Question 1 Consider the following Python module: a = 0 def b(): global a a = c(a) def c(a): return a + 2 After importing the module into the interpreter, you execute:

> > > b() b() b() a ? What value is displayed when the last expression (a) is evaluated? Explain your answer by indicating what happens in every executed statement.

```
a = 10

def b():
  global a
  a = c(a)

def c(a):
  return a + 2


b()
b()
b()
a
```

```
     16
```

Before we first time call funtion b(), the orginal value of a is 10 and also in the function b() variable is defined as global so the value of a in function c(a) will be remain same as it is updated previously.

But after first time calling b(), function b() will call function c(a) and it will change the value of a from 10 to 12 as function c(a) increases the value a by 2.

Again when the function b() is called second time, same process will happen and the value of a will change from 12 to 14 by function c(a).

At the last and third time we call function b(), same process happens and the value of a will change from 14 to 16 by function c(a).

At the end, when we display the value of a, it shows 16.

Question 2 Function fileLength(), given to you, takes the name of a file as input and returns the length of the file:

> > > fileLength('midterm.py') 284 fileLength('idterm.py') Traceback (most recent call last): File "<pyshell#34>",

> > > line 1, in fileLength('idterm.py') File "/Users/me/midterm.py", line 3, in fileLength infile = open(filename) FileNotFoundError: [Errno 2] No such file or directory: 'idterm.py' As shown above, if the file cannot be found by the interpreter or if it cannot be read as a text file, an exception will be raised. Modify function fileLength() so that a friendly message is printed instead: fileLength('midterm.py') 358 fileLength('idterm.py') File idterm.py not found.

```
#Defining function file_length
def file_length(file_name):
    try:
        file = open(file_name)
        contents = file.read()
        file.close()
        print(len(contents))
    except:
        print("File "+file_name+" not found.")


file_length('/content/sample_data/Filelength.txt')
```

```
 123
```

Question 3 Write a class named Marsupial that can be used as shown below:

> > > m = Marsupial() m.put_in_pouch('doll') m.put_in_pouch('firetruck') m.put_in_pouch('kitten') m.pouch_contents() ['doll', 'firetruck', 'kitten']

Now write a class named Kangaroo as a subclass of Marsupial that inherits all the attributes of Marsupial and also: a. extends the Marsupial **init** constructor to take, as input, the coordinates x and y of the Kangaroo object, b. supports method jump that takes number values dx and dy as input and moves the kangaroo by dx units along the x-axis and by dy units along the yaxis, and c. overloads the **str** operator so it behaves as shown below.

> > > k = Kangaroo(0,0) print(k) I am a Kangaroo located at coordinates (0,0) k.put_in_pouch('doll') k.put_in_pouch('firetruck') k.put_in_pouch('kitten') k.pouch_contents() ['doll', 'firetruck', 'kitten'] k.jump(1,0) k.jump(1,0) k.jump(1,0) print(k) I am a Kangaroo located at coordinates (3,0)

```
#Defining Parent class Marsupial
class Marsupial():

  list = []
  def __init__(self):
    self = []

  #Defining function put strings in pouch
  def put_in_pouch(self, str):
    if str not in self.list:
      self.list.append(str)

  #Defining function to print list
  def pouch_contents(self):
    print(self.list)


m = Marsupial()
m.put_in_pouch('doll')
m.put_in_pouch('firetruck')
m.put_in_pouch('kitten')
m.pouch_contents()
```

```
['doll', 'firetruck', 'kitten']
```

```
#Defining child class Kangaroo which inherits Parent class Marsupial
class  Kangaroo(Marsupial):
    def __init__(self, x1=0, y1=0):
        self.x= x1
        self.y= y1

    #Defining sunction to set value of x and y
    def setxy(self,x1,y1):
        self.x = x1
        self.y= y1

    #Defining function display yhe value of x and y
    def get(self):
        return(self.x, self.y)

    #Defining function jump
    def jump(self, dx , dy):
        self.x=self.x+dx
        self.y=self.y+dy

    #Defining function which shows current position of Kangaroo
    def __str__(self):
        return(f'I am a kangaroo located at coordinates {self.x,self.y}')
```

```
k = Kangaroo(0,0)
print(k)
```

```
     I am a kangaroo located at coordinates (0, 0)
```

```
k.put_in_pouch('doll')
k.put_in_pouch('firetruck')
k.put_in_pouch('kitten')
k.pouch_contents()
```

```
     ['doll', 'firetruck', 'kitten']
```

```
k.jump(1,0)
k.jump(1,0)
k.jump(1,0)
print(k)
```

```
     I am a kangaroo located at coordinates (3, 0)
```

Question 4 Write function collatz() that takes a positive integer x as input and prints the Collatz sequence starting at x. A Collatz sequence is obtained by repeatedly applying this rule to the previous number x in the sequence: x = { $x/2$ $if\ x\ is\ even$ $3x$ + 1 $if\ x\ is\ odd$ Your function should stop when the sequence gets to number 1. Your implementation must be recursive, without any loops.

