Assignment2

Zilin Xiong

September 5, 2021

[4]:
```
##Assignment2
##Standard Library
##Question1
import calculator
import numpy as np

list1 = [1,2,3,4,5,6]
def print_result(list1):
    print([(min(list1),max(list1),sum(list1)/len(list1))])

##Question2
int_1 = int(1.0)
print(int_1)
int_2 = int_1
int_2 = int(2.0)
int_1 == int_2
print(int_1)  ##int is immutable

str_1 = str(66)
str_2 = str_1
str_2 = str(88)
str_1 == str_2
print(str_1)  ##str is immutable

list_1 = [1,2,8]
list_2 = list_1
list_2[2] = 10
list_1 == list_2
print(list_1) ##list is mutable

tuple_1 = (1,2)
print(tuple_1)
print(type(tuple_1))
tuple_2 = tuple_1
tuple_2 = (4,5)
tuple_1 == tuple_2
print(tuple_1)  ##tuple is immutable
```

```
set_1 = {'a','b','c'}
set_2 = set_1
set_2 = {'e','f'}
set_1 == set_2 ##set is immutable

##Question3

a1 = 3
a2 = calculator.pro_of_two(a1,a1)
b1 = 4
b2 = calculator.pro_of_two(b1,b1)
c2 = calculator.sum_of_two(a2,b2)
c1 = calculator.sqrt_of_sum(c2)
print(c1)


###Intro to Numpy
##Question1
A = np.array([[3,-1,4],[1,5,-9]])
B = np.array([[2,6,-5,3],[5,-8,9,7],[9,-3,-2,-3]])
print(A, B)
np.dot(A,B)
A@B

##Question2
A = np.array([[3,1,4],[1,5,9],[-5,3,1]])
def matrix_function(x):
    Y = -x@x@x+9*(x@x)-15*x
    return print(Y)

matrix_function(A)


A = np.array([[0,2,4],[1,3,5]])
B = np.array([[3,0,0],[3,3,0],])


##Question3
A = np.array([0,1,2,3,4,5]).reshape(3,2)
A = A.T
print(A)

B = np.full((3,3),3)
B = np.tril(B)
print(B)
```

```
C = np.identity(3)
C = C - C*3
print(C)

I = np.identity(3)
print(I)

AT = A.T
AT

O1 = np.zeros((3,4))
O1

#first line adding
first_line = np.hstack((O1,AT,I))
first_line

#second line
O2 = np.zeros((2,3))
O2
second_line = np.hstack((A,O2,O2))
second_line

#thrid line
O3 = np.zeros((3,3))
third_line = np.hstack((B,O3,C))
third_line

#final step
final = np.vstack((first_line,second_line,third_line))
print(final)




### Object Oriented Programming

class Backpack:
    '''We try to create a Backpack object. Which has a name, a color, a max_size␣
 ↪and
    a list of contents.
    Attributes:
        name(str):the name of the backpack's owner.
        color(str):the color of backpack.
        max_size(int):the number of the backpack.
        contents(list):the contents of the backpack.
    '''
    def __init__(self,name,color,max_size = 5): #This function is the␣
 ↪constructor.
        '''Set the name, color and max_size as a default constent.
           Initialize an enpty list of contents.
        '''
        self.name = name
        self.color = color
        self.max_size = max_size
        self.contents = []

    def put(self,item):
        '''Add 'item' to the backpack's list of contents.'   '''
        if len(self.contents) >= self.max_size:
            print("No Room!")
        else:
            self.contents.append(item)

    def take(self,item):
        '''Remove 'item' to the backpack's list of contents.'   '''
        self.contents.remove(item)

    def dump(self):
        '''Empty all 'item' to the backpack's list of contents.'   '''
        # while len(self.contents) >=1:
            # self.take(self.contents[0])
            # print(self.contents)
            # self.contents.remove(self.contents[0])
            # print(self.contents)
        self.contents.clear()
        print(self.contents)

##question2
class Jetpack(Backpack):
    '''A jetpack object class. Inherits from the Backpack class.
       Atttributes:
           name(str):the name of the jetpack.
           color(str):the color of the jetpack.
           max_size(int):the contetnts of the jetpack.
           amount_fuel:the amount of the fuel.
    '''
    def __init__(self,name,color,max_size = 2,amount_fuel = 10):
        '''Use the Backpack constructor to initialize the name, color, and␣
 ↪max_size attributes.
           A jetpack only holds limited fuel.
           Parameters:
               name(str):the name of the jetpack.
               color(str):the color of the jetpack.
```

```
                max_size(int):the maximum number of the items.
                anount_fuel(int):the amount of the fuel carried.
        '''
        Backpack.__init__(self,name,color,max_size)
        self.amount_fuel = amount_fuel

    def fly(self,amount):
        '''Define a new method to calculate the fuel amount.
        '''
        if  self.amount_fuel < amount:
            print("Not Enough Fuel!")
        else:
            self.amount_fuel = self.amount_fuel - amount
    def dump(self):
        '''Clear all the fuel in the jetpack.
        '''
        self.contents.clear()
        self.amount_fuel = 0

##test
def test_back():
    '''A test of function.   '''
    test_back = Backpack("2","5",5)
    if test_back.name != "20":
        print("2 is good")
    for item in ["one","two","three","four","five","six"]:
        test_back.put(item)
    test_back.dump()

def test_jetpack():
    '''A test function to test if it reports not enough fuel.
    '''
    test_jetpack = Jetpack("namehere","colorhere",6,12)
    test_jetpack.fly(8)
    print(test_jetpack.amount_fuel)
    test_jetpack.fly(5)

if __name__ =="__main__":
    test_jetpack()
```

4

[3]:
```
##calculator
import math as mt
##sum of two
def sum_of_two(x,y):
    num_of_two1 = [x,y]
    return sum(num_of_two1)

##product of two
def pro_of_two(x,y):
    return x*y

##sqrt
def sqrt_of_sum(x):
    return mt.sqrt(x)

if __name__== "__main__":
    print(sum_of_two(1,3))
    pro_of_two(5,6)
    sqrt_of_sum(6)
```

```
4
```

[ ]: