PYTHON ASSIGNMENT BOOK

MAKE A MOVE TO PYTHON

**ASSIGNMENTS**

\

**TASK ONE: NUMBERS AND VARIABLES**

1. Create three variables in a single line and assign different values to them and make sure their data types are invited different. Like one is int, another one is float and last one is string.

Code: x,y,z = 5, "Hello", 3+1j

print(x,y,z)

O/p: 5 Hello (3+1j)

2. Create a variable of value type complex and swap it with another variable whose value is an integer.

Code: z = 3+1j

x = 10

z , x = x, z

print(x,z)

O/P: (3+1j) 5

3. Swap two numbers using third variable as result name and do the same task without using any third variable.

**Using third variable**

Code: x,y = 2,8

z=y

y=x

x=z

print(x,y)

O/P: 8 2

**Without using third variable**

Code: x,y =2,8

x,y=y,x

print(x,y)

O/P: 8 2

4. Write a program to print the value given by the user by using both Python 2.x and Python 3.x Version.

**Python 2**

Code: name = raw\_input()

print name

O/P: Ruchi

Ruchi

**Python 3**

Code: print(input())

O/P: Ruchi

Ruchi

5. Write a program to complete the task given below:

* Ask user to enter any 2 numbers in between 1-10 and add both of them to another variable call z.
* Use z for adding 30 into it and print the final result by using variable result.

Code: x,y = input("enter any 2 numbers in between 1-10").split()

z=int(x)+int(y)

z=z+30

print(z)

O/P: enter any 2 numbers in between 1-10 7 9

46

6. Write a program to check the data type of the entered values. HINT: Printed output should say - The input value data type is : int/float/string/etc

Code: x= type(eval(input("enter a value to check data type")))

print("The data type of x is :", x)

O/P: enter a value to check data type7

The data type of x is : <class 'int'>

7. Create Variable using CamelCase, LadderCase and UPPERCASE. (Refer: <https://capitalizemytitle.com/camel-case/>)

Code: userInput = 4

UserInput= 18

USERCASE= 12

print(userInput,UserInput,USERCASE)

8. If one data type value is assigned to ‘a’ variable and then a different data type value is assigned to ‘a’ again. Will it change the value. If Yes then Why?

Yes, it will change the value because python is a dynamic language i.e. we don't explicitly define variable type.

**TASK TWO: OPERATORS AND DECISION MAKING STATEMENT**

1. Write a program in Python to perform the following operation:

* If a number is divisible by 3 it should print “Consultadd” as a string
* If a number is divisible by 5 it should print “c” as a string
* If a number is divisible by both 3 and 5 it should print “Consultadd Python Training” as a string.

Code: x=300

if x%3==0:

print("Consultadd")

if x%5==0:

print("c")

if x%3==0 and x%5==0:

print("Consultadd Python Training")

O/p: c

2. Write a program in Python to perform the following operator based task:

* Ask user to choose the following option first:
* If User Enter 1 - Addition
* If User Enter 2 - Subtraction
* If User Enter 3 - Division
* If USer Enter 4 - Multiplication
* If User Enter 5 - Average
* Ask user to enter the 2 numbers in a variable for first and second for the first 4 options mentioned above.
* Ask user to enter two more numbers as first and second2 for calculating the average as soon as user choose an option 5.
* At the end if the answer of any operation is Negative print a statement saying “NEGATIVE”
* NOTE: At a time user can perform one action at a time.

userChoice = int(input("Choose the operation: Enter 1 for Addition, Enter 2 for Subtraction, Enter 3 for Division,Enter 4 for Multiplication and Enter 5 for Average"))

first=int(input("Enter first operand"))

second=int(input("Enter second operand"))

if userChoice==1:

result=first+second

elif userChoice==2:

result=first-second

elif userChoice==3:

result=first/second

elif userChoice==4:

result=first\*second

elif userChoice==5:

first2=int(input("Enter third operand"))

second2=int(input("Enter fourth operand"))

result=(first+second+first2+second2)/4

else:

pass

print(result)

if result<0:

print("Negative")

