CSE 160 26su Midterm 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
range([*start*,] *stop* [, *step*])	Returns a sequence of numbers from *start* (inclusive) to *stop* (exclusive) incremented by *step*
len(*lst*)	Returns the number of elements in *lst*

Lists

Function	Description
lst = []	Creates an empty list
lst[*idx*]	Returns the element in *lst* at index *idx*
lst[*start* : *end*]	Returns a sublist of *lst* from index *start* to index *end* (exclusive)
lst[**start : end : step**]	Returns a sublist of *lst* from index *start* (default 0) to index *end* (exclusive, default *len(lst)*), incrementing by *step*
lst.append(*elmt*)	Adds the element *elmt* to the end of *lst*. Returns **None**.
lst.extend(*other*)	Adds each of the elements in the list *other* to the end of *lst*. Returns **None**.
lst.index(*elmt*)	Returns index of the first occurrence of *elmt* in *lst*, error if *elmt* is not in lst
lst.count(*elmt*)	Returns the number of times *elmt* occurs in *lst*
lst.remove(*elmt*)	Removes first occurrence of *elmt* from *lst*, error if *elmt* is not in *lst*. Returns **None**.
lst.pop(*idx*) lst.pop()	Removes and returns the element at index *idx* in *lst*. With no parameter, removes the last element in *lst*
lst.insert(*idx*, *elmt*)	Inserts an element *elmt* in *lst* at index *idx*. Returns **None**.
lst.sort()	Sorts the given list *lst*. Returns **None**.
lst.reverse()	Reverses the order of elements in the list *lst*. Returns **None**.

File I/O

Function	Description
`my_file = open(`***`filepath`***`)`	Opens the file with given ***`filepath`*** for reading, returns a file object
`my_file.close()`	Closes file `my_file`
`with open(`***`filepath`***`) as `***`f`***`:` `  # read file`	Opens the file with given ***`filepath`*** for reading via the file object ***`f`*** in the body of the "with" statement.
`str.strip()`	Given a string `str`, remove any trailing or ending whitespace.
`str.split([separator])`	Given a string `str`, splits the string on the optional separator (will split on spaces if no separator is given) and returns a list of separated strings.

```
file = open(filepath, "r")

# read entire file at once
file.read()
file.close()
```

```
file = open(filepath, "r")

# read line by line
for line in file:
    # process line
```

```
file = open(filepath, "w")

# write entire file at once
file.write(string_to_write)
file.close()
```

Dictionaries

Function	Description
`my_dict` `= {}` ***`my_dict`*** `= dict()`	Creates a new, empty dictionary
`my_dict``[`***`key`***`]`	Returns the value associated with the given ***`key`*** in ***`my_dict`***
`del my_dict[`***`key`***`]`	Removes the ***`key`*** (and its associated value) from ***`my_dict`***
`list(`***`my_dict`***`.keys())`	Returns a list of keys in ***`my_dict`***
`list(`***`my_dict`***`.values())`	Returns a list of values in ***`my_dict`***
`list(`***`my_dict`***`.items())`	Returns a list of tuples of the form ***`(key, value)`***

```
# Process each key-value pair together:
for key, value in my_dict.items():
    # process key and value
```

```
# Process one key at a time
for key in my_dict:
    # use dictionary's key
```