Problem Set #2 (BDAT 1004)

By Daryoush Shabahang

July 8, 2021

Question 1 Python

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.

MY ANSWER:

In this scenario, a is defined as a global variable, then function b is making changes to it by calling function c to add 2 each time a is interpreted. For example, the value starts at 0, then after the first time a = 2, then the second time a = 4 and finally, the third time a = 6.
Therefore, the final value evaluated will be 6 since function b is called three times.

Question 2 Python

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 module>
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.
```

In [88]:

```
def fileLength(filename):
    # try to open the file
    try:
        file = open(filename)
        contents = file.read()
        print(len(contents))
        file.close()

    # otherwise, this text is printed if the file name is incorrect or file is not found
    except FileNotFoundError:
        print("File", filename, "not found")

fileLength("idterm.py")
```

```
File idterm.py not found
```

Question 3 Python

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 y-axis, 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)

In [89]:

```
class Marsupial:
    # initialize the class to be an empty list
    def __init__(self):
        self.q = []

    # append items to the same list
    def put_in_pouch(self, item):
        self.q.append(item)

    # return the contents of the same list
    def pouch_contents(self):
        return self.q

m = Marsupial()
m.put_in_pouch("doll")
m.put_in_pouch("firetruck")
m.put_in_pouch("kitten")
m.pouch_contents()

class Kangaroo(Marsupial):
    def __init__(self, x_coordinate, y_coordinate):
        Marsupial.__init__(self)
        self.x_coordinate = x_coordinate
        self.y_coordinate = y_coordinate

    def jump(self, dx, dy):
        self.x_coordinate = self.x_coordinate + dx
        self.y_coordinate = self.y_coordinate + dy

    def __repr__(self):
        return("I am a Kangaroo located at coordinates ({0},{1})".format(self.x_coordinate, self.

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

k.put_in_pouch("doll")
k.put_in_pouch("firetruck")
k.put_in_pouch("kitten")
k.pouch_contents()

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

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

Question 4 Python

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

In [90]:

```
def collatz(x):
    # this will check if x is a positive integer (i.e. greater than 1 or not)
    if(x) > 1:
        # this will round the number to ensure there aren't any decimals in the output
        print(round(x))
        # if x is even, then the number will be divided by 2
        # and result will be stored in x, then calling the collatz function again
        if x % 2 == 0:
            x /= 2
            collatz(x)

        # if x is odd, then the number will be multiplied by 3 + 1
        # and result will be stored in x, then calling the collatz function again
        else:
            x = x * 3 + 1
            collatz(x)

    # otherwise if x is 1 or less than 1, then the output is only 1.
    else:
        print(1)

collatz(10)
```

```
10
5
16
8
4
2
1
```

Question 5 Python

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

In [91]:

```
def binary(n):
    # if the value is less than 0, there will be an error
    try:
        if n < 0:
            raise ValueError(n)

        # otherwise, the output will print
        elif n > 1:
            binary(n // 2)
        print(n % 2,end = '')

    except ValueError:
        print("Only a non-negative number is permitted, please try again")

binary(9)
```

```
1001
```

Question 6 Python

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()
>>> binary(3)
>>> hp.feed(content)
W3C Mission
Principles
```

In [92]:

```
from html.parser import HTMLParser

class HeadingParser(HTMLParser):

    # initially an evaluation of each heading
    h1_check_val = False
    h2_check_val = False
    h3_check_val = False
    h4_check_val = False
    h5_check_val = False
    h6_check_val = False

    def handle_starttag(self, tag, attrs):
        if tag == "h1":
            self.h1_check_val = True
        if tag == "h2":
            self.h2_check_val = True
        if tag == "h3":
            self.h3_check_val = True
        if tag == "h4":
            self.h4_check_val = True
        if tag == "h5":
            self.h5_check_val = True
        if tag == "h6":
            self.h6_check_val = True

