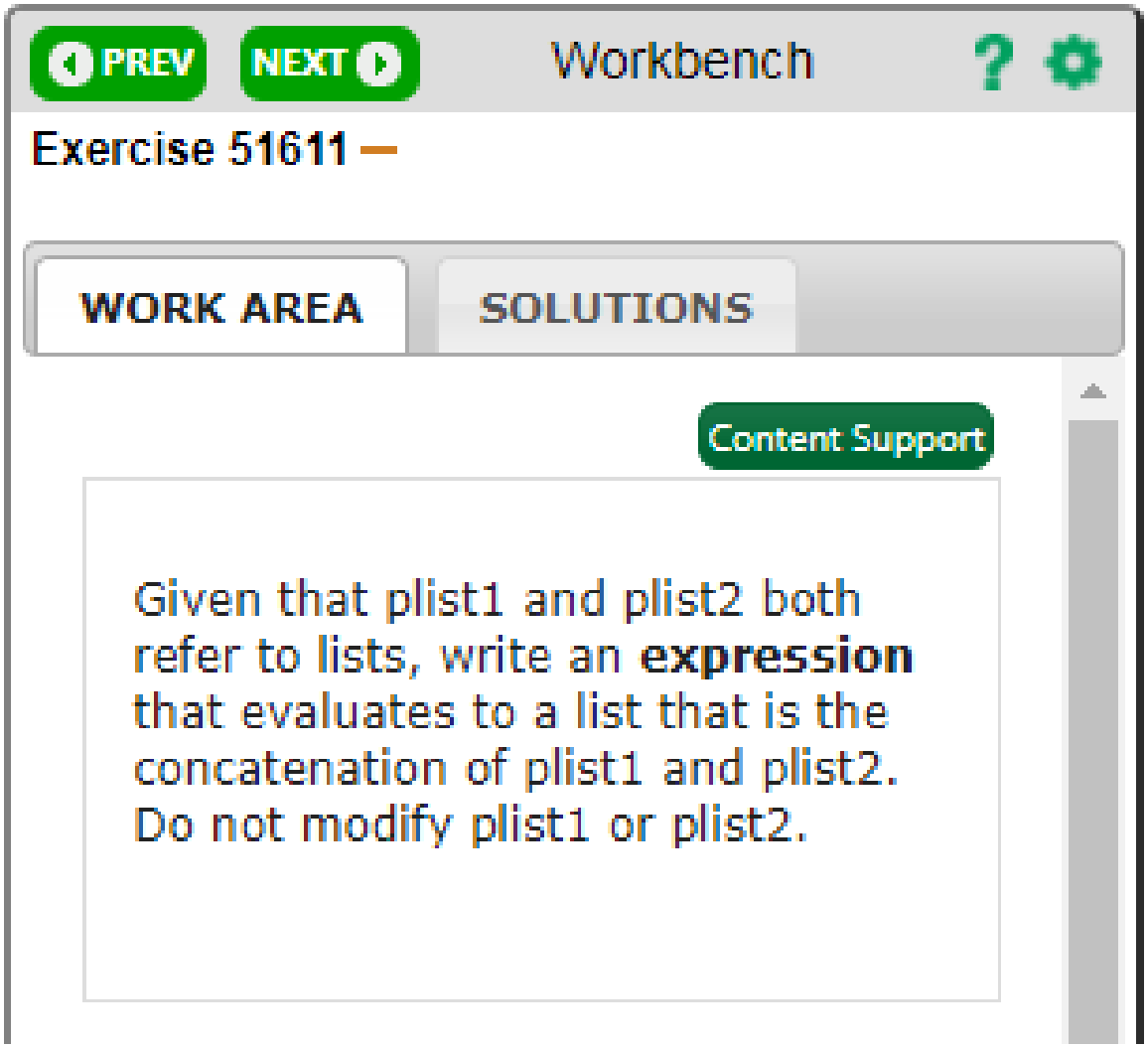
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

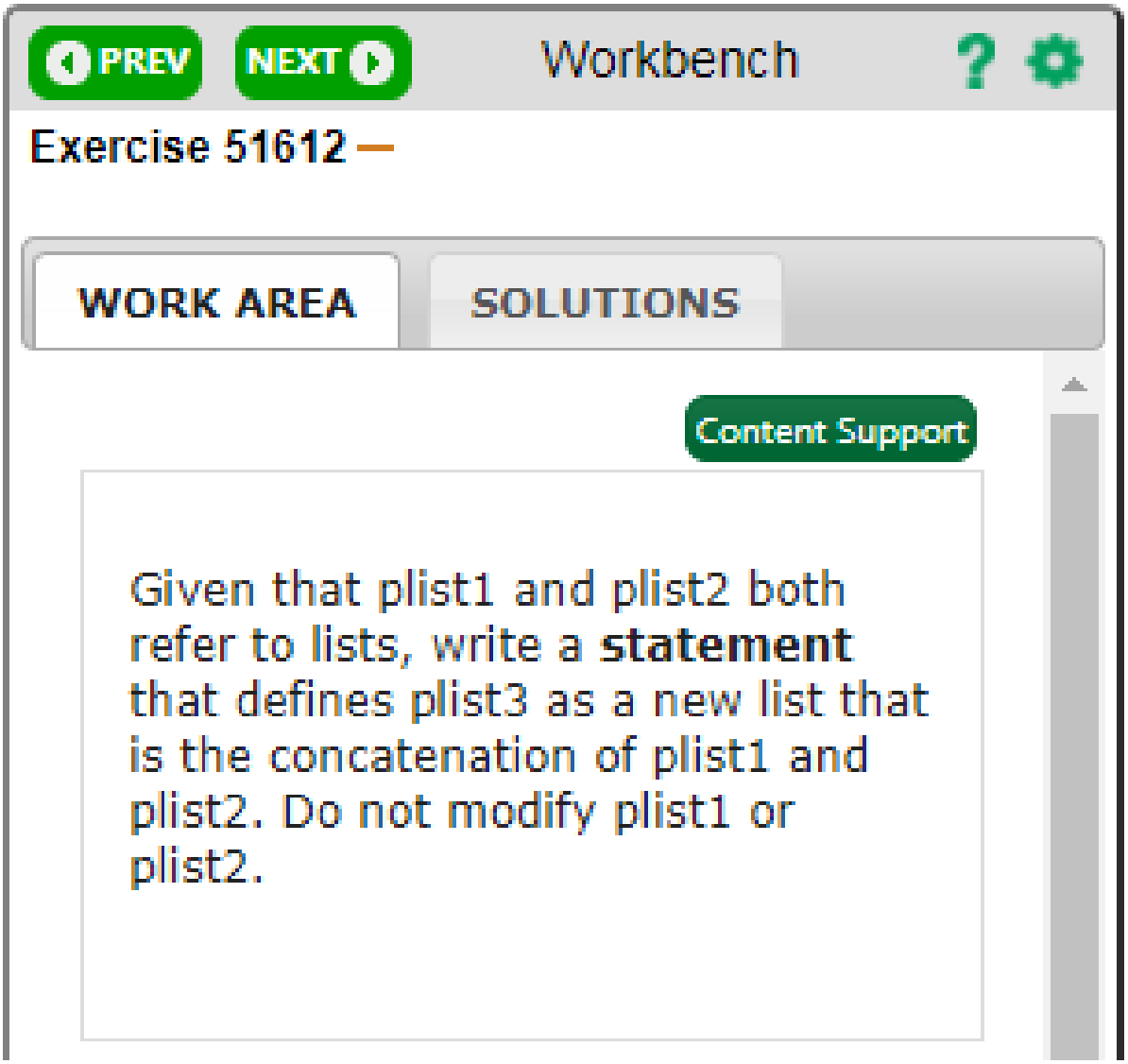
