**Python Mega Assignment**

***Answers***

1. A) Value

B) Item

C) Key

1. False
2. False
3. D) None of the above
4. D) Value of the key, if it exists in the dictionary.
5. B) Tuple containing the pair of last item of the dictionary.
6. A) items()

B) keys()

1. B) Curly Braces {}

***Code:***

dictionary = {"Fahad" : 19, "Ahmed" : 25, "Ali" : 22}

f\_k = input("Enter the key to be searched: ")

f\_k = f\_k.title()

f\_v = int(input("Enter the value to be searched: "))

counter = 1

for key,value in dictionary.items():

if f\_k == key and f\_v == value:

print("Yes the key value pair exist in the dictionary.")

break

elif f\_k == key:

print("Only the key exist in the dictionary.")

break

elif f\_v == value:

print("Only the value exist in the dictionary.")

break

else :

if counter < len(dictionary):

counter+=counter

continue

else:

print("The given key and value pair does not exist in the dictionary.")



**Code:**

string = input("Enter a string: ")

string = string.split(" ")

dictionary = {}

for word in string:

dictionary[word] = 0

for word in string:

dictionary[word]+=1

print("\nThe words that appear even number of times in the string are: ")

for key,value in dictionary.items():

if value % 2 == 0:

print(key)

1. Basically, this would give an error because Aug and all other except “Feb” are unidentified by Python.

But if we assume that it was a printing error

Then the output will be as such:

X[0:3] = [“Feb”, ”Apr”, ”Mar”]

X[2:8] = [“Mar”, “May”, “Jun”, “Jul”, “Aug”, “Jan”]

X[4:9] = [“Jun”, “Jul”, “Aug”, “Jan”]

X[1:7:2] = [“Apr”, “May”, “Jul”]

X[-1:-7] = [“Jan”, “Aug”, “Jul”, “Jun”, “May”, “Mar”]

X[-7:7] = [“Apr”, “Mar”, “May”, “Jun”, “Jul”]

X[-1:-8:-2] = ['Jan', 'Jul', 'May', 'Apr']

X[:4] = [“Feb”, “Apr”, “Mar”, “May”]

1. 3) .remove(9)
2. 1) \*
3. 1) .upper()
4. A) bool

B) str

C) int

D) float

1. 2) p and R2
2. 4) p.X()
3. A) 0

B) 2.0

C) 2

D) 2

1. A) for i in x:

for j in i:

print(j)

B) for i in range(len(x) - 1):

for j in range(len(x[i])):

print(str(x[i][j]) + '\t' + str(x[i + 1][j]))

C) for i in x:

for j in i:

print(j, end = “\t”)

D) for i in x:

Print(\*i)

Or

for i in x:

for j in i:

print(j)

print(“”)

1. for j, z in enumerate(q):

print( 'Item ' + str( j ) + ' - ', str ( z ))

1. B)
2. A)

i) True

ii) True

iii) False

1. 11.333
2. Error as 'Ali' has to be specified as keyword argument otherwise it is considered as an element of the variadic positional argument called \*opt\_values, so last line should be written as avg ( 5 , 9 , 20, 34, 87, 112 , name = 'Ali' ) so that output is name is: AliMarks: 44.5
3. Winner and score are positional arguments so their value will be Manchester and 1-0 respectively. Other\_info is a variadic keyword argument due to which it is saved as a dictionary that can take arbitrary number of keys such as keys of 'overtime' and 'injuries' with values 'yes' and 'none' respectively.

Removing \* from other\_info makes it a variadic positional argument due to which it becomes a list that can take an arbitrary number of elements such as two elements 'yes' and 'none'. However, since it is positional argument, winner and score must be specified as positional arguments as well otherwise there is an error

Removing \*\* from other\_info makes it a normal positional argument so it cannot take more than one element as its value thus its value will be "yes" but there is no parameter for "none" which raises an error

1. Score is positional argument but \*\*other\_info is keyword argument but positional argument can’t come after keyword argument hence causing error.
2. B)
3. C)
4. C) No Output (because there is no print statement)

If we assume the print statement then output will be

D) ‘1.234560, 1.234560, 01.23’

1. No Output ( While False implies that while will not be executed due to false condition)

Output:

a a a a a a

1. Error as joined requires a string and can not be applied on list.
2. Error as second else is without any if and else can not be used alone.
3. 48 .

Lambda Function: It can take in any number of arguments but can have only one expression. Here lambda functions take in an argument name as x and multiply it with y which is 6 in this case and returns the value in z.

1. Output = 0

This is because in the function change we passed one as an argument, while i is defined to be zero. Printing an i will simply return 0 because of the value stored in it.

1. No question
2. No question
3. No question
4. Output:

<class 'tuple'>

(2, 3, 4)

1. <class 'dict'>

1. Line 1: [0]

Line 2: [0, 1]

Line 3: [0, 1, 2]

1. 3 \* (3)\*\*2 - (4)\*3

3 \* 9 - 12

27 - 12

15

1. D)

1. B) and C)
2. Code:

marks = int(input("Enter your marks: "))

if marks >= 90:

print("You got an A grade.")

elif marks >= 80 and marks < 90:

print("You got a B grade.")

elif marks >= 70 and marks < 80:

print("You got a C grade.")

elif marks > 60 and marks < 70:

print("You got a D grade.")

else:

print("You got an E grade.")