O/P: Choose the operation: Enter 1 for Addition, Enter 2 for Subtraction, Enter 3 for Division,Enter 4 for Multiplication and Enter 5 for Average5

Enter first operand2

Enter second operand3

Enter third operand4

Enter fourth operand5

3.5

Choose the operation: Enter 1 for Addition, Enter 2 for Subtraction, Enter 3 for Division,Enter 4 for Multiplication and Enter 5 for Average3

Enter first operand19

Enter second operand10

1.9

3. Write a program in Python to implement the given flowchart:

Code:

a=10

b=20

c=30

avg=(a+b+c)/3

print("avg-", avg)

if (avg>a and avg>b and avg>c):

print("Average is higher than a,b,c")

elif (avg>a and avg>b):

print("Average is higher than a,b")

elif (avg>a and avg>c):

print("Average is higher than a,c")

elif (avg>b and avg>c):

print("Average is higher than b,c")

elif (avg>a):

print("Average is higher than a")

elif (avg>b):

print("Average is higher than b")

elif (avg>c):

print("Average is higher than c")

else:

pass

O/P:

avg- 20.0

Average is higher than a

4. Write a program in Python to break and continue if the following cases occurs:

* If user enters a negative number just break the loop and print “It’s Over”
* If user enters a positive number just continue in the loop and print “Good Going”

Code:

while True:

x=int(input("Enter any number"))

if x<0:

print("It's over")

break

elif x>0:

print("Going good")

continue

else:

pass

O/P:

Enter any number8

Going good

Enter any number9

Going good

Enter any number-8

It's over

5. Write a program in Python which will find all such numbers which are divisible by 7 but are not a multiple of 5, between 2000 and 3200.

Code:

i = 2000

while i<= 3200:

if i%7==0 and i%5!=0 :

print(i)

else:

pass

i+=1

O/P:

2002

2009

2016

2023

2037

2044

2051

2058

2072

2079

2086

2093

2107

2114

2121

2128

2142

2149

2156

2163

2177

2184

2191

2198

2212

2219

2226

2233

2247

2254

2261

2268

2282

2289

2296

2303

2317

2324

2331

2338

2352

2359

2366

2373

2387

2394

2401

2408

2422

2429

2436

2443

2457

2464

2471

2478

2492

2499

2506

2513

2527

2534

2541

2548

2562

2569

2576

2583

2597

2604

2611

2618

2632

2639

2646

2653

2667

2674

2681

2688

2702

2709

2716

2723

2737

2744

2751

2758

2772

2779

2786

2793

2807

2814

2821

2828

2842

2849

2856

2863

2877

2884

2891

2898

2912

2919

2926

2933

2947

2954

2961

2968

2982

2989

2996

3003

3017

3024

3031

3038

3052

3059

3066

3073

3087

3094

3101

3108

3122

3129

3136

3143

3157

3164

3171

3178

3192

3199

​

6. What is the output of the following code examples?

* x=123

for i in x:

print(i)

TypeError: 'int' object is not iterable

* i = 0

while i < 5:

print(i)

i += 1

if i == 3:

break

else:

print(“error”)

0

error

1

error

2

* count = 0

while True:

print(count)

count += 1

if count >= 5:

Break

0

1

2

3

4

NameError: name 'Break' is not defined

7. Write a program that prints all the numbers from 0 to 6 except 3 and 6.

Expected output: 0 1 2 4 5

Note: Use ‘continue’ statement

Code:

i=0

while i<6:

if i!=3:

print(i)

i+=1

if i==3:

continue

O/P:

0

1

2

4

5

8. Write a program that accepts a string as an input from user and calculate the number of digits and letters.

Expected output: consul12

Letters 6

Digits 2

Code:

alphacount=0

numcount=0

str = input("Enter any string")

for i in range(len(str)):

if str[i].isalpha():

alphacount +=1

if str[i].isnumeric():

numcount+=1

print(alphacount,numcount)

O/P:

Enter any stringconsultad123 training

17 3

9. Read the two parts of the question below:

* Write a program such that it asks users to “guess the lucky number”. If the correct number is guessed the program stops, otherwise it continues forever.