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

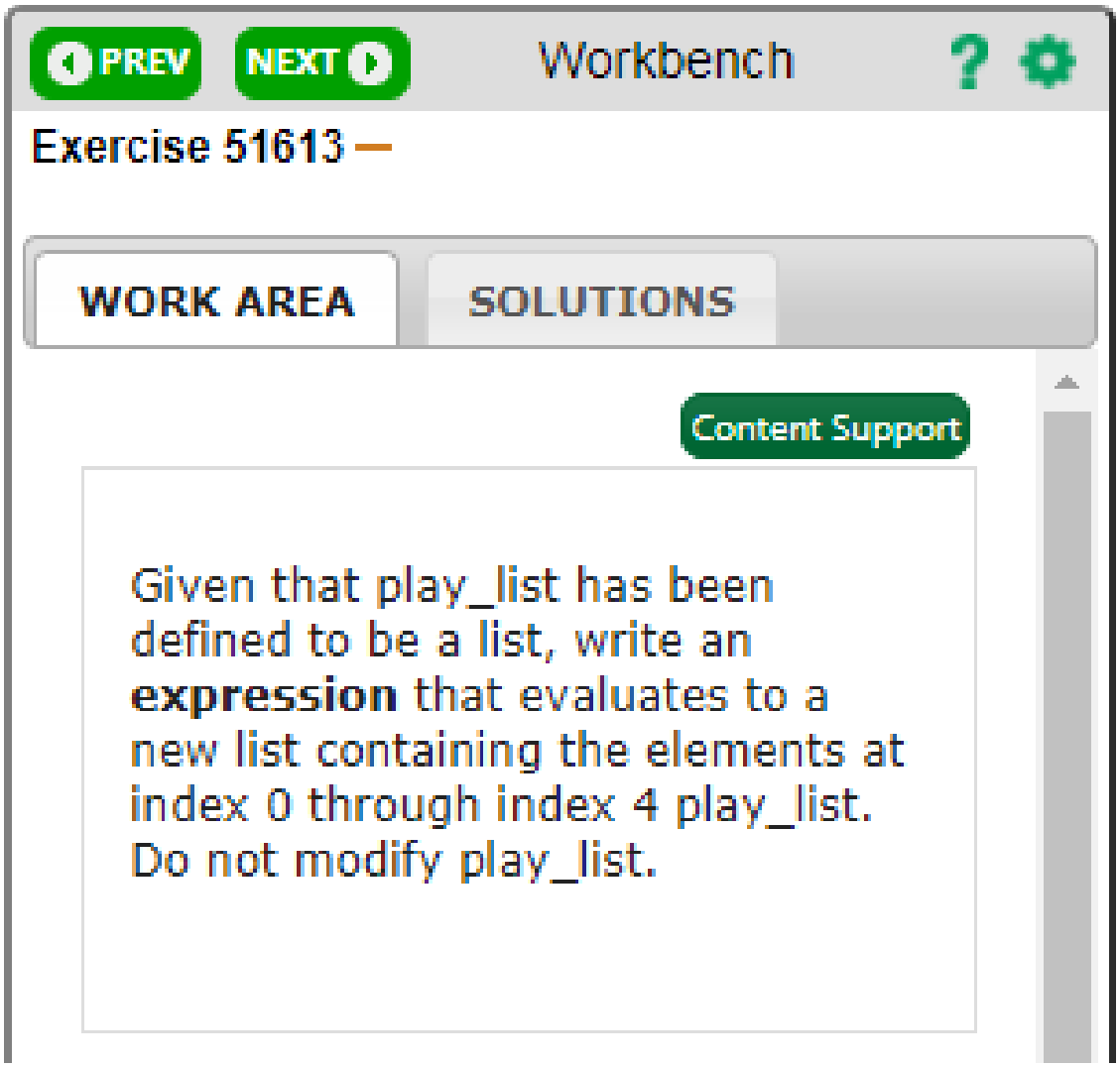
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

