



# PYTHON

A Highly Expressive  
Programming Language..

Computational Thinking with  
Programming

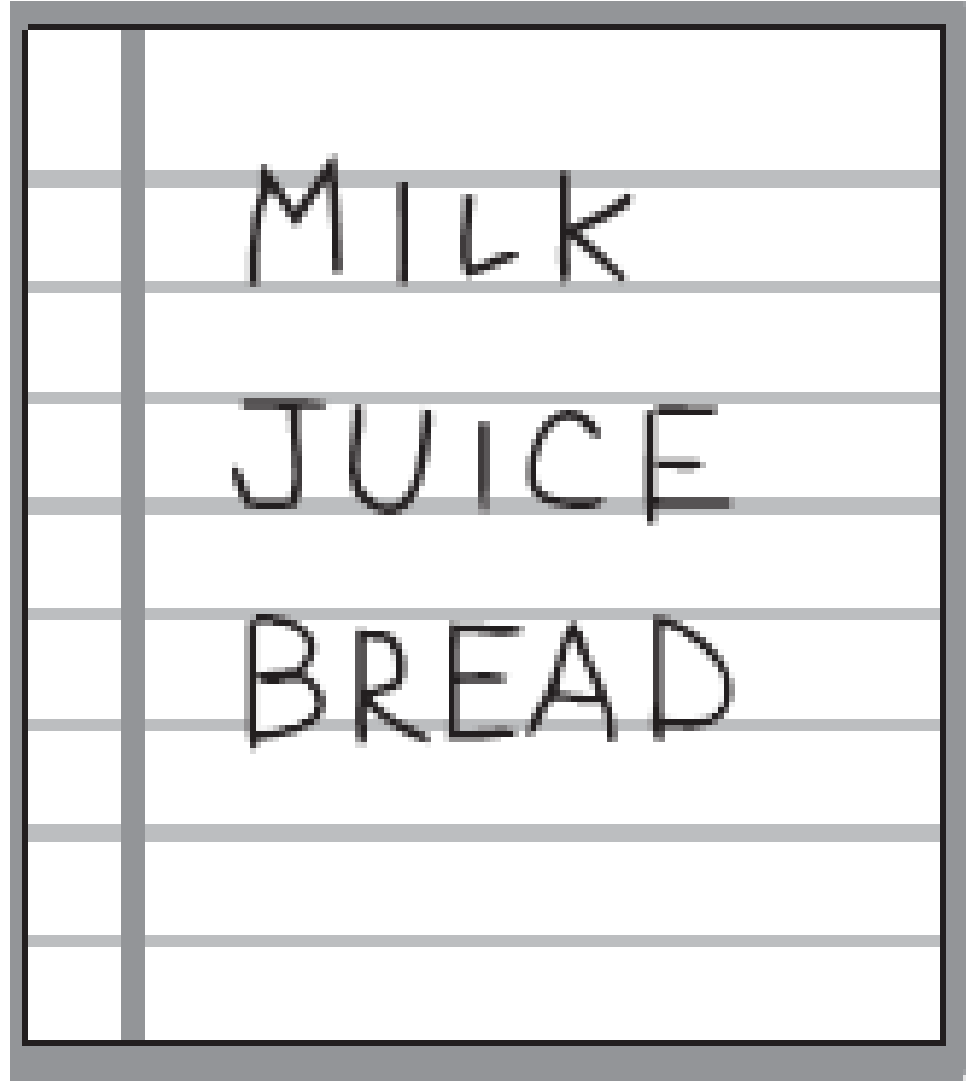
@cse\_bennett

@csebennett



# Python Collections/Sequences (Arrays)

- **List** is a collection which is ordered and changeable. Allows duplicate members.
- **Tuple** is a collection which is ordered and unchangeable. Allows duplicate members.
- **Set** is a collection which is unordered and unindexed. No duplicate members.
- **Dictionary** is a collection which is unordered, changeable and indexed. No duplicate members.



	MILK
	JUICE
	BREAD

## List Structures

---

# List Structures

- A list is a linear data structure, meaning that its elements have a linear ordering. That is, there is a first element, a second element, and so on.