Code:

luckynumber = 9

num = 0

while num != luckynumber:

num = int(input("Guess the lucky number and enter here"))

if num == luckynumber:

break

O/P

Guess the lucky number and enter here8

Guess the lucky number and enter here4

Guess the lucky number and enter here12

Guess the lucky number and enter here9

* Modify the program so that it asks users whether they want to guess again each time. Use two variables, ‘number’ for the number and ‘answer’ for the answer to the question whether they want to continue guessing. The program stops if the user guesses the correct number or answers “no”. ( The program continues as long as a user has not answered “no” and has not guessed the correct number)

Code:

luckynumber = 9

num = int(input("Guess the lucky number and enter here"))

while True:

if num != luckynumber:

ans = input("Do you wish to guess again?")

if ans == "yes":

num = int(input("Guess the lucky number again and enter here"))

if ans == "no":

break

else:

break

O/P:

Guess the lucky number again and enter here7

Do you wish to guess again?yes

Guess the lucky number again and enter here8

Do you wish to guess again?no

Guess the lucky number and enter here5

Do you wish to guess again?yes

Guess the lucky number again and enter here9

10. Write a program that asks five times to guess the lucky number. Use a while loop and a counter, such as

counter=1

While counter <= 5:

print(“Type in the”, counter, “number”

counter=counter+1

The program asks for five guesses (no matter whether the correct number was guessed or not). If the correct number is guessed, the program outputs “Good guess!”, otherwise it outputs “Try again!”. After the fifth guess it stops and prints “Game over!”.

Code:

luckynumber = 9

count = 0

num = int(input("Guess the lucky number and enter here"))

while count<4:

if num != luckynumber:

print("try again!")

num = int(input("Guess the lucky number again and enter here"))

else:

print("Good guess")

num = int(input("Guess the lucky number again and enter here"))

count += 1

O/P:

Guess the lucky number and enter here6

try again!

Guess the lucky number again and enter here5

try again!

Guess the lucky number again and enter here9

Good guess

Guess the lucky number again and enter here8

try again!

Guess the lucky number again and enter here7

11. In the previous question, insert “break” after the “Good guess!” print statement. “break” will terminate the while loop so that users do not have to continue guessing after they found the number. If the user does not guess the number at all, print “Sorry but that was not very successful”.

Code:

luckynumber = 9

count = 0

num = int(input("Guess the lucky number and enter here"))

while count<4:

if num != luckynumber:

print("try again!")

num = int(input("Guess the lucky number again and enter here"))

else:

print("Good guess")

break

count += 1

if num != luckynumber:

print("Sorry but that was not very successful")

O/P:

Guess the lucky number and enter here7

try again!

Guess the lucky number again and enter here8

try again!

Guess the lucky number again and enter here6

try again!

Guess the lucky number again and enter here5

try again!

Guess the lucky number again and enter here4

Sorry but that was not very successful

**TASK THREE: DATA STRUCTURES**

1. Create a list of the 10 elements of four differernt types of Data Type like int, string, complex and float.

L =[2,5,7,6.8,"consultadd", 4.09, 7+1j,8,"training",4]

2. Create a list of size 5 and execute the slicing structure

Code:

fruits = ['orange', 'apple', 'pear', 'banana', 'kiwi']

fruits[:3]

O/P

['orange', 'apple', 'pear']

**If the question means that we need to print each list element along with it’s index:**

Code:

fruits = ['orange', 'apple', 'pear', 'banana', 'kiwi']

for i in range(5):

print(i,fruits[i])

O/P

0 orange

1 apple

2 pear

3 banana

4 kiwi

3. Write a program to get the sum and multiply of all the items in a given list.

Code:

l = [2,5,7,9,10]

sum = 0

product = 1

for i in range(len(l)):

sum += l[i]

product \*= l[i]

print(sum,product)

O/P:

33 6300

4. Find the largest and smallest number from a given list.

Code:

l = [2,5,7,9,10]

print("Largest no. in the given list is: ", min(l))

print("Smallest no. in the given list is: ", max(l))