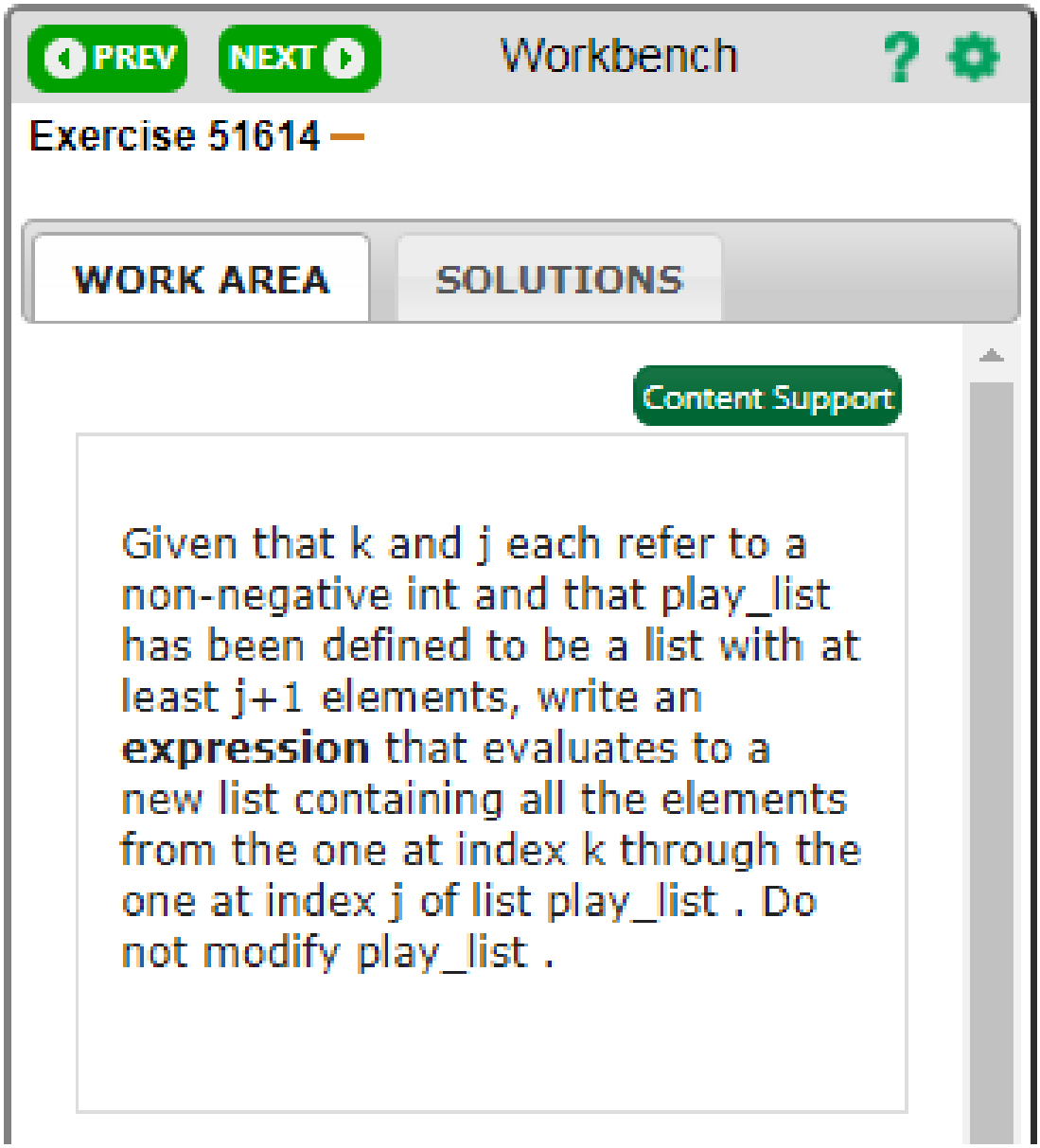
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

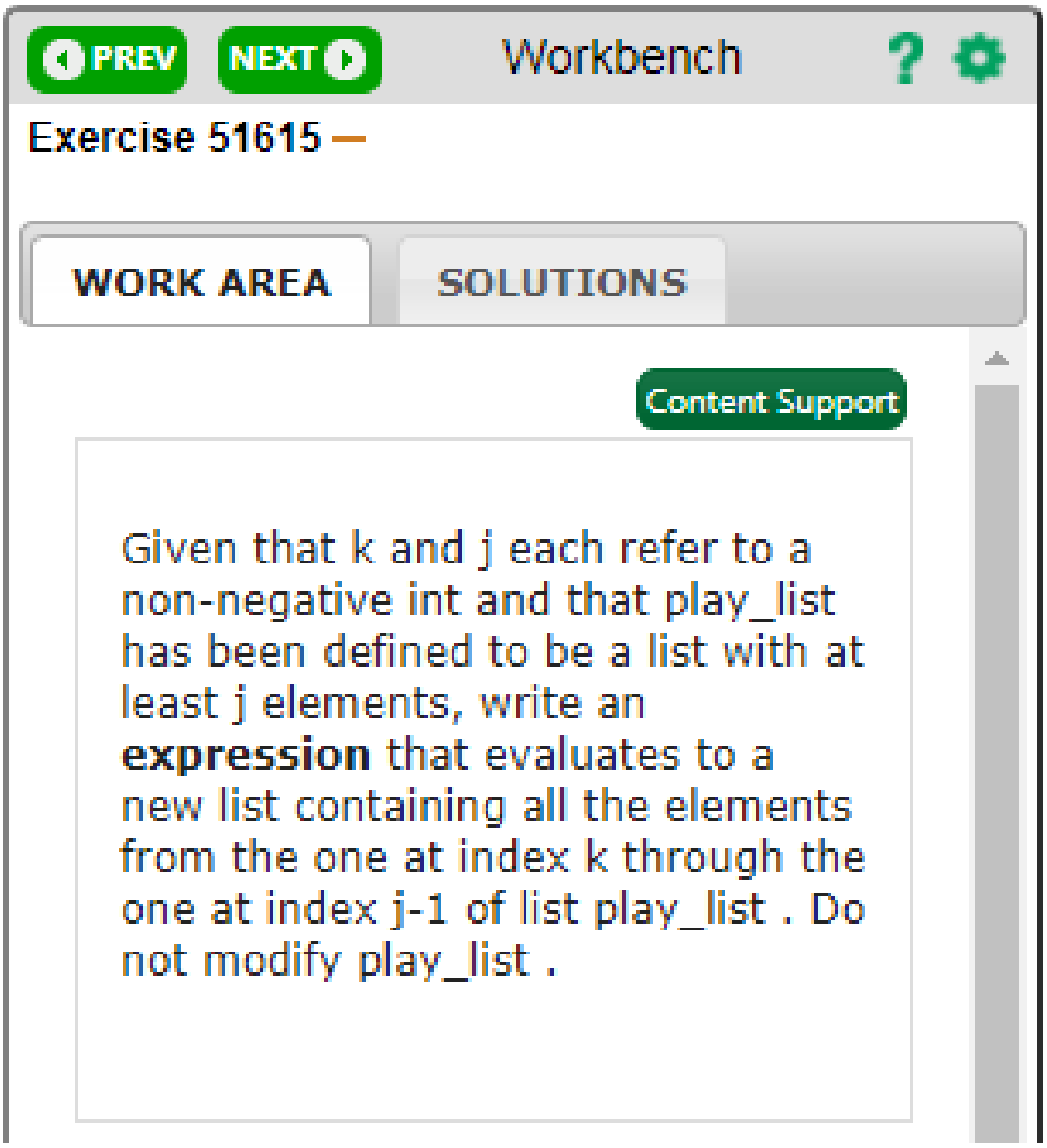
