

# Lists: Indexing and Slicing

# List as an Object

	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]  
>>> x.index('r')  
>>> 5
```

- list is an object
- what are list attributes?

variable name	delimiter	attribute	arguments
x	.	index	('r')

# Indexing for List Slicing

- each element is identified by its position
- slice – elements between two positions

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

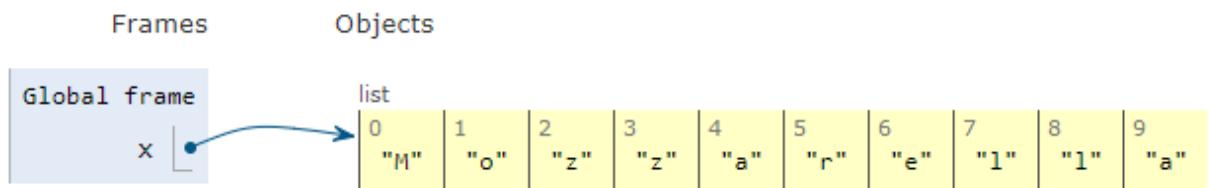
	0	1	2	3	4	5	6	7	8	9	
[	'M',	'o',	'z',	'z',	'a',	'r',	'e',	'l',	'l',	'a'	]
	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	

- indexing starts at 0, ends at (length – 1)
- can refer to slices but positive or negative indices

# List Slicing

	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]
```



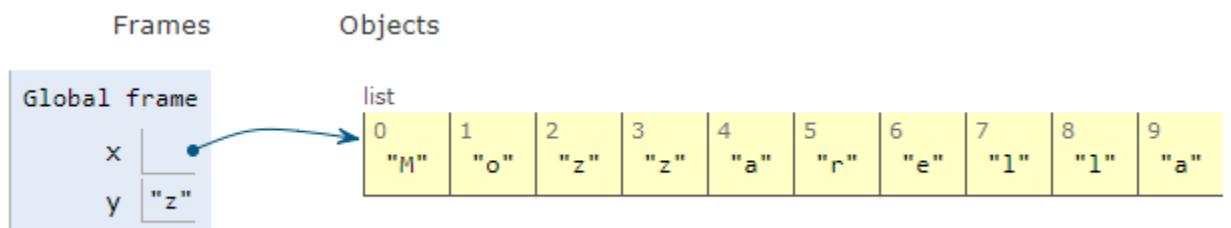
- slice operator:

variable	[	start	:	one past end	]
x	i	:	j		

# Slicing with Single 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

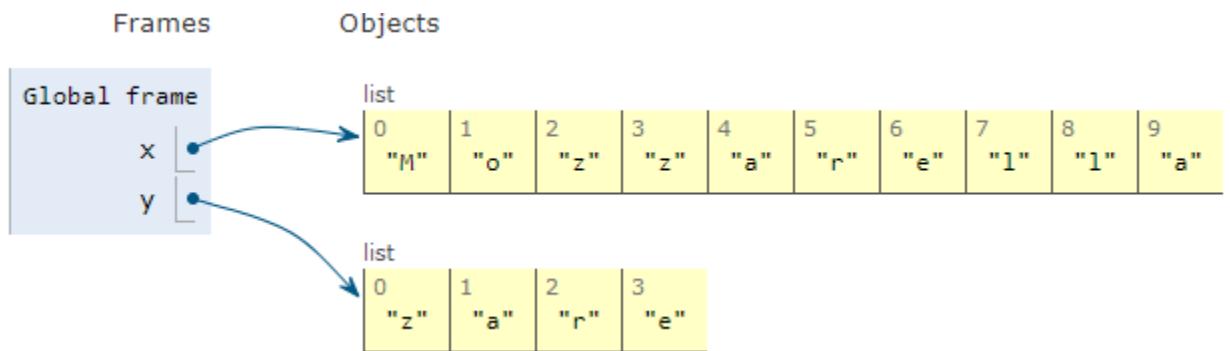
```
>>> x = ['M','o','z','z','a','r','e','l','l','a']
>>> y = x[2]
>>> y
>>> z
```