O/P:

Largest no. in the given list is: 2

Smallest no. in the given list is: 10

5. Create a new list which contains the specified numbers after removing the even numbers from a predefined list.

Code:

l = [2,5,7,9,10]

newl =[]

for i in range(len(l)):

if l[i]%2!=0 :

newl.append(l[i])

else :

pass

print(newl)

O/P:

[5, 7, 9]

6. Create a list of first and last 5 elements where the values are square of numbers between 1 and30 (both included).

Code:

origl = list(range(1,31))

newl =[]

for i in range(5):

newl.append(origl[i]\*origl[i])

for i in range(len(origl)-5,len(origl)):

newl.append(origl[i]\*origl[i])

print(newl)

O/P:

[1, 4, 9, 16, 25, 676, 729, 784, 841, 900]

7. Write a program to replace the last element in a list with another list.

Sample data: [[1,3,5,7,9,10],[2,4,6,8]]

Expected output: [1,3,5,7,9,2,4,6,8]

Code:

l=[[1,3,5,7,9,10],[2,4,6,8]]

newlist=[]

for i in range(len(l)):

l[i].pop(-1)

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

newlist.append(l[i][j])

print(newlist)

O/P:

[1, 3, 5, 7, 9, 2, 4, 6]

8. Create a new dictionary by concatenating the following two dictionaries:

a={1:10,2:20}

b={3:30,4:40}

Expected Result: {1:10,2:20,3:30,4:40}

Code:

a = {1:10,2:20}

b = {3:30,4:40}

a.update(b)

print(a)

O/P:

{1: 10, 2: 20, 3: 30, 4: 40}

9. Create a dictionary that contains a number (between 1 and n) in the form(x,x\*x).

Sample data (n=5)

Expected Output: {1:1,2:4,3:9,4:16,5:25}

Code:

outputdict = {}

n = 7

for i in range(1,n+1):

outputdict[i] = i\*i

print(outputdict)

O/P:

{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49}

10. Write a program which accepts a sequence of comma-separated numbers from console and generate a list and a tuple which contains every number. Suppose the following input is supplied to the program:

34,67,55,33,12,98

The output should be:

[‘34’,’67’,’55’,’33’,’12’,’98’]

(‘34’,’67’,’55’,’33’,’12’,’98’)

Code:

inputdata = input("enter comma seperated sequence to be converted to list an tuple")

outputlist = inputdata.split(',')

outputtuple = tuple(outputlist)

print(outputlist, outputtuple)

O/P:

enter comma seperated sequence to be converted to list an tuple7,8,9,10,20,40

['7', '8', '9', '10', '20', '40'] ('7', '8', '9', '10', '20', '40')

**MORE QUESTIONS ON DATA STRUCTURES**

1. Create a list of the 10 elements of four different types of Data Type like int, string, complex and float.

repeated

2. Create a list of size 5 and execute the slicing structure

repeated

3. Create a list of given structure and run

**x=[100,200,300,400,500,[1,2,3,4,5,[10,20,30,40,50],6,7,8,9],600,700,800]**

* Access list [1, 2, 3, 4]

Code:

x1 = []

for i in range(5):

x1.append(x[5][i])

print(x1)

O/P:

[1, 2, 3, 4, 5]

* Access list [600, 700]

Code:

x=[100,200,300,400,500,[1,2,3,4,5,[10,20,30,40,50],6,7,8,9],600,700,800]

x1 = []

for i in range(6,8):

x1.append(x[i])

print(x1)

O/P:

[600, 700]

* Access list [100, 300, 500, 600, 800]

Code:

x=[100,200,300,400,500,[1,2,3,4,5,[10,20,30,40,50],6,7,8,9],600,700,800]

x1 = []

for i in range(0,9,2):

x1.append(x[i])

print(x1)

O/p:

[100, 300, 500, 600, 800]

* Access list [[800, 700, 600, [1, 2, 3, 4, 5, [10, 20, 30, 40, 50], 6, 7, 8, 9], 500, 400, 300, 200, 100]]