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

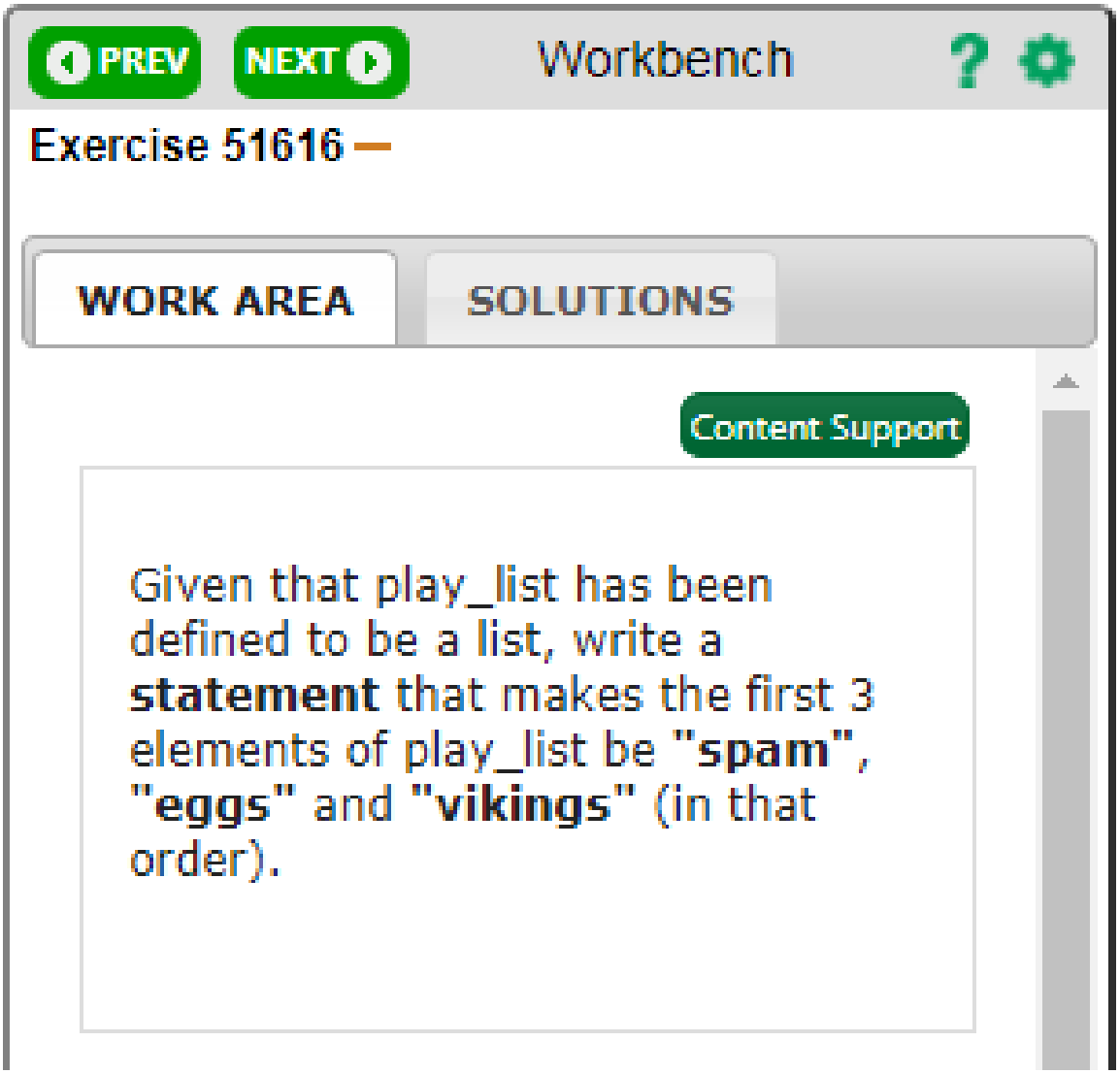
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

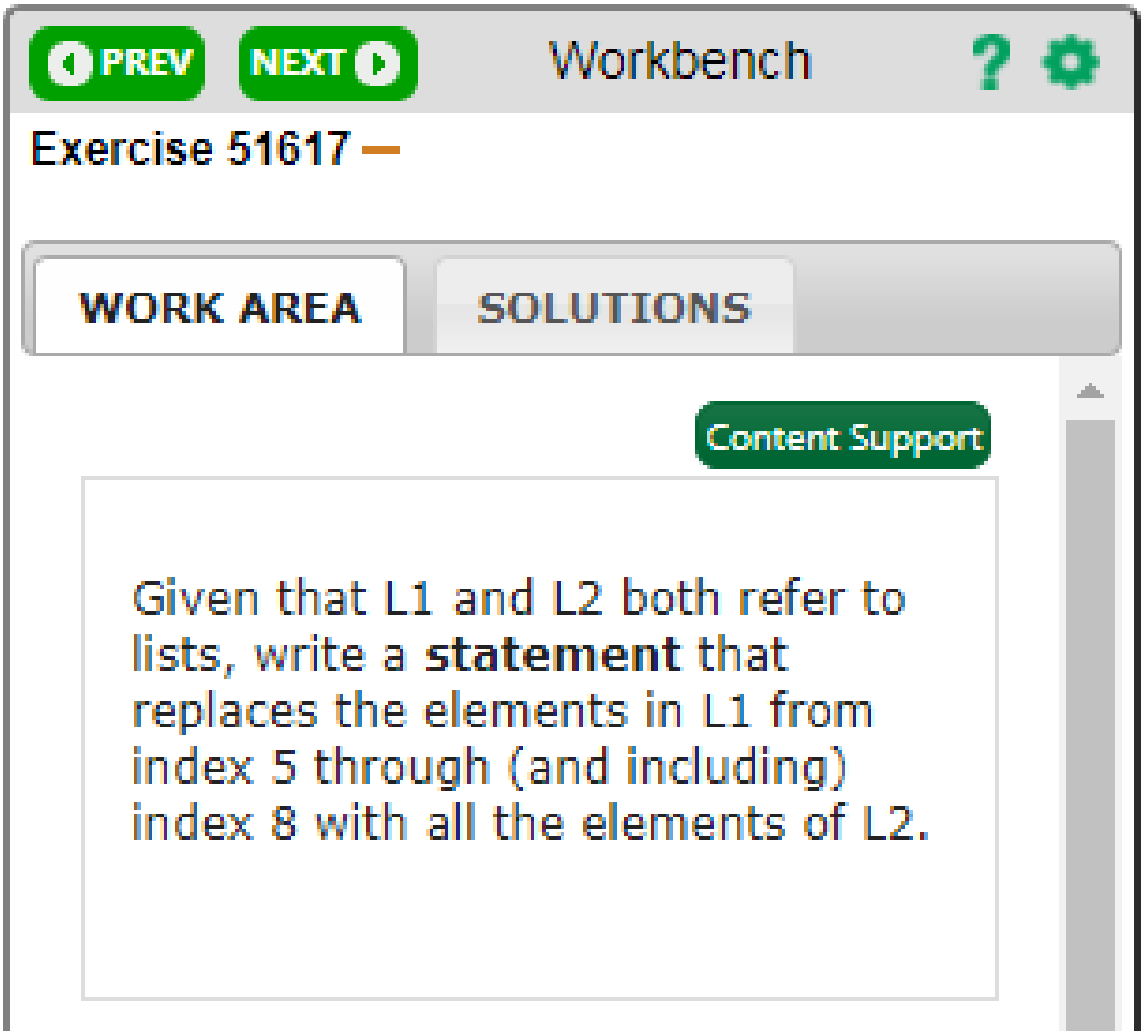
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

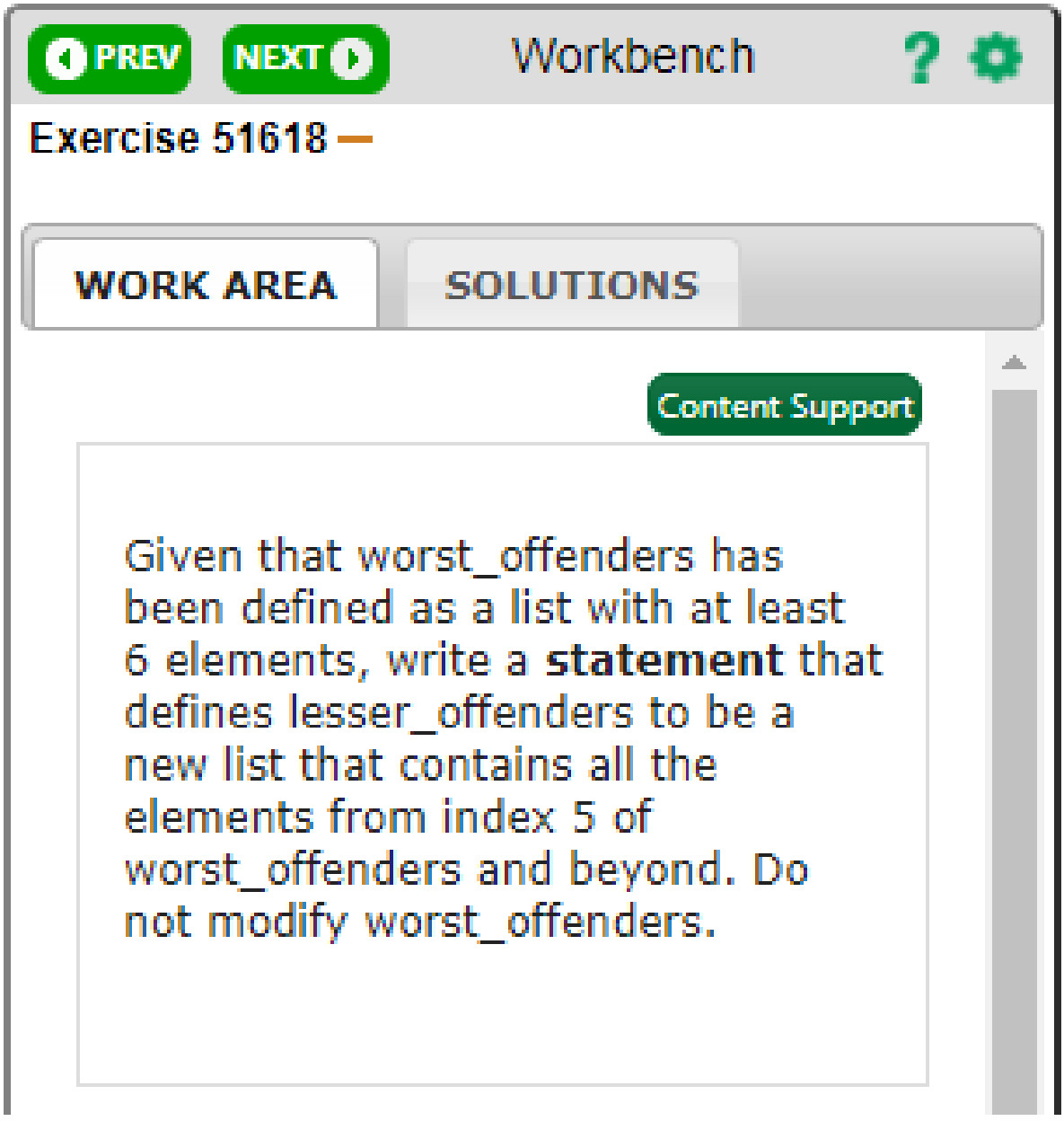