> collatz(1) 1 collatz(10) 10 5 16 8 4 2 1

```
#Defining function collatz
def collatz(number):

    if number == 1:
      return

    elif number % 2 == 0:
        print(number // 2)
        number = number // 2
        collatz(number)

    elif number % 2 == 1:
        result = 3 * number + 1
        print(result)
        number = result
        collatz(number)


#Getting proper input from user
while(1):
```

```
  try:
    n = int(input("Give me a number :"))
    if(n < 0):
      print("Please enter positive integer number....!!!")
    else:
      break
  except:
    print("Please enter positive integer number....!!!")

#Calling collatz function
collatz(n)
```

```
 Give me a number :a
 Please enter positive integer number....!!!
 Give me a number :-4
 Please enter positive integer number....!!!
 Give me a number :5
 16
 8
 4
 2
 1
```

Question 5 Write a recursive method binary() that takes a non-negative integer n and prints the binary representation of integer n.

> binary(0) 0 binary(1) 1 binary(3) 11 binary(9) 1001

```
#Defining function convertToBinary which converts decimal to binary
def convertToBinary(n):
   if n > 1:
       convertToBinary(n//2)
   print(n % 2,end = '')


from copy import Error
from logging import exception

#Getting proper value from user
while(1):
  try:
    dec = int(input("Enter a positive integer number to convert it in binary :"))
    if(dec < 0):
      print("Please enter positive integer number....!!!")
    else:
      break
  except:
    print("Please enter positive integer number....!!!")
```

```
#Calling coonvertToBinary function
convertToBinary(dec)
```

```
    Enter a positive integer number to convert it in binary :a
    Please enter positive integer number....!!!
    Enter a positive integer number to convert it in binary :-3
    Please enter positive integer number....!!!
    Enter a positive integer number to convert it in binary :6
    110
```

Question 6 Implement a class named HeadingParser that can be used to parse an HTML document, and retrieve and print all the headings in the document. You should implement your class as a subclass of HTMLParser, defined in Standard Library module html.parser. When fed a string containing HTML code, your class should print the headings, one per line and in the order in which they appear in the document. Each heading should be indented as follows: an h1 heading should have indentation 0, and h2 heading should have indentation 1, etc. Test your implementation using w3c.html.

> > > infile = open('w3c.html') content = infile.read()
> > > infile.close() hp = HeadingParser() hp.feed(content)
> > > W3C Mission Principles

```
from html.parser import HTMLParser

#Defining class HeadingParser which inherits Parent class HTMLParser
class HeadingParser(HTMLParser):
  h1 = False
  h2 = False
  h3 = False
  h4 = False
  h5 = False
  h6 = False

  #Defining function which handles start tags
  def handle_starttag(self, tag, attrs):
      if tag == 'h1':
        self.h1 = True
      elif tag == 'h2':
        self.h2 = True
      elif tag == 'h3':
        self.h3 = True
      elif tag == 'h4':
        self.h4 = True
      elif tag == 'h5':
        self.h5 = True
      elif tag == 'h6':
        self.h6 = True
```

```
    #Defining function which handles data between start and end tag
    def handle_data(self, data):
        if self.h1:
          print(data)
        elif self.h2:
          print('\t',data)
        elif self.h3:
          print('\t\t',data)
        elif self.h4:
          print('\t\t\t',data)
        elif self.h5:
          print('\t\t\t\t',data)
        elif self.h6:
          print('\t\t\t\t\t',data)

    #Defining function which handles end tags
    def handle_endtag(self, tag):
        self.h1= False
        self.h2= False
        self.h3= False
        self.h4= False
        self.h5= False
        self.h6= False


#Opening and getting data of html file
infile = open('/content/sample_data/w3c.txt')
content = infile.read()
infile.close()

#Making object of HeadingParser class and calling function
hp = HeadingParser()
hp.feed(content)

     W3C Mission
              Principles
```

Question 7 Implement recursive function webdir() that takes as input: a URL (as a string) and non-negative integers depth and indent. Your function should visit every web page reachable from the starting URL web page in depth clicks or less, and print each web page's URL. As shown below, indentation, specified by indent, should be used to indicate the depth of a URL.