# Slicing with Positive Indices

	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	

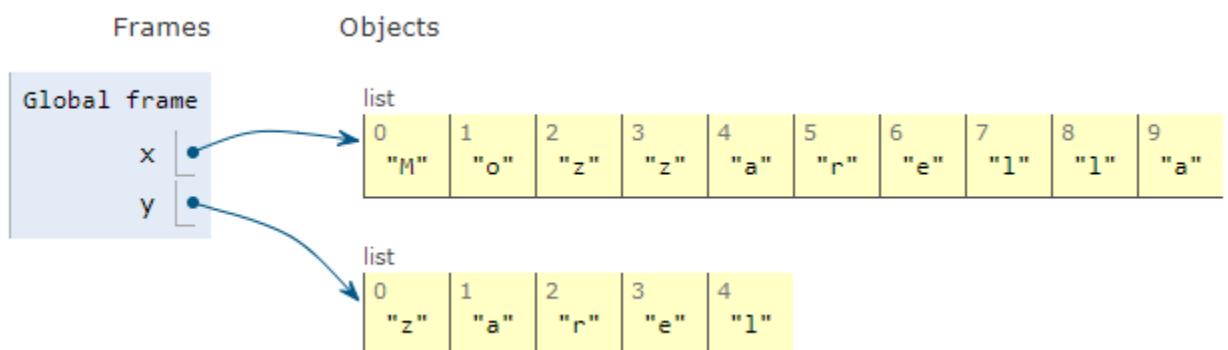
```
>>> x = ['M','o','z','z','a','r','e','l','l',a]  
>>> y = x[3 : 7]  
>>> y  
>>> [ 'z', 'a', 'r', 'e' ]
```



# Slicing w. Negative Indices

	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	

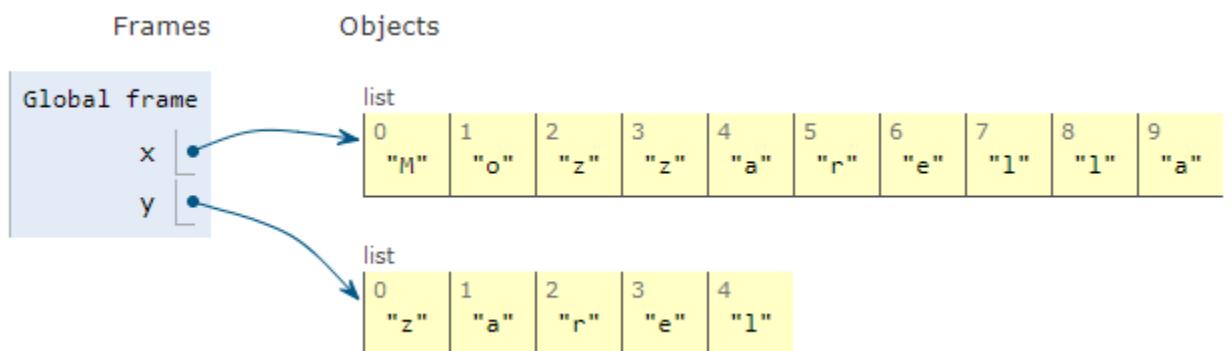
```
>>> x = ['M','o','z','z','a','r','e','l','l',a]
>>> y = x[-7:-2]
>>> y
>>> [ 'z', 'a', 'r', 'e', 'l' ]
```



