HW_2_Function

October 26, 2023

0.1 Exercise 2: functin and class

20215178 1 ” ” “ ” 2 2023.11.3 3 ipynb pdf ;

```
[ ]: # 1. Convert Minutes into Seconds
     # Write a function that takes an integer minutes and converts it to seconds.

     # Examples
     # convert(5)   300
     # convert(3)   180
     # convert(2)   120
     def convert(minutes):
         return minutes * 60

     print(convert(5))
     print(convert(3))
     print(convert(2))
```

```
300
180
120
```

```
[ ]: # 2. String to Integer and Vice Versa
     # Write two functions:
     # to_int() : A function to convert a string to an integer.
     # to_str() : A function to convert an integer to a string.

     # Examples
     # to_int("77")   77
     # to_int("532")   532
     # to_str(77)   "77"
     # to_str(532)   "532"
     def to_int(txt):
         return int(txt)

     def to_str(num):
         return str(num)
```

```
print(to_int("77"))
print(to_int("532"))
print(to_str(77))
print(to_str(532))
```

```
77
532
77
532
```

[ ]:

```
# 3 Let's Sort This List!
# Create a function that takes a list of numbers lst, a string s and return a␣
 ↪list of numbers as per the following rules:
# "Asc" returns a sorted list in ascending order.
# "Des" returns a sorted list in descending order.
# "None" returns a list without any modification.

# Examples
# asc_des_none([4, 3, 2, 1], "Asc" )   [1, 2, 3, 4]
# asc_des_none([7, 8, 11, 66], "Des")   [66, 11, 8, 7]
# asc_des_none([1, 2, 3, 4], "None")   [1, 2, 3, 4]

def asc_des_none(lst, s):
    if s == "Asc":
        return sorted(lst)
    elif s == "Des":
        return sorted(lst, reverse=True)
    else:
        return lst

print(asc_des_none([4, 3, 2, 1], "Asc" ))
print(asc_des_none([7, 8, 11, 66], "Des"))
print(asc_des_none([1, 2, 3, 4], "None"))
```

```
[1, 2, 3, 4]
[66, 11, 8, 7]
[1, 2, 3, 4]
```

[ ]:

```
# 4. Circle or Square
# Given the radius of a circle and the area of a square, return True if the␣
 ↪circumference of the circle is greater than the square's perimeter and False␣
 ↪if the square's perimeter is greater than the circumference of the circle.

# Examples
# circle_or_square(16, 625)   True
# circle_or_square(5, 100)   False
# circle_or_square(8, 144)   True
```

```
import math

def circle_or_square(rad, area):
    if (2 * math.pi * rad) > (4 * math.sqrt(area)):
        return True
    else:
        return False

print(circle_or_square(16, 625))
print(circle_or_square(5, 100))
print(circle_or_square(8, 144))
```

```
True
False
True
```

[ ]:

```
# 5. Length of Number
# Create a function that takes a number num and returns its length.

# Examples
# number_length(10)   2
# number_length(5000)   4
# number_length(0)   1

def number_length(num):
    return len(str(num))

print(number_length(10))
print(number_length(5000))
print(number_length(0))
```

```
2
4
1
```

[ ]:

```
# 6. Get the Area of a Country
# Create a function that takes a country's name and its area as arguments and returns the area of the country's proportion of the total world's landmass.

# Examples
# area_of_country("Russia", 17098242)   "Russia is 11.48% of the total world's landmass"
# area_of_country("USA", 9372610), "USA is 6.29% of the total world's landmass"
# area_of_country("Iran", 1648195)   "Iran is 1.11% of the total world's landmass"

# Notes
# The total world's landmass is 148,940,000 [Km^2]
# Round the result to two decimal places.

def area_of_country(name, area):
    print("{} is {:.2%} of the total world's landmass".format(name, area /␣
 ↪148940000))

area_of_country("Russia", 17098242)
area_of_country("USA", 9372610)
area_of_country("Iran", 1648195)
```

```
Russia is 11.48% of the total world's landmass
USA is 6.29% of the total world's landmass
Iran is 1.11% of the total world's landmass
```