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

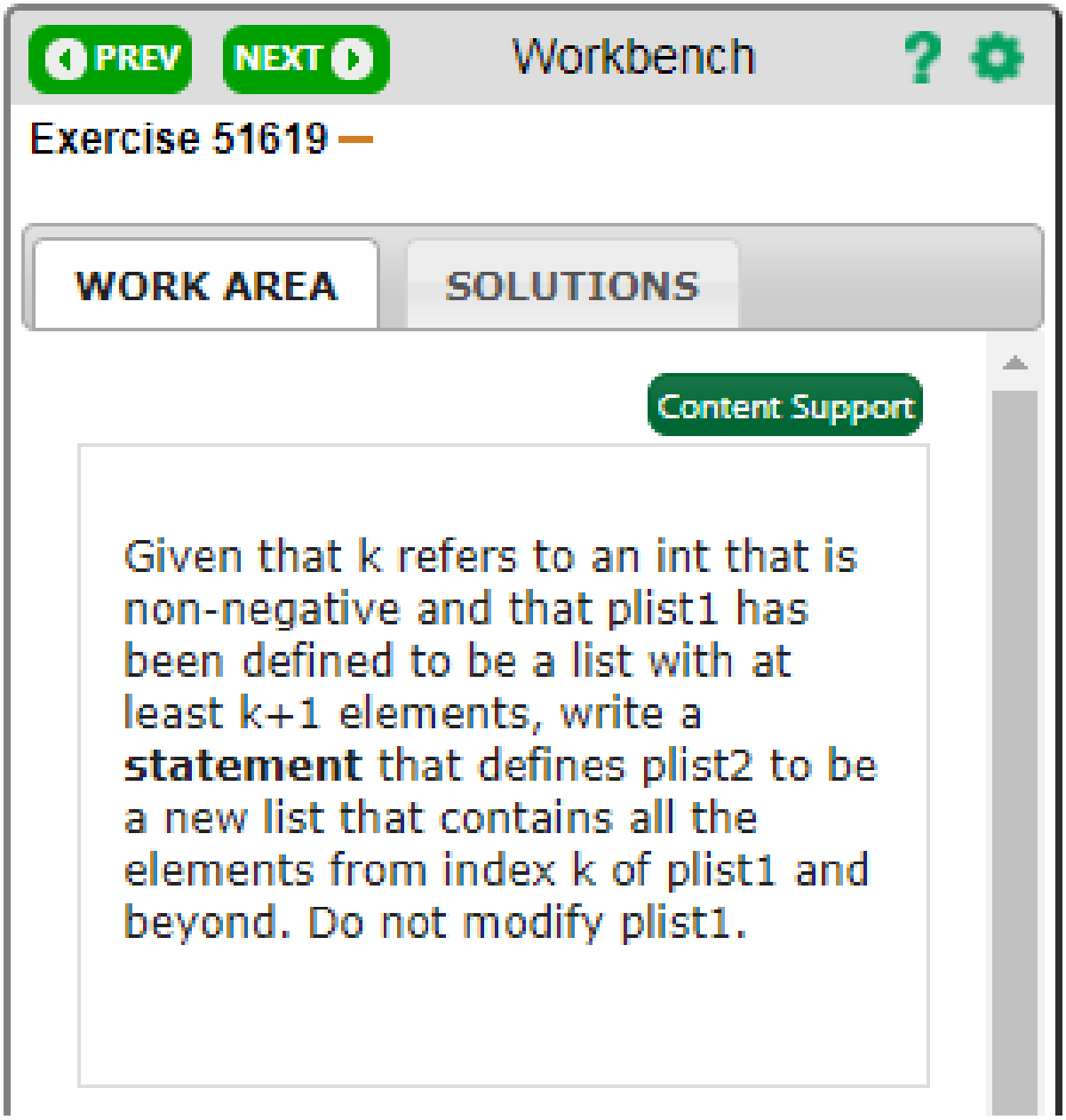
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