Code:

x=[100,200,300,400,500,[1,2,3,4,5,[10,20,30,40,50],6,7,8,9],600,700,800]

x1 = []

for i in range(1,9):

x1.append(x[-i])

print(x1)

O/P:

[800, 700, 600, [1, 2, 3, 4, 5, [10, 20, 30, 40, 50], 6, 7, 8, 9], 500, 400, 300, 200]

* Access list [10]

Code:

x=[100,200,300,400,500,[1,2,3,4,5,[10,20,30,40,50],6,7,8,9],600,700,800]

x1 = []

x1.append(x[5][5][0])

print(x1)

O/P:

[10]

* Access list [ ]

Code:

x=[100,200,300,400,500,[1,2,3,4,5,[10,20,30,40,50],6,7,8,9],600,700,800]

x.clear()

print(x)

O/P:

[]

4. Create a list of thousand number using range and xrange and see the difference between each other.

Code:

x = list(range(1,1000))

print(x)

y = list(xrange(1,1000))

print(y)

Differences(python 2):

* Type of range is list, whereas type of xrange is xrange object.
* Range occupies more memory than xrange as the return type is object.
* Xrange execution is faster.
* List operation can be used on range but not on xrange.

5. How Tuple is beneficial as compare to the list?

* Lists are mutable while tuple is immutable, thus can be used in case we need a static datastructure to store our sequence.
* List is good for performing operations whereas tuple is good for accessing elements
* Tuple use less memory as compared to list.
* Tuple’s iteration execution is faster as compared to list.

6. Write a program in Python to iterate through the list of numbers in the range of 1,100 and print the number which is divisible by 3 and a multiple of 2.

Code:

outputlist =[]

for i in range(1,100):

if i%2==0 and i%3==0 :

outputlist.append(i)

else:

pass

print(outputlist)

O/p:

[6, 12, 18, 24, 30, 36, 42, 48, 54, 60, 66, 72, 78, 84, 90, 96]

7. Write a program in Python to reverse a string and print only the vowel alphabet if exist in the string with their index.

Code:

origstring = "consultadd training"

revstring = origstring[::-1]

for i in range(len(revstring)):

if revstring[i] =='a'or revstring[i] == 'e' or revstring[i] == 'i'or revstring[i] == 'o'or revstring[i] == 'u':

print(i,revstring[i])

else:

pass

O/P:

2 i

4 i

5 a

11 a

14 u

17 o

8. Write a program in Python to iterate through the string “hello my name is abcde” and print the string which has even length of word.

Code:

origstring = "hello my name is abcde"

origlist = origstring.split()

outputlist =[]

outputstring =" "

for i in range(len(origlist)) :

if len(origlist[i])%2 == 0:

outputlist.append(origlist[i])

else:

pass

outputstring = outputstring.join(outputlist)

print(outputstring)

O/P:

my name is

9. Write a program in python to print the pair of numbers whose sum is equal to result number that is let's say 8.

**x=[1,2,3,4,5,6,7,8,9,-1]**

Code:

resnumber = 5

x=[1,2,3,4,5,6,7,8,9,-1]

for i in x:

for j in x:

if i+j == resnumber:

print(i,j)

else:

pass

O/P:

1 4

2 3

3 2

4 1

6 -1

-1 6

10. Write a program in Python to complete the following task:

* Create two different list as in even\_list and odd\_list
* Ask user to enter the number in the range of 1,50 and make sure if the entered number is even append it to the even\_list and if the entered number is odd append it to the odd list.
* Keep that in mind you can only add 5 items in each list
* Make sure once you entered the total 5 element calculate the sum of the list and return the maximum out of the list.