# Slicing with Positive and Negative Indices

	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		

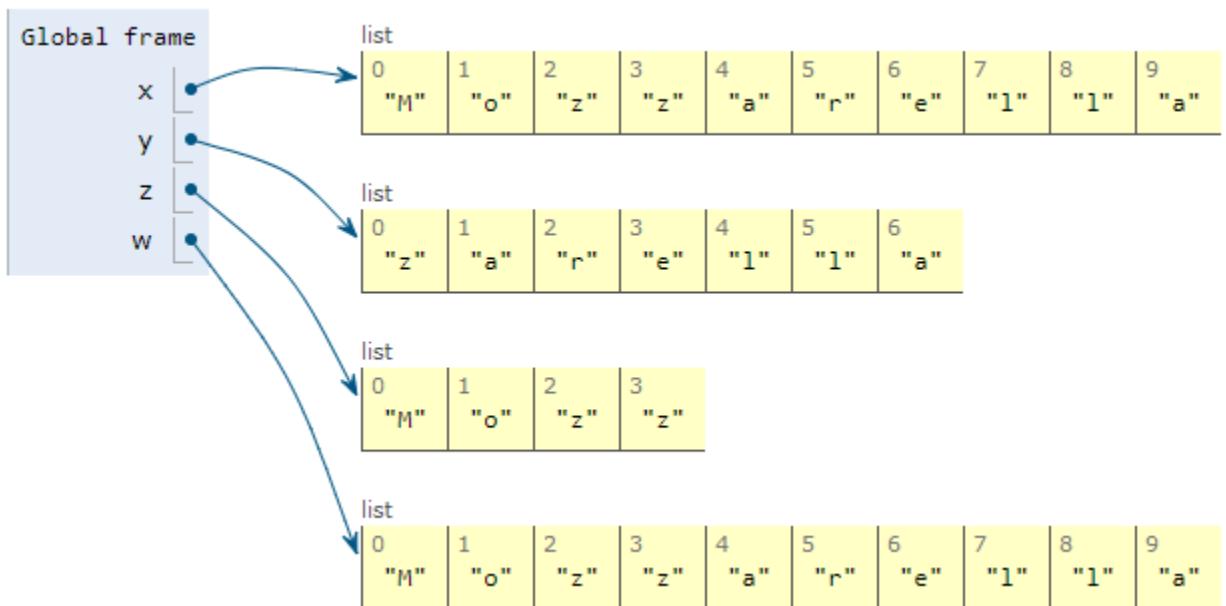
```
>>> x = ['M','o','z','z','a','r','e','l','l',a]  
>>> y = x[3 : -2]  
>>> y  
>>> ['z', 'a', 'r', 'e', 'l']
```



# List Slicing w/o Start/End

	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[3 : ]
>>> z = x[ : 4]
>>> w = x[ : ]
```



# Valid vs. Invalid Slicing

	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		

- why is this not legal?

The screenshot shows a Jupyter Notebook interface with two tabs: 'IP: Kernel 1' and 'Python 2'. The Python 2 tab contains the following code and error message:

```
>>>
>>> word = "Mozarella"
>>> word[2:5] = "xyz"
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' object does not support item assignment
>>>
```

The error message is 'TypeError: 'str' object does not support item assignment'. Below the code, status information is displayed: Encoding: ASCII, Line: 2, Column: 17, Memory: 48 %.

- why is legal!

The screenshot shows a Jupyter Notebook interface with two tabs: 'IP: Kernel 1' and 'Python 2'. The Python 2 tab contains the following code and output:

```
>>>
>>>
>>> word = "Mozarella"
>>> word = word.replace("zar", "xyz")
>>> print(word)
Moxyzella
>>>
```

The output is 'Moxyzella'. Below the code, status information is displayed: Encoding: ASCII, Line: 2, Column: 17, Memory: 48 %.

# List Slicing: Quiz

	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	

- what is the result of these operations?

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