    # depending on the type of heading used, a tab with whitespace is added except for H1
    def handle_data(self, data):
        if(self.h1_check_val):
            print(data)
            self.h1_check_val = False

        if(self.h2_check_val):
            print("\t" * 2 + data)
            self.h2_check_val = False

        if(self.h3_check_val):
            print("\t" * 3 + data)
            self.h3_check_val = False

        if(self.h4_check_val):
            print("\t" * 4 + data)
            self.h4_check_val = False

        if(self.h5_check_val):
            print("\t" * 5 + data)
            self.h5_check_val = False

        if(self.h6_check_val):
            print("\t" * 6 + data)
            self.h6_check_val = False

infile = open("w3c.html")
content = infile.read()
infile.close()

headingParser = HeadingParser()
headingParser.feed(content)
```

```
W3C Mission
 Principles
```

Question 7 Python

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

In [93]:

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

class Collector(HTMLParser):

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

    def handle_starttag(self, tag, attrs):
        if tag == "a":
            for attr in attrs:
                if attr[0] == "href":
                    absolute = urljoin(self.url, attr[1])
                    if absolute[:4] == "http":
                        self.links.append(absolute)

    def getLinks(self):
        return self.links

visited = set()

def webdir(url, depth, indent):
    global visited
    visited.add(url)
    links = analyze(url)
    for link in links:
        if link not in visited:
            try:
                print('',link)
            except:
                print(link)

webdir("http://reed.cs.depaul.edu/lperkovic/one.html", 2, 0)
```

```
http://reed.cs.depaul.edu/lperkovic/two.html
http://reed.cs.depaul.edu/lperkovic/three.html
```

Question 8 SQL

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.

I am first creating the table and displaying it since the table doesn't yet exist. Then I will answer each question.
I learned how to connect SQL with Jupyter Notebook by using these two resources:

https://www.datacamp.com/community/tutorials/sql-interface-within-jupyterlab

https://towardsdatascience.com/how-to-run-sql-queries-from-a-jupyter-notebook-aaa18e59e7bc