List Characteristics	Elements
Element Type	All elements of the same type
	Elements of different types
Length	Fixed length
	Varying length
Modifiability	Mutable (alterable)
	Immutable (unalterable)
Common Operations	Determine if a list is empty
	Determine the length of a list
	Access (retrieve) elements of a list
	Insert elements into a list
	Replace elements of a list
	Delete elements of a list
	Append elements to (the end of) a list

0: 68.8

1: 70.2

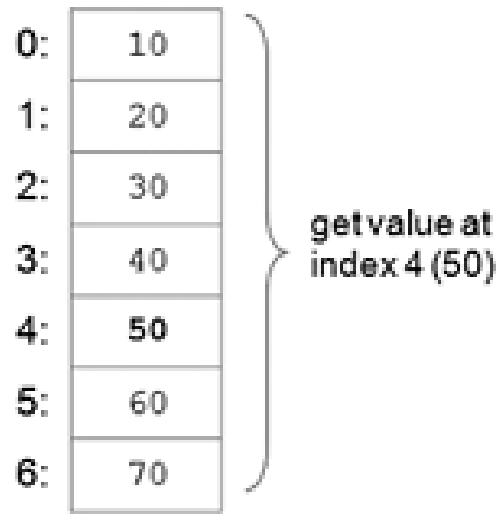
2: 67.2

3: 71.8

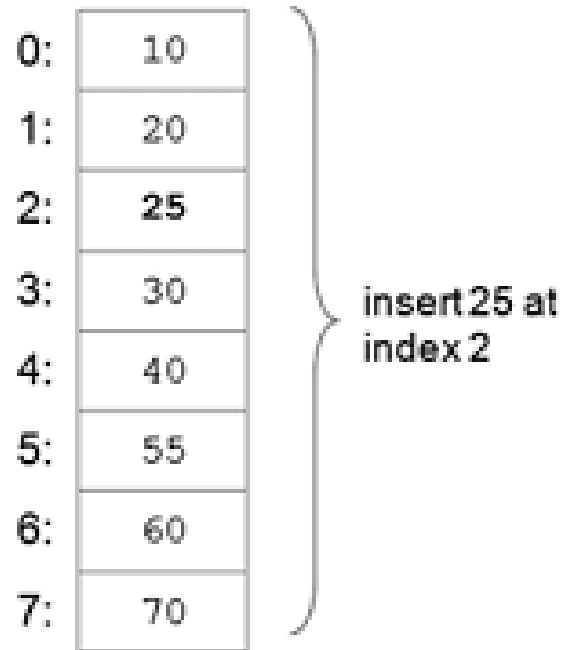
4: 73.2

5: 75.6

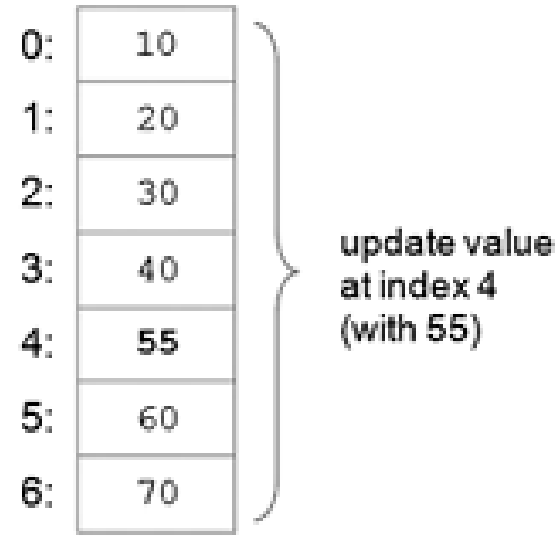
6: 74.0



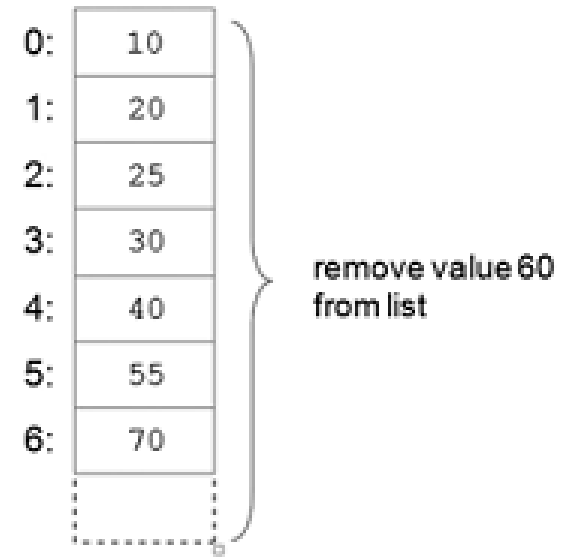
(a) retrieve



(c) insert



(b) replace



(d) remove

# Common List Operations

Operations commonly performed on lists include retrieve, update, insert, remove, and append.

# List Traversal

- A list traversal is a means of accessing, one-by-one, the elements of a list either from first to last or last to first element.