```
>>> y = x[-1]
```

```
>>> z = x[2 : 6]
```

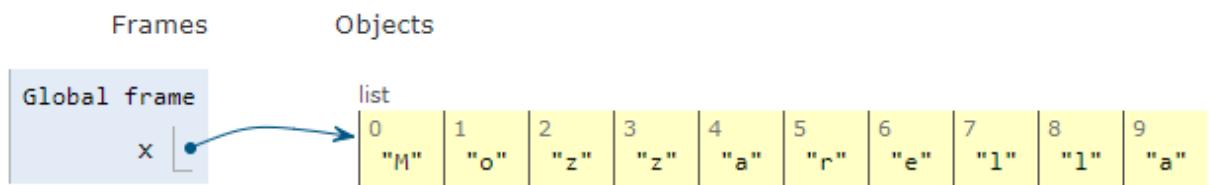
```
>>> w = x[-7 : -3]
```

```
>>> v = x[ : ]
```

# List Slicing: Quiz (cont'd)

	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	

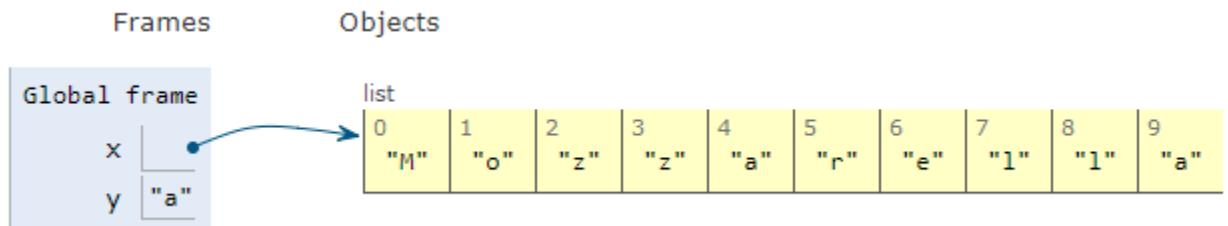
```
>>> x = ['M','o','z','z','a','r','e','l','l',a]
```



# List Slicing: Quiz (cont'd)

	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

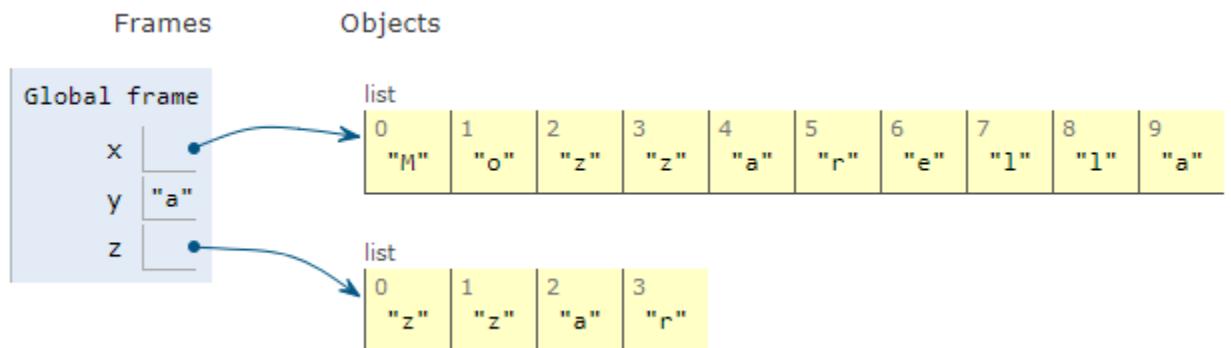
```
>>> x = ['M','o','z','z','a','r','e','T','l','a']
>>> y = x[-1]
>>> a
```



# List Slicing: Quiz (cont'd)

	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[-1]  
>>> z = x[2 : 6]
```