[ ]:

```
# 7. List of Multiples
# Create a function that takes two numbers as arguments (num, length) and␣
 ↪returns a list of multiples of num until the list length reaches length.

# Examples
# list_of_multiples(7, 5)   [7, 14, 21, 28, 35]
# list_of_multiples(12, 10)   [12, 24, 36, 48, 60, 72, 84, 96, 108, 120]
# list_of_multiples(17, 6)   [17, 34, 51, 68, 85, 102]

def list_of_multiples(num, length):
    return [num * i for i in range(1, length + 1)]

print(list_of_multiples(7, 5))
print(list_of_multiples(12, 10))
print(list_of_multiples(17, 6))
```

```
[7, 14, 21, 28, 35]
[12, 24, 36, 48, 60, 72, 84, 96, 108, 120]
[17, 34, 51, 68, 85, 102]
```

[ ]:

```
# 8. Factorial of Factorials
# Create a function that takes an integer n and returns the factorial of␣
 ↪factorials. See below examples for a better understanding:

# Examples
# fact_of_fact(4)   288
# # 4! * 3! * 2! * 1! = 288
# fact_of_fact(5)   34560
# fact_of_fact(6)   24883200

def fact_of_fact(num):
    fact = 1
    for i in range(1, num + 1):
        fact *= math.factorial(i)
    return fact

print(fact_of_fact(4))
print(fact_of_fact(5))
print(fact_of_fact(6))
```

```
288
34560
24883200
```

[ ]:

```
# 9. Censor Words from List
# Create a function that takes a string txt and censors any word from a given␣
 ↪list lst. The text removed must be replaced by the given character char.

# Examples
# censor_string("Today is a Wednesday!", ["Today", "a"], "-")   "----- is -␣
 ↪Wednesday!"
# censor_string("The cow jumped over the moon.", ["cow", "over"], "*"), "The␣
 ↪*** jumped **** the moon.")
# censor_string("Why did the chicken cross the road?", ["Did", "chicken",␣
 ↪"road"], "*")   "Why *** the ******* cross the ****?"

def censor_string(txt, lst, char):
    for i in lst:
        txt = txt.replace(i, char * len(i))
    return txt

print(censor_string("Today is a Wednesday!", ["Today", "a"], "-"))
print(censor_string("The cow jumped over the moon.", ["cow", "over"], "*"))
print(censor_string("Why did the chicken cross the road?", ["Did", "chicken",␣
 ↪"road"], "*"))
```

```
----- is - Wednesd-y!
The *** jumped **** the moon.
Why did the ******* cross the ****?
```

[ ]:

```
# 10. Classes For Fetching Information on a Sports Player
# Create a class that takes the following four arguments for a particular␣
 ↪football player:
# name
# age
# height
# weight

# Also, create three functions for the class that returns the following strings:
# get_age() returns "name is age age"
# get_height() returns "name is heightcm"
```

```python
# get_weight() returns "name weighs weightkg"

# Examples
#  p1 = player("David Jones", 25, 175, 75)
#  p1.get_age()   "David Jones is age 25"
#  p1.get_height()   "David Jones is 175cm"
#  p1.get_weight()   "David Jones weighs 75kg"

class player:
    def __init__(self, name, age, height, weight):
        self.name = name
        self.age = age
        self.height = height
        self.weight = weight

    def get_age(self):
        return "{} is age {}".format(self.name, self.age)

    def get_height(self):
        return "{} is {}cm".format(self.name, self.height)

    def get_weight(self):
        return "{} weighs {}kg".format(self.name, self.weight)

p1 = player("David Jones", 25, 175, 75)
print(p1.get_age())
print(p1.get_height())
print(p1.get_weight())
```

```
David Jones is age 25
David Jones is 175cm
David Jones weighs 75kg
```