In [4]:

```
%%sql
CREATE TABLE various_cities
(
    City        varchar(25),
    Country     varchar(25),
    Season      varchar(10),
    Temperature float,
    Rainfall    float
);

INSERT INTO various_cities
VALUES ('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);
```

```
 * sqlite://
Done.
12 rows affected.
```

```
---------------------------------------------------------------------------
ResourceClosedError                       Traceback (most recent call last)
<ipython-input-4-9f4b72d7b715> in <module>
----> 1 get_ipython().run_cell_magic('sql', '', "CREATE TABLE various_cities\n(\n    City        varchar(25),\n    Country     varchar(25),\n    Season      varchar(10),\n    Temperature float,\n    Rainfall    float\n);\n\nINSERT INTO various_cities\nVALUES ('Mumbai', 'India', 'Winter', 24.8, 5.9),\n       ('Mumbai', 'India', 'Spring', 28.4, 16.2),\n       ('Mumbai', 'India', 'Summer', 27.9, 1549.4),\n       ('Mumbai', 'India', 'Fall', 27.6, 346.0),\n       ('London', 'United Kingdom', 'Winter', 4.2, 207.7),\n       ('London', 'United Kingdom', 'Spring', 8.3, 169.6),\n       ('London', 'United Kingdom', 'Summer', 15.7, 157.0),\n       ('London', 'United Kingdom', 'Fall', 10.4, 218.5),\n       ('Cairo', 'Egypt', 'Winter', 13.6, 16.5),\n       ('Cairo', 'Egypt', 'Spring', 20.7, 6.5),\n       ('Cairo', 'Egypt', 'Summer', 27.7, 0.1),\n       ('Cairo', 'Egypt', 'Fall', 22.2, 4.5);\n")

~\anaconda3\lib\site-packages\IPython\core\interactiveshell.py in run_cell_magic(self, magic_name, line, cell)
   2397             with self.builtin_trap:
   2398                 args = (magic_arg_s, cell)
-> 2399                 result = fn(*args, **kwargs)
   2400             return result
   2401

~\anaconda3\lib\site-packages\decorator.py in fun(*args, **kw)
    229             if not kwsyntax:
    230                 args, kw = fix(args, kw, sig)
--> 231             return caller(func, *(extras + args), **kw)
    232     fun.__name__ = func.__name__
    233     fun.__doc__ = func.__doc__

~\anaconda3\lib\site-packages\IPython\core\magic.py in <lambda>(f, *a, **k)
    185     # but it's overkill for just that one bit of state.
    186     def magic_deco(arg):
--> 187         call = lambda f, *a, **k: f(*a, **k)
    188
    189         if callable(arg):

~\anaconda3\lib\site-packages\decorator.py in fun(*args, **kw)
    229             if not kwsyntax:
    230                 args, kw = fix(args, kw, sig)
--> 231             return caller(func, *(extras + args), **kw)
    232     fun.__name__ = func.__name__
    233     fun.__doc__ = func.__doc__

~\anaconda3\lib\site-packages\IPython\core\magic.py in <lambda>(f, *a, **k)
    185     # but it's overkill for just that one bit of state.
    186     def magic_deco(arg):
--> 187         call = lambda f, *a, **k: f(*a, **k)
    188
    189         if callable(arg):

~\anaconda3\lib\site-packages\sql\magic.py in execute(self, line, cell, local_ns)
    215
    216         try:
--> 217             result = sql.run.run(conn, parsed["sql"], self, user_ns)
    218
    219             if (
```

```
~\anaconda3\lib\site-packages\sql\run.py in run(conn, sql, config, user_namespace)
    369             if result and config.feedback:
    370                 print(interpret_rowcount(result.rowcount))
--> 371         resultset = ResultSet(result, statement, config)
    372         if config.autopandas:
    373             return resultset.DataFrame()

~\anaconda3\lib\site-packages\sql\run.py in __init__(self, sqlaproxy, sql, config)
    105
    106     def __init__(self, sqlaproxy, sql, config):
--> 107         self.keys = sqlaproxy.keys()
    108         self.sql = sql
    109         self.config = config

~\anaconda3\lib\site-packages\sqlalchemy\engine\result.py in keys(self)
    705
    706         """
--> 707         return self._metadata.keys
    708
    709

~\anaconda3\lib\site-packages\sqlalchemy\engine\cursor.py in keys(self)
   1199     @property
   1200     def keys(self):
-> 1201         self._we_dont_return_rows()
   1202
   1203

~\anaconda3\lib\site-packages\sqlalchemy\engine\cursor.py in _we_dont_return_rows(self, err)
   1176
   1177     def _we_dont_return_rows(self, err=None):
-> 1178         util.raise_(
   1179             exc.ResourceClosedError(
   1180                 "This result object does not return rows. "

~\anaconda3\lib\site-packages\sqlalchemy\util\compat.py in raise_(***failed resolving arguments***
*)
    209
    210         try:
--> 211             raise exception
    212         finally:
    213             # credit to

ResourceClosedError: This result object does not return rows. It has been closed automatically.
```

I am displaying the entire table prior to answering each question.

In [78]:
```
%%sql

SELECT *
FROM various_cities;
```

```
 * sqlite://
Done.
```

Out[78]:

City	Country	Season	Temperature	Rainfall
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

Question 8a SQL

a) All the temperature data.

In [79]:
```
%%sql

SELECT Temperature
FROM various_cities;
```

```
 * sqlite://
Done.
```

Out[79]:

Temperature
24.8
28.4
27.9
27.6
4.2
8.3
15.7
10.4
13.6
20.7
27.7
22.2

Question 8b SQL

b) All the cities, but without repetition.

In [80]:
```
%%sql

SELECT DISTINCT(city)
FROM various_cities;
```

```
 * sqlite://
Done.
```

Out[80]:

City
Mumbai
London
Cairo

Question 8c SQL

c) All the records for India.