# List Slicing: Quiz (cont'd)

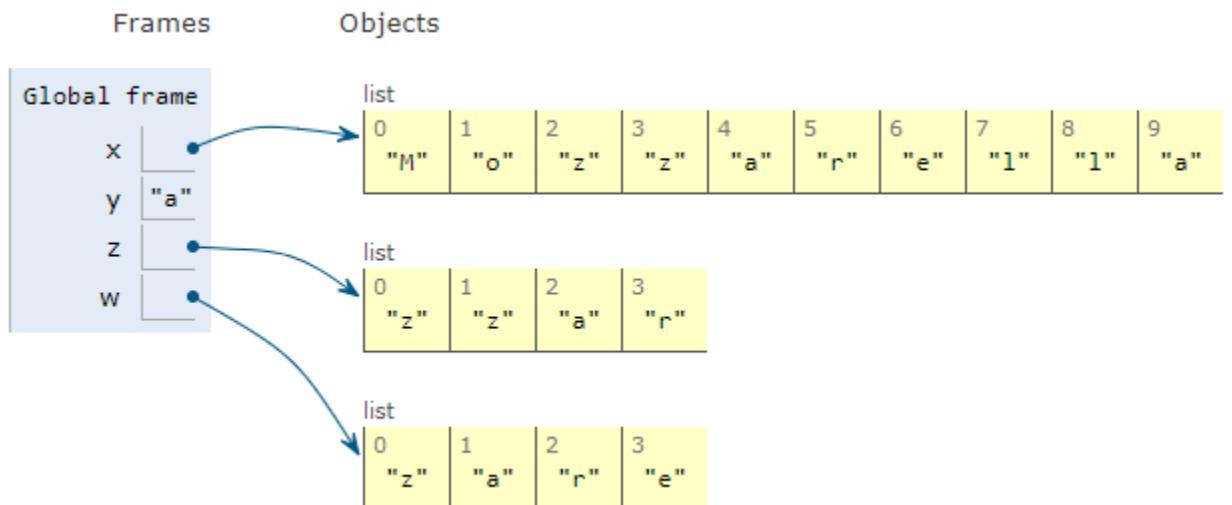
	0	1	2	3	4	5	6	7	8	9	
[	'M'	'O'	'Z'	'Z'	'a'	'r'	'e'	'l'	'l'	'a'	]
	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	

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

```
>>> y = x[-1]
```

```
>>> z = x[2 : 6]
```

