Exercise 37
Symbol Review

Raymart Jay E. Canoy

August 4, 2023

1 Keywords

- del: Used to delete objects, variables, lists or part of a list.

```
# defining a class
class myClass:
        name = "John"

# defining a variable
x = "Hello"

# defining a list
y = ["apple", "banana", "cherry"]

del myClass, x, y
print(myClass, x, y)
```

- from: Used to import only a specified section from a module
- as: Used to create a alias
- global: Declares a global variable inside a function, and use it outside the function

```
def myFunc():
        global z
        z = "This is a global variable"

myFunc()
print(z)
```

- with: Used in exception handling to make the code cleaner and much more readable. It simplifies the management of common resources like file streams.

```
# 1) without using the with statement
file = open("file_path", "w")
file.write("hello world!")
file.close()

# 2) without using with statement
file = open("file_path", 'w')
try:
        file.write("Hello World!")
finally:
        file.close()

# 3) Using the with statement
with open("file_path", 'w') as file:
        file.write("Hello World!")
```

- assert: Used when debugging code. Lets you test if a condition in your code returns True, if not, the program will raise an AssertionError.

- pass: Used as a placeholder for future code. When pass statement is executed, nothing happens, but you avoid getting an error when empty code is not allowed.

```
def myFunc():
        pass
```

- yield: similar to a return statement used for returning values in Python which returns a generator object to the one who calls the function which contains yield, instead simply returning the value.

```
def find_even(list_):
        for elem_ in list_:
                if elem_ % 2 == 0:
                        yield elem_

list_ = [i for i in range(101)]
list_even = []

for j in find_even(list_):
        list_even.append(j)
```

```
def find_word(list_, word_):
        for word in list_:
                if word == word_:
                        yield word

list_ = "Geeks are gorgeous"
count = 0

for count_ in find_word(list_.lower()split(), "geeks":
        count += 1

print(count)
```

- break: Used to break out a for loop, or a while loop.
- try: Used in try...except blocks. Defines a block of code test if it contains any errors.
- except: A keyword used in the try...except blocks. It defines a block of code to run if the try block raises an error.
- finally: Used in try...except...else blocks if the block is final. This block will be executed no matter if the try block raises an error or not.
- exec: Used for the dynamic execution of Python programs which can either be a string or object code.

```
exec(object[, globals[, locals]])
>>> exec("print('Add: %d' % (5+1))")
>>> exec("print(dir())", {})
>>> exec("print(dir())", {'dir':dir, 'fact':factorial})
```

- in
- raise: Used to raise an exception.
- continue
- is
- lambda