Code:

oddlist=[]

evenlist=[]

while True:

x =int(input("Enter any number from 1 to 50"))

if x%2==0:

if len(evenlist) != 5:

evenlist.append(x)

else:

if len(oddlist) != 5:

oddlist.append(x)

if len(evenlist) == 5 and len(oddlist) == 5:

break

print(oddlist,evenlist)

sumodd = sum(oddlist)

sumeven = sum(evenlist)

print(sumodd,sumeven)

print(max(oddlist),max(evenlist))

O/p:

Enter any number from 1 to 501

Enter any number from 1 to 501

Enter any number from 1 to 503

Enter any number from 1 to 507

Enter any number from 1 to 509

Enter any number from 1 to 505

Enter any number from 1 to 507

Enter any number from 1 to 508

Enter any number from 1 to 502

Enter any number from 1 to 506

Enter any number from 1 to 504

Enter any number from 1 to 502

[1, 1, 3, 7, 9] [8, 2, 6, 4, 2]

21 22

9 8

​11. Write a program to find out the occurrence of a specific word from an alphanumeric statement. **Example:** 12abcbacbaba344ab

**Output:** a=5 b=5 c=2 make sure you should avoid the numbers in you logic

Code:

x = "consultadd training1231"

newx =""

countlist = []

for i in range(len(x)):

if x[i] not in newx:

if x[i].isalpha():

count=1

for j in range(i+1,len(x)):

if x[j].isalpha():

if x[i]==x[j]:

count += 1

newx = newx + x[i]

countlist.append(count)

for i in range(len(newx)):

print(newx[i],"occured",countlist[i],"times" )

O/P:

c occured 1 times

o occured 1 times

n occured 3 times

s occured 1 times

u occured 1 times

l occured 1 times

t occured 2 times

a occured 2 times

d occured 2 times

r occured 1 times

i occured 2 times

g occured 1 times

12. Generate and print another tuple whose values are even numbers in the given tuple (1,2,3,4,5,6,7,8,9,10).

Code:

x=(1,2,3,4,5,6,7,8,9,10)

x1 = list(x)

newx =[]

for i in x:

if i%2==0:

newx.append(i)

else:

pass

print(tuple(newx))

O/P:

(2, 4, 6, 8, 10)

**TASK FOUR: FUNCTIONS**

1. Write a program to reverse a string.

Sample data: “1234abcd”

Expected Output: “dcba4321”

Code:

x = "1234abcd"

print(x[::-1])

O/P:

dcba4321

2. Write a function that accepts a string and calculate the number of uppercase letters and lowercase letters.

Expected Output:

No. of Upper case characters : 3

No. of Lower case Characters : 12

Code:

def countcase(inputstring):

countupper = 0

countlower = 0

for i in inputstring:

if i.isupper():

countupper += 1

elif i.islower():

countlower += 1

else:

pass

print("No. of Upper case characters : ", countupper)

print("No. of Lower case characters : ", countlower)

countcase("ConsultADD Training1234")

O/P:

No. of Upper case characters : 5

No. of Lower case characters : 13

3. Create a function that takes a list and returns a new list with unique elements of the first list.

Code:

def uniquelements(l):

newl = []

for i in l:

if i not in newl:

newl.append(i)

print(newl)

uniquelements([2,7,8,3,4,2,5,8,10])

O/P:

[2, 7, 8, 3, 4, 5, 10]

4. Write a program that accepts a hyphen-separated sequence of words as input and prints the words in a hyphen-separated sequence after sorting them alphabetically.

Code:

inputseq = input("Enter a hyphen-separated sequence of words: ")

inputlist = inputseq.split("-")

outputlist =[]

outputlist1 =[]

word = ""

outputseq = "-"

for i in inputlist:

outputlist.append(sorted(i))

for j in outputlist:

outputlist1.append(word.join(j))

outputseq = outputseq.join(outputlist1)

print(outputseq)

O/P:

Enter a hyphen-separated sequence of words: consult-add-training-april-2020-batch

clnostu-add-agiinnrt-ailpr-0022-abcht

5. Write a program that accepts a sequence of lines as input and prints the lines after making all characters in the sentence capitalized.