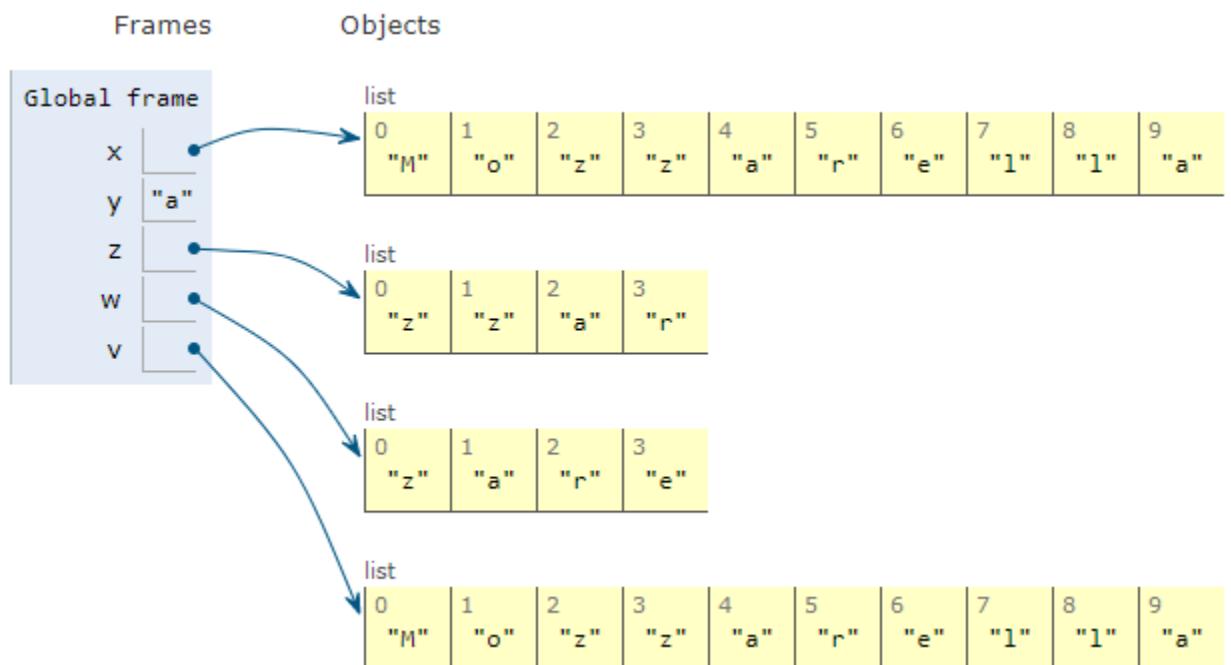
```
>>> w = x[-7 : -3]
```



# List Slicing: Quiz (cont'd)

	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

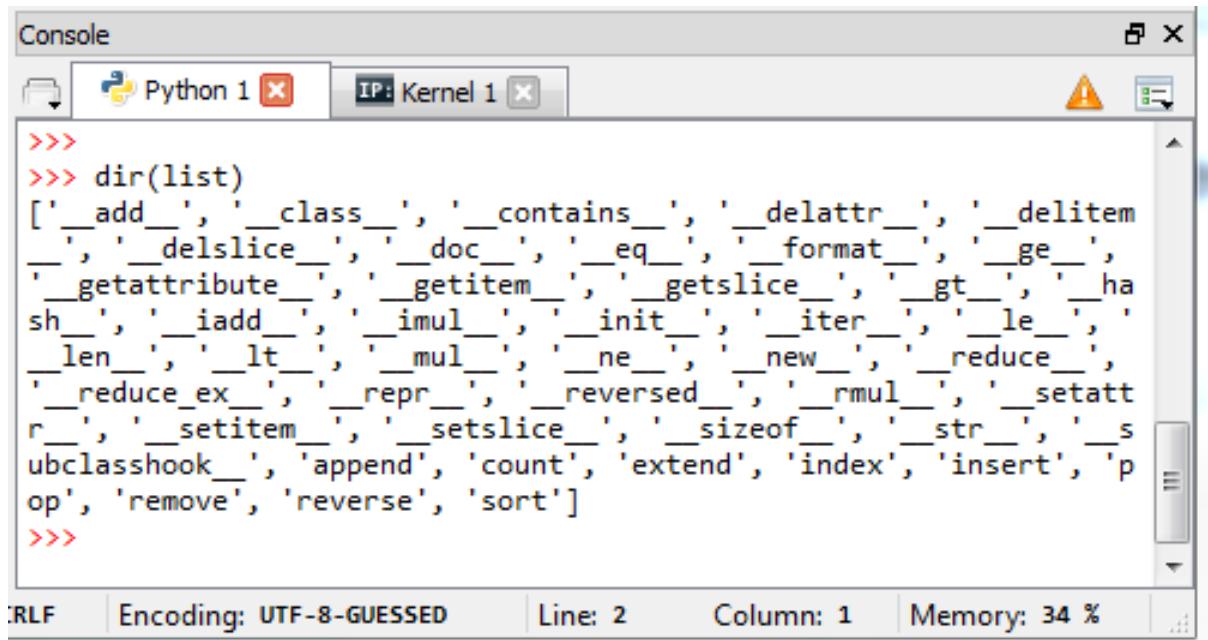
```
>>> x = ['M', 'o', 'z', 'z', 'a', 'r', 'e', 'T', 'l', 'a']
>>> y = x[-1]
>>> z = x[2 : 6]
>>> w = x[-7 : -3]
>>> v = x[ : ]
```