Adding up all  
values in the list

				sum
0:	10	←	+10	10
1:	20	←	+20	30
2:	30	←	+30	60
3:	40	←	+40	100
4:	50	←	+50	150
5:	60	←	+60	210
6:	70	←	+70	280

Searching for the  
value 50 in the list

			Find
0:	10	← 50?	no
1:	20	← 50?	no
2:	30	← 50?	no
3:	40	← 50?	no
4:	50	← 50?	<b>yes</b>
5:	60		
6:	70		

# MCQs: Answers

1. What would be the range of index values for a list of 10 elements?

~~(a) 0–9~~ (b) 0–10 (c) 1–10

2. Which one of the following is NOT a common operation on lists?

(a) access (b) replace ~~(c) interleave~~ (d) append (e) insert (f) delete

3. Which of the following would be the resulting list after inserting the value 50 at index 2?

0:	35
1:	15
2:	45
3:	28

~~(a)~~

0:	35
1:	50
2:	15
3:	45
4:	28

~~(b)~~

0:	35
1:	15
2:	50
3:	45
4:	28

~~(c)~~

0:	50
1:	35
2:	15
3:	45
4:	28

# MCQs: Answers

1. What would be the range of index values for a list of 10 elements?  
(a) **0–9** (b) 0–10 (c) 1–10
2. Which one of the following is NOT a common operation on lists?  
(a) access (b) replace (c) **interleave** (d) append (e) insert (f) delete
3. Which of the following would be the resulting list after inserting the value 50 at index 2?

0:	35
1:	15
2:	45
3:	28

(a)

0:	35
1:	50
2:	15
3:	45
4:	28

(b)

0:	35
1:	15
2:	50
3:	45
4:	28

(c)

0:	50
1:	35
2:	15
3:	45
4:	28



# Lists (Sequences) in Python

- A **list** in Python is a mutable linear data structure, denoted by a comma-separated list of elements within square brackets, allowing mixed-type elements. *Mutable* means that the contents of the list may be altered.
- Elements are indexed from 0 to n-1 , where n is number of elements in the list.
- **Example:** [1, 2, 3],                      ['one', 'two', 'three'],                      ['apples', 50, True]

<code>lst = [1, 2, 3]</code>	<code>lst[0] → 1</code>	access of first element
	<code>lst[1] → 2</code>	access of second element
	<code>lst[2] → 3</code>	access of third element

- An empty list is denoted by an empty pair of square brackets, [].
- Negative indexing, i.e, -1 and -2 refers to the last and second last items. Ex. `lst[-1]=3`, `lst[-2] = 2`.
- Range Search : `lst[0:2] = [1, 2]`

# List Modification Operations in Python

Operation	<code>fruit = ['banana', 'apple', 'cherry']</code>	
Replace	<code>fruit[2] = 'coconut'</code>	<code>['banana', 'apple', 'coconut']</code>
Delete	<code>del fruit[1]</code>	<code>['banana', 'cherry']</code>
Insert	<code>fruit.insert(2, 'pear')</code>	<code>['banana', 'apple', 'pear', 'cherry']</code>
Append	<code>fruit.append('peach')</code>	<code>['banana', 'apple', 'cherry', 'peach']</code>
Sort	<code>fruit.sort()</code>	<code>['apple', 'banana', 'cherry']</code>
Reverse	<code>fruit.reverse()</code>	<code>['cherry', 'banana', 'apple']</code>

# Tuples

- A tuple is an ordered and immutable linear data structure and used for different Data Types. Thus, in contrast to lists, once a tuple is defined, it cannot be altered.
- Tuples are denoted by parentheses instead of square brackets.
- **Example:** *nums = (10, 20, 30)*  
*student = ('John Smith', 48, 'Computer Science', 3.42)*
- Tuples of one element must include a comma following the element. Otherwise, the parenthesized element will not be made into a tuple.

CORRECT

```
>>> (1,)  
(1)
```

WRONG

```
>>> (1)  
1
```

- An empty tuple is represented by a set of empty parentheses, ().

```
>>> nums[0]  
10
```

```
>>> student[0]  
'John Smith'
```