In [81]:
```
%%sql

SELECT *
FROM various_cities
WHERE country = 'India';
```

```
 * sqlite://
Done.
```

Out[81]:

City	Country	Season	Temperature	Rainfall
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

Question 8d SQL

d) All the Fall records.

In [82]:
```
%%sql

SELECT *
FROM various_cities
WHERE season = 'Fall';
```

```
 * sqlite://
Done.
```

Out[82]:

City	Country	Season	Temperature	Rainfall
Mumbai	India	Fall	27.6	346.0
London	United Kingdom	Fall	10.4	218.5
Cairo	Egypt	Fall	22.2	4.5

Question 8e SQL

e) The city, country, and season for which the average rainfall is between 200 and 400 millimeters.

In [83]:
```
%%sql

SELECT city, country, season, rainfall
FROM various_cities
WHERE rainfall BETWEEN 200 AND 400;
```

```
 * sqlite://
Done.
```

Out[83]:

City	Country	Season	Rainfall
Mumbai	India	Fall	346.0
London	United Kingdom	Winter	207.7
London	United Kingdom	Fall	218.5

Question 8f SQL

f) The city and country for which the average Fall temperature is above 20 degrees, in increasing temperature order.

In [84]:
```
%%sql

SELECT city, country
FROM various_cities
WHERE season = 'Fall' AND temperature > 20
ORDER BY temperature ASC;
```

```
 * sqlite://
Done.
```

Out[84]:

City	Country
Cairo	Egypt
Mumbai	India

Question 8g SQL

g) The total annual rainfall for Cairo.

In [85]:
```
%%sql

SELECT city, sum(rainfall) AS Total_Annual_Rainfall
FROM various_cities
WHERE city = 'Cairo';
```

```
 * sqlite://
Done.
```

Out[85]:

City	Total_Annual_Rainfall
Cairo	27.6

Question 8h SQL

h) The total rainfall for each season.

In [86]:
```
%%sql

SELECT season, sum(round(rainfall)) AS Total_Rainfall
FROM various_cities
GROUP BY season;
```

```
 * sqlite://
Done.
```

Out[86]:

Season	Total_Rainfall
Fall	570.0
Spring	193.0
Summer	1706.0
Winter	231.0

Question 9 Python

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:

In [87]:

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

words = ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog']
listA = [x.upper() for x in words]
print(listA)

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

listB = [x.lower() for x in words]
print(listB)

# c) [3, 5, 5, 3, 5, 4, 3, 4, 3] (the list of lengths of words in list words).

listC = [len(x) for x in words]
print(listC)

# d) [['THE', 'the', 3], ['QUICK', 'quick', 5], ['BROWN', 'brown', 5], ['FOX', 'fox', 3], ['JUMPS
# ['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
# and the length of the word.)

listD = [[x.upper(), x.lower(), len(x)] for x in words]
print(listD)

# e) ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog']

# (the list of words in list words containing 4 or more characters.)

listE = [x for x in words if len(x) >= 4]
print(listE)

# Please note: in the PDF, it shows a lot more words, but the only words with 4 or more character
# output ("quick", "brown", "jumps", "over", "lazy")
```

```
['THE', 'QUICK', 'BROWN', 'FOX', 'JUMPS', 'OVER', 'THE', 'LAZY', 'DOG']
['the', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog']
[3, 5, 5, 3, 5, 4, 3, 4, 3]
[['THE', 'the', 3], ['QUICK', 'quick', 5], ['BROWN', 'brown', 5], ['FOX', 'fox', 3], ['JUMPS', 'j
umps', 5], ['OVER', 'over', 4], ['THE', 'the', 3], ['LAZY', 'lazy', 4], ['DOG', 'dog', 3]]
['quick', 'brown', 'jumps', 'over', 'lazy']
```

In [2]:

```
%load_ext sql
```

In [3]:

```
%sql sqlite://
```