- example of a “shallow” copy

# List Attributes

- use `dir(list)` command



The screenshot shows a Jupyter Notebook interface with a "Console" tab active. It displays the output of the `dir(list)` command. The output is a list of method names starting with underscores, such as `__add__`, `__contains__`, `__delattr__`, etc. The console also shows the standard Python 2 prompt (`>>>`) and the standard IPython kernel icon.

```
>>>
>>> dir(list)
['__add__', '__class__', '__contains__', '__delattr__', '__delitem__',
 '__delslice__', '__doc__', '__eq__', '__format__', '__ge__',
 '__getattribute__', '__getitem__', '__getslice__', '__gt__', '__ha__',
 '__iadd__', '__imul__', '__init__', '__iter__', '__le__',
 '__len__', '__lt__', '__mul__', '__ne__', '__new__', '__reduce__',
 '__reduce_ex__', '__repr__', '__reversed__', '__rmul__', '__setatt__r__',
 '__setitem__', '__setslice__', '__sizeof__', '__str__', '__ubclasshook__',
 'append', 'count', 'extend', 'index', 'insert', 'pop',
 'remove', 'reverse', 'sort']
>>>
```

Below the code, the status bar indicates: RLF | Encoding: UTF-8-GUESSED | Line: 2 | Column: 1 | Memory: 34 %

- names starting with `__` refer to “local” class names/methods

# Review Problems

# Interview Problem

- write a *for* loop to print elements in list and their positions

# Interview Problem

- convert a list into other data types

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' in green rounded rectangles, followed by a 'Workbench' button with a gear icon, and a help icon with a question mark and settings gear. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A 'Content Support' button is located at the bottom right of the main area. The main content area contains the following text:

Write **statement** that defines **plist** to be a **list** of the following ten elements: **10, 20, 30, ..., 100** in that order.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there is a toolbar with green buttons for 'PREV' and 'NEXT', a 'Workbench' button, and a help/symbol button. Below the toolbar, the text 'Exercise 51193 —' is displayed. Underneath this, there are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located above a large text box. The text box contains the following instruction:

Create a **list** named `tax_rates`, consisting of the following five elements: `0.10, 0.15, 0.21, 0.28, 0.31`, in that order.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and a gear icon on the right. Below these is the title 'Exercise 51194 —'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'WORK AREA' being the active tab. A green button labeled 'Content Support' is located above the main text area. The main text area contains the following instruction:

Write a **statement** that defines the variable **denominations**, and associates it with a **list** consisting of the following six **elements**: 1, 5, 10, 25, 50, 100, in that order.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon ('?') and settings icon ('⚙️') on the right. Below these is the title 'Exercise 51195'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'WORK AREA' being the active tab. A green button labeled 'Content Support' is located above the main content area. The main content area contains the following text:

Given a variable plist, that refers to a non-empty list, write an **expression** that refers to the **first** element of the list.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' in green rounded rectangles, followed by a 'Workbench' button in blue and orange, and a help icon ('?') and settings icon ('gear') in green. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'SOLUTIONS' currently selected. A 'Content Support' button is located in the top right corner of the main content area. The main content area contains the following text:

Given a variable `plist`, that refers to a list with 34 elements, write an **expression** that refers to the **last element** of the list.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and a gear icon on the right. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located near the bottom of the tabs. The main content area contains the following text:

Assume that the variable `plist` has been defined and refers to a list. Write a **statement** that assigns the **next to last** element of the list to `x`.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and gear icon on the right. Below these is the title 'Exercise 51199'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'WORK AREA' being the active tab. A green button labeled 'Content Support' is located below the tabs. The main content area contains the following text:

Given that a variable named **plist** has been defined and refers to a non-empty list, write a **statement** that associates its **first** element with 3.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' in green rounded rectangles, followed by a 'Workbench' button in blue and red, and a help/gear icon. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A 'Content Support' button is located in the top right of the main area. The main content area contains the following text:

Assume that `salary_steps` refers to a non-empty list, write a **statement** that assigns the value 30000 to the **first** element of this **list**.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top is a toolbar with 'PREV' and 'NEXT' buttons, a 'Workbench' tab, and a help icon. Below the toolbar is the title 'Exercise 51201 —'. There are two tabs: 'WORK AREA' (selected) and 'SOLUTIONS'. A 'Content Support' button is located above the main content area. The main content area contains the following text:

Assume that a list of integers named salary\_steps that contains exactly five elements has been defined.

