

CSE 160 23sp Final Exam Cheat Sheet

if/elif/else syntax

if **condition1**:

statements

elif **condition2**:

other statements

else:

more statements

for loop syntax

for **i** in **sequence**:

statements

function definition syntax

def **function_name**(**param1**, **param2**, ...):

statements

Function	Description
<code>range([<i>start</i>,] <i>stop</i> [, <i>step</i>])</code>	Returns a sequence of numbers from <i>start</i> (inclusive) to <i>stop</i> (exclusive) incremented by <i>step</i>
<code>len(<i>Lst</i>)</code>	Returns the number of elements in <i>Lst</i>

Lists

Function	Description
<code>lst = []</code>	Creates an empty list
<code>lst[<i>idx</i>]</code>	Returns the element in <i>Lst</i> at index <i>idx</i>
<code>lst[<i>start</i> : <i>end</i>]</code>	Returns a sublist of <i>Lst</i> from index <i>start</i> to index <i>end</i> (exclusive)
<code>lst[<i>start</i> : <i>end</i> : <i>step</i>]</code>	Returns a sublist of <i>Lst</i> from index <i>start</i> to index <i>end</i> (exclusive), incrementing by <i>step</i>
<code>lst.append(<i>eLmt</i>)</code>	Adds the element <i>eLmt</i> to the end of <i>Lst</i> . Returns None .
<code>lst.index(<i>eLmt</i>)</code>	Returns index of the first occurrence of <i>eLmt</i> in <i>Lst</i> , Error if <i>eLmt</i> is not in <i>lst</i>
<code>lst.count(<i>eLmt</i>)</code>	Returns the number of times <i>eLmt</i> occurs in <i>Lst</i>
<code>lst.remove(<i>eLmt</i>)</code>	Removes first occurrence of <i>eLmt</i> from <i>Lst</i> , Error if <i>eLmt</i> is not in <i>Lst</i> . Returns None .
<code>lst.pop(<i>idx</i>)</code> <code>lst.pop()</code>	Removes and returns the element at index <i>idx</i> in <i>Lst</i> . With no parameter, removes the last element in <i>Lst</i>
<code>lst.insert(<i>idx</i>, <i>eLmt</i>)</code>	Inserts an element <i>eLmt</i> in list at index <i>idx</i> . Returns None .

File I/O

Function	Description
<code>my_file = open(<i>filepath</i>)</code>	Opens the file with given <i>filepath</i> for reading, returns a file object
<code>my_file.close()</code>	Closes file <code>my_file</code>

```
# Process one line at a time:  
for line_of_text in my_file:  
    # process line_of_text
```

```
# Process entire file at once  
all_data_as_a_big_string = my_file.read()
```

Dictionaries

Function	Description
<code>my_dict = {}</code> <code>my_dict = dict()</code>	Creates a new, empty dictionary
<code>my_dict[key]</code>	Returns the value associated with the given <i>key</i> in <i>my_dict</i>
<code>list(my_dict.keys())</code>	Returns a list of keys in <i>my_dict</i>
<code>list(my_dict.values())</code>	Returns a list of values in <i>my_dict</i>

Sorting

Function	Description
<code>sorted(<i>collection</i> [,key=<i>sort_key</i>, reverse=<i>bool_val</i>])</code>	Returns a sorted copy of <i>collection</i> , based on optional sort key (<i>key</i>) and optional order preference (<i>reverse</i>)
<code><i>lst</i>.sort([key=<i>sort_key</i>, reverse=<i>bool_val</i>])</code>	Sorts the given list <i>lst</i> , based on optional sort key (<i>key</i>) and optional order preference (<i>reverse</i>), and returns None

Common Error Names

- IndexError – Index out of range
- KeyError – Key not found in dictionary
- IndentationError – Invalid indentation
- TypeError – Operation applied to invalid combination of types
- ValueError – Function gets properly typed argument, but invalid value
- SyntaxError – Invalid Python syntax
- NameError – Variable name not found
- FloatingPointError – Floating point operation fails
- RuntimeError – Otherwise Unknown Error

Graphs

Function	Description
<code>import networkx as nx</code>	Imports the graph library and aliases the library name to "nx", usable as "nx.Graph()"
<code>g = nx.Graph()</code>	Creates a new graph and assigns the variable g to reference it.
<code>g.add_edge("A", "B")</code>	Adds an edge between nodes "A" and "B", creating the nodes if needed.
<code>g.add_node("A")</code>	Adds node "A" to the graph
<code>g.neighbors("A")</code>	Returns a collection of the neighbors of node "A"
<code>g.nodes()</code> <code>g.edges()</code>	Returns sets of nodes and edges, respectively, in the graph.

Sets

Function	Description
<code>s1 = set()</code>	Creates a new empty set
<code>s1 = set([...])</code>	Create a new set containing all of the elements within the given list.
<code>s1 s2</code>	Evaluates to the union of s1 and s2
<code>s1 & s2</code>	Evaluates to the intersection of s1 and s2
<code>s1 - s2</code>	Evaluates to the difference of s1 and s2

Classes

Function	Description
<code>__init__(self)</code>	The function that is called during class construction, as in <code>Class()</code> .
<code>self</code>	Required parameter for all class methods (functions). Refers to the specific instance of the class. Can hold any number of arbitrary variables, as in self.name