# Tuples: Cont..

- A Python tuple is created using parentheses around the elements in the tuple. Although using parentheses in tuple is only optional, it is considered a good practice to use them.
- To access an element of a tuple, we simply use the index of that element. We use square brackets.
- Reverse Indexing by using indexes as  $-1$ ,  $-2$ ,  $-3$ , and so on, where  $-1$  represents the last element.
- Slicing that is, extract some elements from the tuple.

```
>>> a = (1,2,3,4)
>>> print(a)
(1,2,3,4)
>>> a = ('ABC','DEF','XYZ')
>>> print(a)
(ABC,DEF,XYZ)
```

```
>>> a = (1,2,3,4)
>>> print(a[1])
2
>>> a = ('ABC','DEF','XYZ')
>>> print(a[2])
XYZ
```

```
>>> a = (1,2,3,4)
>>> print(a[-1])
4
>>> a = ('ABC','DEF','XYZ')
>>> print(a[1:])
(DEF, XYZ)
```

# What will happen?

From the Python Shell, enter the following and observe the results.

```
>>> t = (10, 20, 30)
```

```
>>> t[0]
```

```
???
```

10

```
>>> del t[2]
```

```
???
```

del

```
>>> t.insert(1, 15)
```

```
>>> ???
```

```
???
```

→

```
>>> t.append(40)
```

```
???
```

→

# Characteristics of Tuple

- **Tuple Item:**
  - Tuple items are ordered, unchangeable, and allow duplicate values.
  - Tuple items are indexed, the first item has index **[0]**, the second item has index **[1]** etc.
- **Ordered:**
  - The items have a defined order, and that order will not change.
- **Unchangeable:**
  - we cannot change, add or remove items after the tuple has been created.
- **Allow Duplicates:**
  - Since tuple are indexed, tuples can have items with the same value

# Characteristics of Tuple (Cont..)

- You can access tuple items by referring to the **index number**, inside square brackets:

```
>>> a = ("Bennett", "University", "Computer")
>>> print(a[1])
University
```

- **Negative Indexing:** tuples are defined as objects with the data type 'tuple'.

```
>>> a = (1, 2, 3, 4)
>>> print(type(a))
<class 'tuple'>
```

- **The tuple() Constructor:** It is also possible to use the **tuple()** constructor to make a tuple.
  - Using the **tuple()** method to make a tuple:

```
>>> tup = tuple(("apple", "banana", "cherry")) # note the double round-brackets
>>> print(tup)
```

# Sequences and Sequence Operations in Python

- A sequence in Python is a linearly ordered set of elements accessed by an index number. Lists, tuples, and strings are all sequences.
- Strings, like tuples, are immutable ; therefore, they cannot be altered

Operation		String s = 'hello' w = '!'	Tuple s = (1,2,3,4) w = (5,6)	List s = [1,2,3,4] w = [5,6]
Length	len(s)	5	4	4
Select	s[0]	'h'	1	1
Slice	s[1:4]	'ell'	(2, 3, 4)	[2, 3, 4]
	s[1:]	'ello'	(2, 3, 4)	[2, 3, 4]
Count	s.count('e')	1	0	0
	s.count(4)	<i>error</i>	1	1
Index	s.index('e')	1	--	--
	s.index(3)	--	2	2
Membership	'h' in s	True	False	False
Concatenation	s + w	'hello!'	(1, 2, 3, 4, 5, 6)	[1, 2, 3, 4, 5, 6]
Minimum Value	min(s)	'e'	1	1
Maximum Value	max(s)	'o'	4	4
Sum	sum(s)	<i>error</i>	10	10



# What Will Happen ?

From the Python Shell, enter the following and observe the results.

<pre>&gt;&gt;&gt; s = 'coconut' &gt;&gt;&gt; s[4:7] ???</pre>	<pre>&gt;&gt;&gt; s = (10, 30, 20, 10) &gt;&gt;&gt; s[1:3] ???</pre>	<pre>&gt;&gt;&gt; s = [10, 30, 20, 10] &gt;&gt;&gt; s[1:3] ???</pre>
<pre>&gt;&gt;&gt; s.count('o') ???</pre>	<pre>&gt;&gt;&gt; s.count(10) ???</pre>	<pre>&gt;&gt;&gt; s.count(10) ???</pre>
<pre>&gt;&gt;&gt; s.index('o') ???</pre>	<pre>&gt;&gt;&gt; s.index(10) ???</pre>	<pre>&gt;&gt;&gt; s.index(10) ???</pre>
<pre>&gt;&gt;&gt; s + ' juice' ???</pre>	<pre>&gt;&gt;&gt; s + (40, 50) ???</pre>	<pre>&gt;&gt;&gt; s + (40, 50) ???</pre>

# Nested Lists

- Lists and tuples can contain elements of any type, including other sequences.
- Thus, lists and tuples can be nested to create arbitrarily complex data structures.
- **Example:** `class_grades = [ [85, 91, 89], [78, 81, 86], [62, 75, 77], ...]`,