Sample input:

Hello world

Practice makes perfect

Expected Output:

HELLO WORLD

PRACTICE MAKES PERFECT

Code:

inputlist =[]

while True:

inputline = input("Enter the input line. Leave the input blank if you have no more lines to enter")

inputlist.append(inputline.upper())

if not inputline:

break

print("\n".join(inputlist))

O/P:

Enter the input line. Leave the input blank if you have no more lines to enterHello World

Enter the input line. Leave the input blank if you have no more lines to enterPractice makes you perfect

Enter the input line. Leave the input blank if you have no more lines to enterPractice is not a task it's a habit

Enter the input line. Leave the input blank if you have no more lines to enter

HELLO WORLD

PRACTICE MAKES YOU PERFECT

PRACTICE IS NOT A TASK IT'S A HABIT

6. Define a function that can receive two integral numbers in string form and compute their sum and print it in console.

Code:

def sum(n1,n2):

s = int(n1) + int(n2)

print(s)

sum('3','6')

O/P:

9

7. Define a function that can accept two strings as input and print the string with maximum length in console. If two strings have the same length, then the function should print all strings line by line.

Code:

def strprnt(str1,str2):

if len(str1) != len(str2):

print(str1,len(str1))

print(str2,len(str2))

else:

print(str1)

print(str2)

strprnt("PRACTICE MAKES YOU PERFECT","PRACTICE MAKES YOU PERFECT")

strprnt("HELLO WORLD","PRACTICE MAKES YOU PERFECT")

O/P:

PRACTICE MAKES YOU PERFECT

PRACTICE MAKES YOU PERFECT

HELLO WORLD 11

PRACTICE MAKES YOU PERFECT 26

8. Define a function which can generate and print a tuple where the value are square of numbers between 1 and 20.

Code:

def square():

l=[]

for i in range(1,21):

l.append(i\*i)

print(tuple(l))

square()

O/P:

(1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400)

9. Write a function called showNumbers that takes a parameter called limit. It should print all the numbers between 0 and limit with a label to identify the even and odd numbers.

Example: If the limit is 3 , it should print:

0 EVEN

1 ODD

2 EVEN

3 ODD

Code:

def showNumbers(limit):

for i in range(limit+1):

if i%2==0:

print(i, "EVEN")

else:

print(i, "ODD")

showNumbers(7)

O/P:

0 EVEN

1 ODD

2 EVEN

3 ODD

4 EVEN

5 ODD

6 EVEN

7 ODD

10. Write a program which can filter() to make a list whose elements are even number between 1 and 20 ( both included)

Code:

var = filter(lambda x:x%2== 0, range(1,21))

print(list(var))

O/P:

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]

11. Write a program which can map() and filter() to make a list whose elements are square of even number in [1,2,3,4,5,6,7,8,9,10]

Hints: Use map() to generate a list.

Use filter() to filter elements of a list

Use lambda to define anonymous functions

Code:

var = filter(lambda x:x%2== 0, [1,2,3,4,5,6,7,8,9,10])

square =map(lambda x: x\*x,list(var))

print(list(square))

O/P:

[4, 16, 36, 64, 100]

12. Write a function to compute 5/0 and use try/except to catch the exceptions

Code:

def divbyzero():

res = 5/0

print(res)

try:

divbyzero()

except:

print("division by zero is not a valid operation")

O/p:

division by zero is not a valid operation

13. Flatten the list [[1,2,3],[4,5],[6,7,8]] into [1,2,3,4,5,6,7,8] using reduce

Goal : Turn [1,2,3,4,5,6,7] to 1234567

Code:

from functools import reduce

red= reduce(lambda x,y:x+y,[[1,2,3],[4,5],[6,7,8]])

outputstr =""

for i in red:

outputstr += str(i)

print(outputstr)

O/P:

12345678

14. What is the output of the following codes:

(i) def foo():

try:

return 1

finally:

return 2

k = foo()

print(k)

O/P:

2

(ii) def a():

try:

f(x, 4)

finally:

print('after f')

print('after f?')

a()

O/P:

after f

Nameerror: name 'f' is not defined

**TASK FIVE: FILE HANDLING AND EXCEPTION HANDLING**

1. Write a program in Python to allow the error of syntax to go in exception. HINT: use SyntaxError

Python interpreter parses the file before execution. So, syntax errors can be caught only if you are parsing anything i.e. if the syntax error arises using eval, import or exec.