> webdir('http://reed.cs.depaul.edu/lperkovic/csc242/test1.html' , 2, 0)
> http://reed.cs.depaul.edu/lperkovic/csc242/test1.html
> http://reed.cs.depaul.edu/lperkovic/csc242/test2.html
> http://reed.cs.depaul.edu/lperkovic/csc242/test4.html

> > http://reed.cs.depaul.edu/lperkovic/csc242/test3.html

```
from urllib.request import urlopen
from urllib.parse import urljoin
from html.parser import HTMLParser

#Defining Collector class which inherits HTMLParser class
class Collector(HTMLParser):

    def __init__(self, url):
        HTMLParser.__init__(self)
        self.url = url
        self.links = []

    #Defining function which handles start tags
    def handle_starttag(self, tag, attrs):
        if tag == 'a':
            for attr in attrs:
                if attr[0] == 'href':
                    # construct absolute URL
                    absolute = urljoin(self.url, attr[1])
                    if absolute[:4] == 'http':
                        self.links.append(absolute)

    #Defining function which handles links in data
    def getLinks(self):
        return self.links

#Defining function webdir
def webdir(url, depth, indent):

    content = urlopen(url).read().decode()
    collector = Collector(url)
    collector.feed(content)
    urllist = collector.getLinks()
    d= depth
    i = indent

    print("\t"*indent+"{}".format(url))
    for url in urllist:
        if (d>0):
            webdir(url,d,i+1)
        else:
            return
        d = d-1

#Calling function webdir for given link but it shows error
webdir('http://reed.cs.depaul.edu/lperkovic/csc242/test1.html' , 2, 0)
```

```
---------------------------------------------------------------------------
HTTPError                                 Traceback (most recent call last)
<ipython-input-44-5cbcf3a17141> in <module>
     44
     45 #Calling function webdir for given link but it shows error
---> 46 webdir('http://reed.cs.depaul.edu/lperkovic/csc242/test1.html' , 2, 0)

 6 frames
/usr/lib/python3.7/urllib/request.py in http_error_default(self, req, fp, code, msg,
hdrs)
    647 class HTTPDefaultErrorHandler(BaseHandler):
    648     def http_error_default(self, req, fp, code, msg, hdrs):
--> 649         raise HTTPError(req.full_url, code, msg, hdrs, fp)
    650
    651 class HTTPRedirectHandler(BaseHandler):

HTTPError: HTTP Error 404:
```

```
#Calling function webdir for applicable link
webdir('http://reed.cs.depaul.edu/lperkovic/test1.html' , 2, 0)
```

```
http://reed.cs.depaul.edu/lperkovic/test1.html
        http://reed.cs.depaul.edu/lperkovic/test2.html
                http://reed.cs.depaul.edu/lperkovic/test4.html
        http://reed.cs.depaul.edu/lperkovic/test3.html
                http://reed.cs.depaul.edu/lperkovic/test4.html
```