Here, `class_grades[0]` equals `[85, 91, 89]`, and `class_grades[1]` equals `[78, 81, 86]`

`class_grades[0][0] → [85, 91, 89][0] → 85`

# Exercise

(Consider the Previous List)

- Calculate the class average on the first exam: a while loop can be constructed that iterates over the first grade of each student's list of grades.

```
sum = 0
k = 0

while k < len(class_grades):
    sum = sum + class_grades[k][0]
    k = k + 1

average_exam1 = sum / float(len(class_grades))
```

- We can produce a new list containing the exam average for each student in the class as follows

```
exam_avgs = []
k = 0

while k < len(class_grades):
    avg = (class_grades[k][0] + class_grades[k][1] + \
           class_grades[k][2]) / 3.0

    exam_avgs.append(avg)
    k = k + 1
```

# What will happen?

From the Python Shell, enter the following and observe the results.

```
>>> lst = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

```
>>> lst[0]
```

```
???
```

```
>>> lst[0][1]
```

```
???
```

```
>>> lst[1]
```

```
???
```

```
>>> lst[1][1]
```

```
???
```

# List Methods

Example List:

```
fruits = ['apple', 'banana', 'cherry']
```

Method	Description	Example Use
<a href="#">append()</a>	Adds an element at the end of the list.	fruits.append("orange")
<a href="#">clear()</a>	Removes all the elements from the list	fruits.clear()
<a href="#">copy()</a>	Returns a copy of the list	x = fruits.copy()
<a href="#">count()</a>	Returns the number of elements with the specified value	x = fruits.count("cherry")
<a href="#">extend()</a>	Add the elements of a list (or any iterable), to the end of the current list	fruits.extend([1, 2])
<a href="#">index()</a>	Returns the index of the first element with the specified value	x = fruits.index("cherry")
<a href="#">insert()</a>	Adds an element at the specified position	fruits.insert(1, "orange")
<a href="#">pop()</a>	Removes the element at the specified position (0: first, 1: second, etc.), default value is -1, which returns the last item (pop returns the popped item)	fruits.pop(1)
<a href="#">remove()</a>	Removes the item with the specified value	fruits.remove("banana")
<a href="#">reverse()</a>	Reverses the order of the list	fruits.reverse()
<a href="#">sort()</a>	Sorts the list (default: ascending). Descending: fruits.sort(reverse=True)	cars.sort()

# MCQs

- Which of the following sequence types is a mutable type?  
a) strings   ☒ b) lists   c) tuples
- Which of the following is true?  
a) Lists and tuples are denoted by the use of square brackets.  
☒ b) Lists are denoted by use of square brackets and tuples are denoted by the use of parentheses.  
c) Lists are denoted by use of parentheses and tuples are denoted by the use of square brackets.
- Lists and tuples must each contain at least one element.  
a) True  
☒ b) False
- For `lst = [4, 2, 9, 1]`, what is the result of the following operation, `lst.insert(2, 3)` ?  
☒ a) `[4, 2, 3, 9, 1]`   b) `[4, 3, 2, 9, 1]`   c) `[4, 2, 9, 2, 1]`
- Which of the following is the correct way to denote a tuple of one element?  
a) `[6]`   b) `(6)`   c) `[6,]`   ☒ d) `(6,)`
- Which of the following set of operations can be applied to any sequence?  
☒ a) `len(s)`, `s[i]`, `s+w` (concatenation)  
b) `max(s)`, `s[i]`, `sum(s)`  
☒ c) `len(s)`, `s[i]`, `s.sort()`

# MCQs: Answers

- Which of the following sequence types is a mutable type?  
a) strings   **b) lists**   c) tuples
- Which of the following is true?  
a) Lists and tuples are denoted by the use of square brackets.  
**b) Lists are denoted by use of square brackets and tuples are denoted by the use of parentheses.**  
c) Lists are denoted by use of parentheses and tuples are denoted by the use of square brackets.
- Lists and tuples must each contain at least one element.  
a) True  
**b) False**
- For *lst = [4, 2, 9, 1]*, what is the result of the following operation, *lst.insert(2, 3)* ?  
**a) [4, 2, 3, 9, 1]**   b) [4, 3, 2, 9, 1]   c) [4, 2, 9, 2, 1]
- Which of the following is the correct way to denote a tuple of one element?  
a) [6]   b) (6)   c) [6,]   **d) (6,)**
- Which of the following set of operations can be applied to any sequence?  
**a) len(s), s[i], s+w (concatenation)**  
b) max(s), s[i], sum(s)  
c) len(s), s[i], s.sort()

**Thank You**