Example code:

try:

eval('x === x')

except SyntaxError:

print ("let syntax error go")

O/P:

let syntax error go

2. Write a program in Python to allow user to open a file by using argv module. If the entered name is incorrect throw an exception and ask them to enter the name again. Make sure to use read only mode.

Code:

from sys import argv

nameofprogram, filename = argv

print("name of program is:", nameofprogram)

print("name of file is:", filename)

while True:

try:

file = open(filename, "r")

content = file.read()

print(content)

file.close()

break

except:

filename =input("Please enter the correct file name again")

O/P:

PS C:\Users\Person\Desktop\work> python argv.py dat.txt

name of program is: argv.py

name of file is: dat.txt

Please enter the correct file name againdata.txt

Hello world. Ruchi Goel

Congratulations!!

Congratulations!!

Congratulations!!

3. Write a program to handle an error if the user entered the number more than four digits it should return “Please length is too short/long !!! Please provide only four digits”

Code: **In case we consider string input too from user**

while True:

try:

inputnumber = int(input("Please enter a 4 digit number"))

except:

print("It is not a valid 4 digit number")

continue

if inputnumber not in range(1000,10000):

if inputnumber < 1000:

print("Please length is too short!!! Please provide only four digits")

else:

print("Please length is too long!!! Please provide only four digits")

else:

break

print("The 4 digit input is: ",inputnumber)

O/P:

Please enter a 4 digit number-7

Please length is too short!!! Please provide only four digits

Please enter a 4 digit numbergyug

It is not a valid 4 digit number

Please enter a 4 digit number45678989

Please length is too long!!! Please provide only four digits

Please enter a 4 digit number4567

The 4 digit input is: 4567

Code: **In case we just consider only int input from user**

while True:

try:

inputnumber = int(input("Please enter a 4 digit number"))

if inputnumber not in range(1000,10000):

raise ValueError("It is not a valid 4 digit number")

else:

print("The 4 digit input is: ",inputnumber)

break

except ValueError:

print("Please length is too long/short!!! Please provide only four digits")

O/P:

Please enter a 4 digit number3567769879809

Please length is too long/short!!! Please provide only four digits

Please enter a 4 digit number234

Please length is too long/short!!! Please provide only four digits

Please enter a 4 digit number1234

The 4 digit input is: 1234

4. Create a login page backend to ask user to enter the UserEmail and password. Make sure to ask Re-Type Password and if the password is incorrect give chance to enter it again but it should not be more than 3 times.

Code:

import re

regex = '^\w+([\.-]?\w+)\*@\w+([\.-]?\w+)\*(\.\w{2,3})+$'

while True:

email = input("Enter your email")

if(re.search(regex,email)):

pwd= input("Enter the password")

repwd=input("Please re-enter the password")

for i in range(0,3):

if pwd==repwd:

print("Success!")

break

else:

repwd=input("Passwords don't match. Please enter correct password.Max 3 chances")

break

else:

print("invalid email")

O/P:

Enter your emailhjygjhgkj

invalid email

Enter your emailfygfygu

invalid email

Enter your emailabc@fuiei.com

Enter the passwordqwerty

Please re-enter the passwordtfgyguihy

Passwords don't match. Please enter correct password.Max 2 chancesqwerty

Success!

5. <https://www.programiz.com/python-programming/exception-handling> Go through this link to understand Finally and Raise concept.

Read

6. Read any file using Python File handling concept and return only the even length string from the doc.txt file.  
Consider the content as:

Hello I am a file

Where you need to return the data string

Which is of even length

Make sure you return the content in

The same link as it is present.

Code:

file = open("doc.txt")

content = file.read()

file.close()

contentlist = content.split("\n")

newcontentlist =[]

for each in contentlist:

if len(each)%2==0:

newcontentlist.append(each)

else:

pass

print("\n".join(newcontentlist))

O/P:

Hello I am a file

Which is of even length

Make sure you return the content in