OS module in python
The OS module in python provides functions for interacting with the operating system. OS, comes under Python’s standard utility modules.

1
```
import os
print(os.name)
```

2
```
import os
print(os.getcwd())
# To print absolute path on your system
# os.path.abspath('.')

# To print files and directories in the current directory
# on your system
# os.listdir('.')
```

3
```
import os
try:
    # If the file does not exist,
    # then it would throw an IOError
    filename = 'GFG.txt'
    f = open(filename, 'rU')
    text = f.read()
    f.close()

# Control jumps directly to here if
#any of the above lines throws IOError.
except IOError:

    # print(os.error) will <class 'OSError'>
    print('Problem reading: ' + filename)

# In any case, the code then continues with
# the line after the try/except
```

File object manipulation
4
```
import os
fd = "GFG.txt"
```

```
# popen() is similar to open()
file = open(fd, 'w')
file.write("Hello")
file.close()
file = open(fd, 'r')
text = file.read()
print(text)

# popen() provides a pipe/gateway and accesses the file directly
file = os.popen(fd, 'w')
file.write("Hello")
# File not closed, shown in next function.
```

5

```
import os
fd = "GFG.txt"
file = open(fd, 'r')
text = file.read()
print(text)
os.close(file)
```

6

```
import os
fd = "GFG.txt"
os.rename(fd,'New.txt')
os.rename(fd,'New.txt')
```

Analysis the code and explain

1

```
class Person(object):


    def __init__(self, name):
            self.name = name

        def getName(self):
            return self.name


    def isEmployee(self):
            return False
```

```
class Employee(Person):


    def isEmployee(self):
            return True

emp = Person("nerd1") # An Object of Person
print(emp.getName(), emp.isEmployee())

emp = Employee("nerd2") # An Object of Employee
print(emp.getName(), emp.isEmployee())
```

2

```
class MyClass:


    __hiddenVariable = 0


    def add(self, increment):
            self.__hiddenVariable += increment
            print (self.__hiddenVariable)

myObject = MyClass()
myObject.add(2)
myObject.add(5)
```

3

```
print (myObject.__hiddenVariable)

class MyClass:


    __hiddenVariable = 10

myObject = MyClass()
print(myObject._MyClass__hiddenVariable)
```