Question 8 Write SQL queries on the below database table that return: a) All the temperature data. b) All the cities, but without repetition. c) All the records for India. d) All the Fall records. e) The city, country, and season for which the average rainfall is between 200 and 400 millimeters. f) The city and country for which the average Fall temperature is above 20 degrees, in increasing temperature order. g) The total annual rainfall for Cairo. h) The total rainfall for each season. City Country Season Temperature (C) Rainfall (mm) Mumbai India Winter 24.8 5.9 Mumbai India Spring 28.4 16.2 Mumbai India Summer 27.9 1549.4 Mumbai India Fall 27.6 346.0 London United Kingdom Winter 4.2 207.7 London United Kingdom Spring 8.3 169.6 London United Kingdom Summer 15.7 157.0 London United Kingdom Fall 10.4 218.5 Cairo Egypt Winter 13.6 16.5 Cairo Egypt Spring 20.7 6.5 Cairo Egypt Summer 27.7 0.1 Cairo Egypt Fall 22.2 4.5

```
import sqlite3
content = sqlite3.connect('weather_city_data.db')
cursor = content.cursor()

#Creating table and inserting data into it
cursor.execute("CREATE TABLE Weatdata (City text, Country text, Season text, Temp_in_c double
cursor.execute("INSERT INTO Weatdata VALUES ('Mumbai','India','Winter',24.8,5.9),('Mumbai','I

content.commit()
```

```
import sqlite3
connection = sqlite3.connect('databaseName.sqlite')


#All the weather data.
cursor.execute('SELECT * FROM Weatdata')
print(cursor.fetchall())
```

```
[('Mumbai', 'India', 'Winter', 24.8, 5.9), ('Mumbai', 'India', 'Spring', 28.4, 16.2), (
```

```
#All the temperature data.
cursor.execute('SELECT Temp_in_c FROM Weatdata')
print(cursor.fetchall())
```

```
[(24.8,), (28.4,), (27.9,), (27.6,), (4.2,), (8.3,), (15.7,), (10.4,), (13.6,), (20.7,)
```

```
#All the cities, but without repetition.
cursor.execute('SELECT DISTINCT City FROM Weatdata ')
print(cursor.fetchall())
```

```
[('Mumbai',), ('London',), ('Cairo',)]
```

```
#All the records for India.
cursor.execute('SELECT * FROM Weatdata WHERE Country="India"')
print(cursor.fetchall())
```

```
[('Mumbai', 'India', 'Winter', 24.8, 5.9), ('Mumbai', 'India', 'Spring', 28.4, 16.2), (
```

```
#All the Fall records
cursor.execute('SELECT * FROM Weatdata WHERE Season="Fall"')
print(cursor.fetchall())
```

```
[('Mumbai', 'India ', 'Fall', 27.6, 346.0), ('London', 'United Kingdom', 'Fall', 10.4,
```

```
#The city, country, and season for which the average rainfall is between 200 and 400 millimet
cursor.execute('SELECT City,Country,Season FROM Weatdata WHERE "rainfall_in_mm" BETWEEN 200 A
print(cursor.fetchall())
```

```
[('Mumbai', 'India ', 'Fall'), ('London', 'United Kingdom', 'Winter'), ('London', 'Unit
```

```
#The city and country for which the average Fall temperature is above 20 degrees, in increasi
```

```
cursor.execute('SELECT City,Country FROM Weatdata WHERE Season="Fall" AND Temp_in_c > 20 ORDE
print(cursor.fetchall())
```

```
[('Cairo', 'Egypt'), ('Mumbai', 'India ')]
```

```
#The total annual rainfall for Cairo.
cursor.execute('SELECT SUM(rainfall_in_mm) FROM Weatdata WHERE City="Cairo"')
print(cursor.fetchall())
```

```
[(27.6,)]
```

```
#The total rainfall for each season.
cursor.execute('SELECT Season,SUM(rainfall_in_mm) FROM Weatdata GROUP BY Season')
print(cursor.fetchall())
```

```
[('Fall', 569.0), ('Spring', 192.29999999999998), ('Summer', 1706.5), ('Winter', 230.1)
```

Question 9 . Suppose list words is defined as follows:

> words = ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog'] Write list comprehension expressions that use list words and generate the following lists: a) ['THE', 'QUICK', 'BROWN', 'FOX', 'JUMPS', 'OVER', 'THE', 'LAZY', 'DOG'] b) ['the', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog'] c) [3, 5, 5, 3, 5, 4, 3, 4, 3] (the list of lengths of words in list words). d) [['THE', 'the', 3], ['QUICK', 'quick', 5], ['BROWN', 'brown', 5], ['FOX', 'fox', 3], ['JUMPS', 'jumps', 5], ['OVER', 'over', 4], ['THE', 'the', 3], ['LAZY', 'lazy', 4], ['DOG', 'dog', 3]] (the list containing a list for every word of list words, where each list contains the word in uppercase and lowercase and the length of the word.) e) ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog'] (the list of words in list words containing 4 or more characters.)

```
# ['THE', 'QUICK', 'BROWN', 'FOX', 'JUMPS', 'OVER', 'THE', 'LAZY', 'DOG']
words = ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog']
```

```
#Converting all data into upper case
wordsu = [element.upper() for element in words]
print(wordsu)
```

```
['THE', 'QUICK', 'BROWN', 'FOX', 'JUMPS', 'OVER', 'THE', 'LAZY', 'DOG']
```

```
# ['the', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog']

#Converting all data into lower case
wordsl = [element.lower() for element in words]
print(wordsl)
```

```
['the', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog']
```

```
# [3, 5, 5, 3, 5, 4, 3, 4, 3] (the list of lengths of words in list words).
wordslen = [len(word) for word in words]
print(wordslen)
```

```
[3, 5, 5, 3, 5, 4, 3, 4, 3]
```

```
# [['THE', 'the', 3], ['QUICK', 'quick', 5], ['BROWN', 'brown', 5], ['FOX', 'fox', 3], ['JUMP
newwords = [[word.upper(),word.lower(),len(word)] for word in words]
print(newwords)
```

```
[['THE', 'the', 3], ['QUICK', 'quick', 5], ['BROWN', 'brown', 5], ['FOX', 'fox', 3], ['
```

```
# ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog'] (the list of words
twords=[word for word in words if len(word)>3]
print(twords)
```

```
['quick', 'brown', 'jumps', 'over', 'lazy']
```

Colab paid products - Cancel contracts here

✓ 0s completed at 18:26