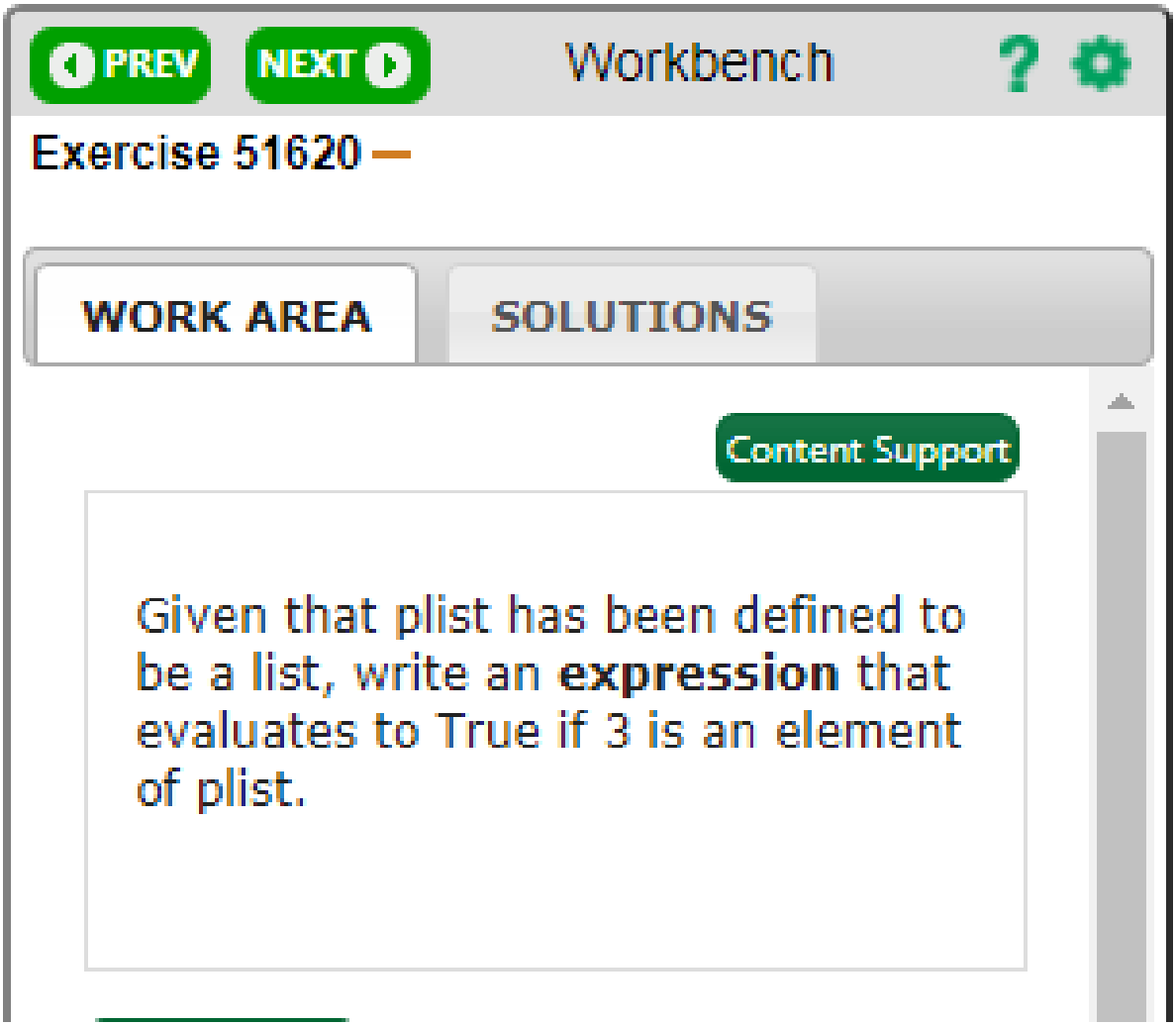
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

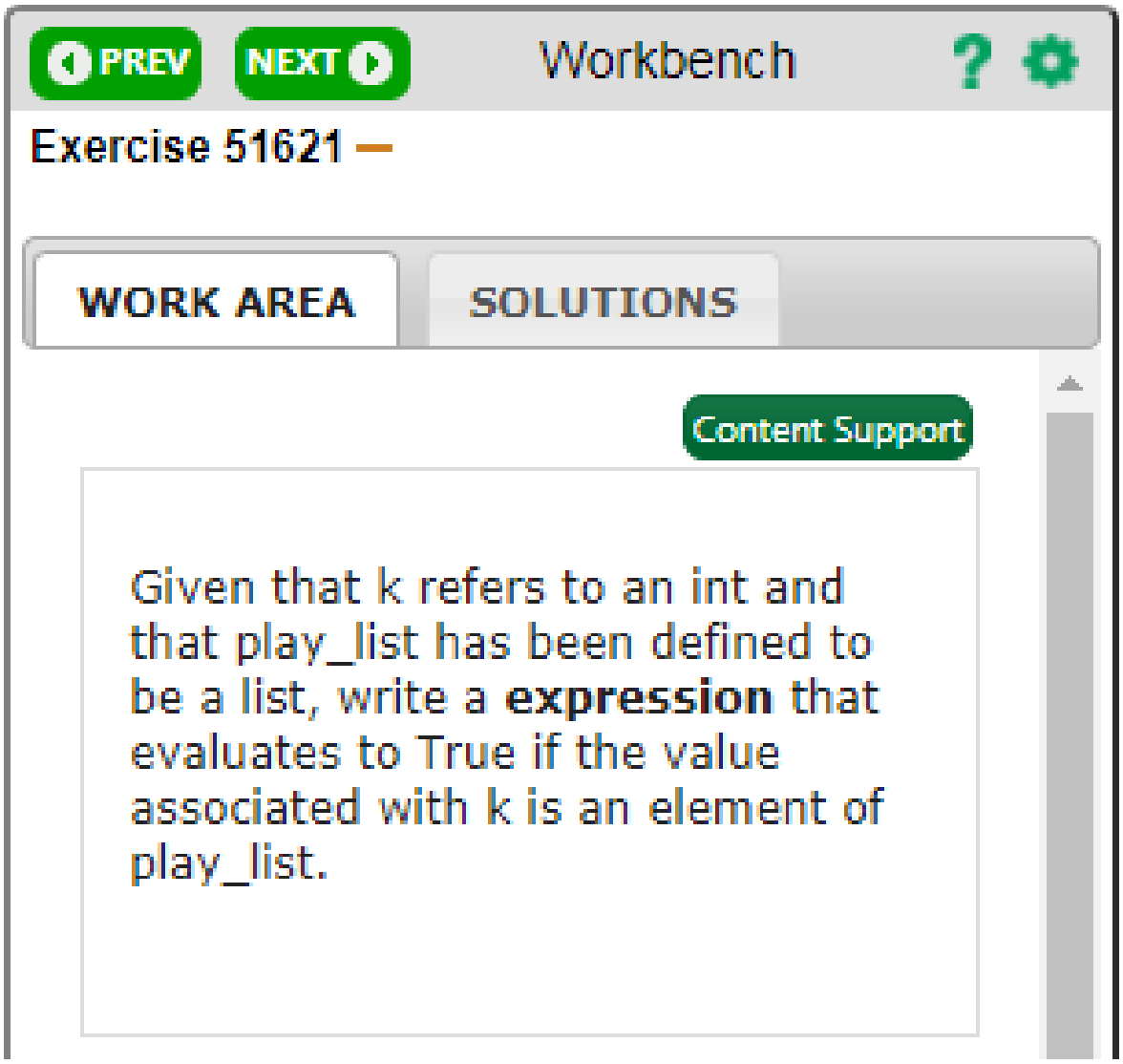
Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank

Programming Exercise



Programming Exercise Worksheet

this page is intentionally left blank