HW1 Notebook

April 14, 2020

1 Stats 21: Homework 1

1.1 Yingyi Zhu

I've started the homework file for you. You'll need to fill in the rest with your answers. My encouragement is to use the keyboard shortcuts as much as possible and use the mouse as little as possible while working the Jupyter Notebook.

After you complete the homework with your answers, go to the menu and choose Kernel > Restart & Run All. Go through the document to **make sure all requested output is visible**. You will not get credit for problems where the requested output is not visible, even if the function you coded is correct.

When you are satisfied with the output, choose File > Download As … > PDF, or choose HTML. Submit the PDF or HTML file to CCLE.

2 Problem 1

(A) In a print statement, what happens if you leave out one of the parentheses, or both? What kind of error messages does Python produce for these mistakes?

Answer: If you leave out a parenthesis, a syntax error with a message of unexpected EOF while pasing occurs. If you leave out both parenthese, syntax error with a message of missing parentheses in call to 'print' indeed.

(B) If you are trying to print a string, what happens if you leave out one of the quotation marks, or both? What kind of error messages does Python produce for these mistakes?

Answer: - case1:print(hallo, world!), is a SyntaxError. - case2: print(hallo, world), is a NameError with message name 'hallo' is not defined - case3: print("hallo, world) is a SyntaxError with message EOL while scanning string literal.

(C) Spaces do not make differences in Python.Operations are applied under the order of precedence. Along all of the calculations, only 3-*-2 has a SyntaxError: invalid syntax.

(D) In math notation, leading zeros are ok, as in 09. What happens if you try this in Python? What about 011? Search the internet for information about Python and leading zeros and write a sentence summarizing your findings.

Answer: In Python3, leading zeros are not recognized during tokenizaton as Python2 will do. Leading Zeros mean octal numbers in Python2, but such numbers have changed to a different representation in Python3.

3 Problem 2

(A) How many seconds are there in 42 minutes 42 seconds?

[25]:
```
sec = 42*60+42
print('There are %s seconds in total.'%sec)
```

```
There are 2562 seconds in total.
```

(B) There are 1.61 kilometers in a mile. How many miles are there in 10 kilometers?

[49]:
```
miles = 10/1.61
print('There are about %.3f miles in total.'%miles)
```

```
There are about 6.211 miles in total.
```

(C) If you run a 10 kilometer race in 42 minutes 42 seconds, what is your average 1-mile pace (time to complete 1 mile in minutes and seconds)? What is your average speed in miles per hour?

[50]:
```
minute = int(sec/60//miles)
second = sec/miles-minute*60
print('The average 1-mile pace is about {} minutes, and {} \
seconds.'.format(minute,round(second, 2)))
```

```
The average 1-mile pace is about 6 minutes, and 52.48 seconds.
```

4 Problem 3

(A) We've seen that n = 42 is legal. What about 42 = n? What is the error message?

Answer: SyntaxError: can't assign to literal

(B) Is the following a legal statement? x = y = 1

Answer: Yes, it assigns value 1 to x and y.

(C) In some languages every statement ends with a semi-colon ;. What happens if you put a semi-colon at the end of a Python statement? What function do semi-colons serve in Python?

Answer: It allows to do multiple statements in one line.

(D) What happens if you put a period at the end of a statement

Answer: a period at the end of a statement would create a syntax error.

(E) Explain the difference between the following two lines: **Answer:** They have different data type; a period denotes a float data type.

[84]:
```
n=42
print(n, type(n))
```

```
42 <class 'int'>
```

[83]:
```
n=42.
print(n, type(n))
```

```
42.0 <class 'float'>
```

5 Problem 4

(A) (10 points) The volume of a sphere with radius r is 4/3 r^3 . Write a function sphere_volume(r) that will accept a radius as an argument and return the volume.

[87]:
```
from math import pi
def sphere_volume(r):
    return (4/3) * pi * r ** 3
```

[89]:
```
sphere_volume(5)
```

[89]:
```
523.5987755982989
```

[90]:
```
sphere_volume(15)
```

[90]:
```
14137.166941154068
```

[108]:
```
def wholesale_cost(books):
    assert books >= 0, 'must have non-negative integer for input!'
    new_price = 24.95*(1 - 0.4)
    if books:
        return round(books * new_price + (books - 1) * 0.75 + 3, 2)
    else:
        return 0
```

[102]:
```
wholesale_cost(60)
```

[102]:
```
945.45
```

[103]:
```
wholesale_cost(10)
```

[103]:
```
159.45
```

(C) (10 points) A person runs several miles. The first and last miles are run at an 'easy' pace. Other than the first and last miles, the other miles are at a faster pace.

Write a function run_time(miles, warm_pace, fast_pace) to calculate the time the runner will take. The function accepts three input arguments: how many miles the runner travels (minimum value is 2), the warm-up and cool-down pace, the fast pace. The function will print the time in the format minutes:seconds, and will return a tuple of values: (minutes, seconds) Use the function to find the time to run a total of 5 miles. The warm-up pace is 8:15 per mile. The speed pace is 7:12 per mile.

For now, you can call the function using:

run_time(miles = 5, warm_pace = 495, fast_pace = 432)

[1]:
```
def run_time(miles,warm_pace,fast_pace):
    assert miles>=2
    time = warm_pace*2 + (miles - 2) * fast_pace
    minutes = time // 60
    seconds = time - minutes * 60
    print('Required time is: ' + \
          str(minutes) + ':' + str(seconds))
    return((minutes, seconds))
```

[2]:
```
run_time(miles = 5, warm_pace = 495, fast_pace = 432)
```

```
Required time is: 38:6
```

[2]:
```
(38, 6)
```

[3]:
```
def run_time_super(miles,warm_pace,fast_pace):
    assert miles>=2
    def transform(i):
        i = i.split(':')
        return  int(i[0]) * 60 + int(i[1])
    time = transform(warm_pace) * 2 + (miles - 2) * transform(fast_pace)
    minutes = time // 60
    seconds = time - minutes * 60
    print('Required time is: ' + \
          str(minutes) + ':' + str(seconds))
    return((minutes, seconds))
```

[4]:
```
run_time_super(5, "8:15", "7:12")
```

```
Required time is: 38:6
```

[4]:
```
(38, 6)
```

6 Problem 5

You can use import math to gain access to math functions.

Create a function polar(real, imaginary) that will return the polar coordinates of a complex number. The input arguments are the real and imaginary components of a complex number. The function will return a tuple of values: the value of the radius r and the angle theta.

[18]:
```
import math
def polar(real, imaginary):
    r = pow((real**2 + imaginary**2), 0.5)
    theta = math.atan(imaginary / real)
    return (r, theta)
```

[19]:
```
polar(1, 1)
```

[19]:
```
(1.4142135623730951, 0.7853981633974483)
```

[15]:
```
polar(-2,-3)
```

[15]:
```
(3.605551275463989, 0.982793723247329)
```

[16]:
```
polar(4,2)
```

[16]:
```
(4.47213595499958, 0.46364760900080615)
```

[ ]:
```

```