Write a statement that changes the value of the last element in the list to 160000.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and a gear icon on the right. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located above the main content area. The main content area contains the following text:

Assume that `plist` has been defined and is associated with a non-empty list. Write a **statement** that increases the **first** element of this list by 10.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and a gear icon on the right. Below these are two tabs: 'WORK AREA' (highlighted in blue) and 'SOLUTIONS'. A 'Content Support' button is located above a large text area. The main text area contains the following instruction:

Given that plist has been defined to be a list of 30 elements, add 5 to its last element.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, a 'Workbench' button in the center, and a help/gear icon on the right. Below these, the title 'Exercise 51204 —' is displayed. Underneath the title, there are two tabs: 'WORK AREA' (highlighted in blue) and 'SOLUTIONS'. A 'Content Support' button is located above a large text area. The main text area contains the following problem statement:

Assume that a variable named `plist` refers to a list with 12 elements, each of which is an int. Assume that the variable `k` refers to a value between 0 and 6. Write a statement that assigns 15 to the list element whose index is `k`.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and gear icon on the right. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located near the bottom of the tabs. The main content area contains the following text:

Given that `plist` refers to a non-empty list , write a **statement** that assigns the int  
-1  
to the **last element of the list**.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT'. To the right of these are links for 'Workbench', '?', and a gear icon. Below the navigation is the title 'Exercise 51206 —'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'WORK AREA' being the active tab. A green button labeled 'Content Support' is located below the tabs. The main content area contains the following text:

Assume that `plist` is associated with a list that has 12 elements. Assume further that `k` refers to an int between 2 and 8. Assign 9 to the element **just after** the element in `plist` whose **index** is `k`.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' in green rounded rectangles, followed by a 'Workbench' button in blue and orange, and a help/gear icon. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A 'Content Support' button is located above a scrollable text area. The text area contains the following question:

Assume that a variable named **plist** has been defined and is associated with a **list** that consists of 12 elements. Assume further that **k** refers to an **int** between 2 and 8. Assign 22 to the element **just before** the element in **plist** whose **index** is **k**.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon ('?') and settings icon ('⚙️') on the right. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is visible above the main content area. The main content area contains the following text:

Assume that `plist` refers to a list containing exactly five elements. Assume, in addition, that `j` refers to an int with a value that is between 0 and 3.

Write a **statement** that associates a new value with the element of the list indexed by `j`. This new value should be equal to **twice** the value of the next element of the list (i.e. the element after the element indexed by `j`).

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' in green rounded rectangles, a 'Workbench' button in blue, and a help icon with a question mark and gear symbol. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'SOLUTIONS' currently selected. A 'Content Support' button is located in the top right of the main area. The main content area contains the following text:

Assume that `play_list` refers to a non-empty list, and that all its elements refer to values of type int. Write a statement that associates a new value with the first element of the list. The new value should be equal to twice the value of the last element of the list.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and a gear icon on the right. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located above the main content area. The main content area contains the following text:

Associate True with the variable  
has\_dups if the list list1 has any  
duplicate elements (that is if any  
element appears more than once),  
and False otherwise.

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon ('?') and settings icon ('⚙️') on the right. Below these is the title 'Exercise 51600 —'. Underneath the title are two tabs: 'WORK AREA' (which is selected) and 'SOLUTIONS'. In the bottom right corner of the main area, there is a green button labeled 'Content Support'. The main content area contains the following text: 'Write a statement that defines plist to be the empty list.'

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**

# Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon with a gear symbol on the right. Below the navigation bar, the title 'Exercise 51601 —' is displayed. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located below the tabs. The main area contains a text box with the following instruction:

Write a **statement** that defines  
plist as the list containing exactly  
these elements (in order): "spam",  
"eggs", "vikings" .

# **Programming Exercise**

## **Worksheet**

**this page is intentionally left blank**