**import** random

**class** **Lottery**:

**def** **shuffle**(self):

results = []

**for** i **in** range(5):

results.append(random.randint(1, 20))

**return** results

**class** **PowerBall**(Lottery):

**def** **shuffle**(self):

results = []

**for** i **in** range(6):

results.append(random.randint(1, 99))

**return** results

**class** **Airplane**:

**def** **\_\_init\_\_**(self, first\_class, business\_class, coach):

self.first\_class = first\_class

self.business\_class = business\_class

self.coach = coach

**def** **total**(self):

**return** self.first\_class + self.business\_class + self.coach

**class** **Train**:

**def** **\_\_init\_\_**(self, car1, car2, car3, car4, car5):

self.car1 = car1

self.car2 = car2

self.car3 = car3

self.car4 = car4

self.car5 = car5

**def** **total**(self):

**return** self.car1 + self.car2 + self.car3 + self.car4 + self.car5

**def** **passengers**(obj):

print(f'There are {obj.total()} passengers on board.')

**class** **Characters**:

**def** **\_\_init\_\_**(self, phrases):

self.phrases = phrases

**def** **total\_characters**(self):

total = 0

**for** phrase **in** self.phrases:

total += len(phrase)

**return** total

**def** **\_\_gt\_\_**(self, obj):

**return** self.total\_characters() > obj.total\_characters()

**def** **\_\_lt\_\_**(self, obj):

**return** self.total\_characters() < obj.total\_characters()

**def** **\_\_eq\_\_**(self, obj):

**return** self.total\_characters() == obj.total\_characters()

sample\_phrases1 = ['cat in the hat', 'green eggs and ham', 'the lorax']

sample\_phrases2 = ['the taming of the shrew', 'hamlet', 'othello']

c1 = Characters(sample\_phrases1)

c2 = Characters(sample\_phrases2)

print(c1 > c2) *# prints 'True'*

print(c1 < c2) *# prints 'False'*

print(c1 == c1) *# prints 'True'*

**class** **Median**:

**def** **calculate\_median**(self, n1, n2, n3=None, n4=None, n5=None):

**if** n3 **is** **not** **None** **and** n4 **is** **not** **None** **and** n5 **is** **not** **None**:

numbers = [n1, n2, n3, n4, n5]

**elif** n3 **is** **not** **None** **and** n4 **is** **not** **None**:

numbers = [n1, n2, n3, n4]

**elif** n3 **is** **not** **None**:

numbers = [n1, n2, n3]

**else**:

numbers = [n1, n2]

numbers.sort()

median\_index = len(numbers) // 2

**if** len(numbers) % 2 == 0:

median = (numbers[median\_index] + numbers[median\_index - 1]) / 2

**else**:

median = numbers[median\_index]

**return** median

source\_file = '/home/codio/workspace/code/polymorphism/text\_1\_exercise5.txt'

answer\_file = '/home/codio/workspace/code/polymorphism/answer\_exercise5.txt'

**class** **Substitute**:

**def** **\_\_init\_\_**(self, source\_file, answer\_file):

self.source\_file = source\_file

self.answer\_file = answer\_file

self.words = **None**

**def** **string\_to\_list**(self):

'''Read text file, turn it into a

2D list of words for each line'''

words = []

**with** open(self.source\_file, 'r') **as** file\_object:

lines = file\_object.read().split('\n')

**for** line **in** lines:

words.append(line.split())

self.words = words

**def** **list\_to\_string**(self):

'''Convert 2D list back into a

string with newline characters'''

lines = []

**for** line **in** self.words:

lines.append(' '.join(line))

string = '\n'.join(lines)

self.words = string

**def** **swap\_words**(self):

self.string\_to\_list()

**for** line **in** self.words:

**for** i **in** range(len(line)):

**if** (i + 1) % 5 == 0:

word = line[i]

line[i] = 'HELLO'

self.list\_to\_string()

**class** **Stars**(Substitute):

**def** **swap\_words**(self):

self.string\_to\_list()

**for** line **in** self.words:

**for** i **in** range(len(line)):

**if** (i + 1) % 3 == 0:

word = line[i]

line[i] = '\*' \* len(word)

self.list\_to\_string()

file = open(self.answer\_file, 'w')

file.writelines(self.words)

file.close()

s = Stars(source\_file, answer\_file)

s.swap\_words()