

Python Assignment 1:

Q1. What is JPython & CPython?

Ans: JPython is an implementation of the high-level, dynamic, object-oriented language Python seamlessly integrated with the Java platform.

CPython can be defined as both an interpreter and a compiler as it compiles Python code into bytecode before interpreting it.

Q2. What is the basic difference between python 2 and python 3?

Ans: **Python 2** stores need to define Unicode string value with "u." whereas **Python 3** default storing of strings is Unicode.

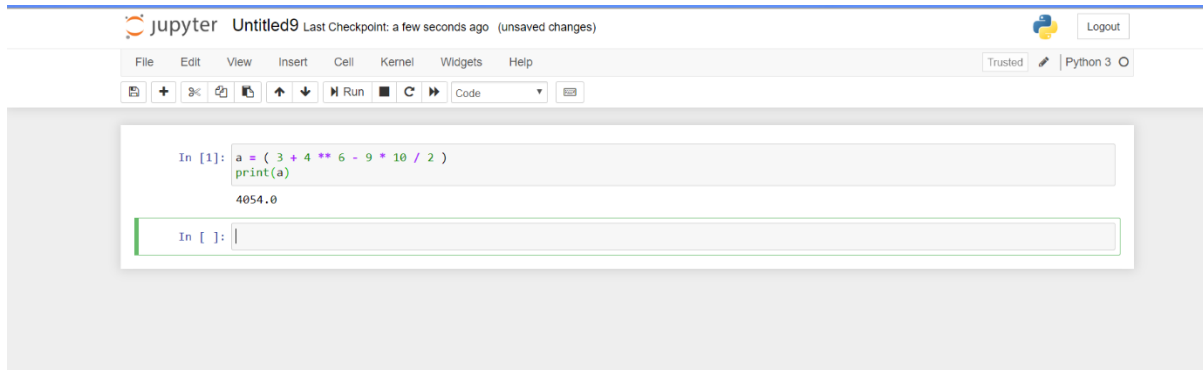
Python 2 syntax is difficult to interstand whereas **Python 3** syntax **is** simpler and easily understandable.

Q3. Difference between ASCII & Unicode?

Ans: ASCII uses 7-bits to encode each character whereas Unicode uses variable bit encoding program where we can choose between 32, 16, and 8-bit encodings.

Python assignment 2

Q1. What should be the output? $(3 + 4 ** 6 - 9 * 10 / 2)$?



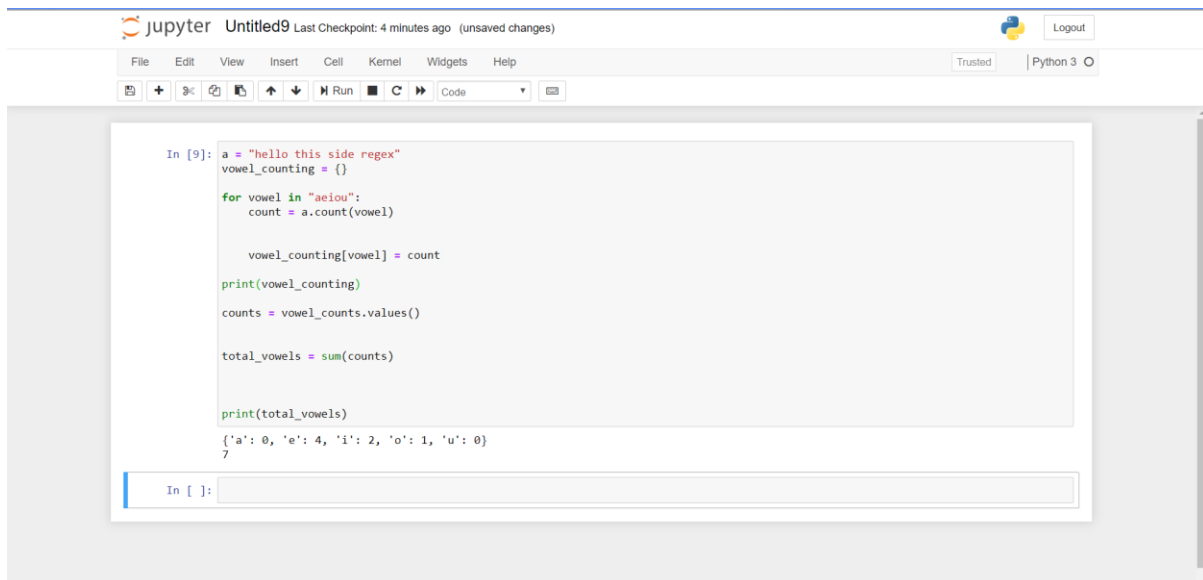
A screenshot of a Jupyter Notebook interface. The top bar shows 'jupyter' and 'Untitled9' with a timestamp 'Last Checkpoint: a few seconds ago (unsaved changes)'. The menu bar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The toolbar has icons for file operations and execution. The code cell contains the following Python code:

```
In [1]: a = ( 3 + 4 ** 6 - 9 * 10 / 2 )
        print(a)
        4054.0
```

The output of the code is 4054.0.

Q2. Let say I have, some string "hello this side regex"

- Find out the count of the total vowels
 - vowels - ['a','e','i','o','u']



A screenshot of a Jupyter Notebook interface. The top bar shows 'jupyter' and 'Untitled9' with a timestamp 'Last Checkpoint: 4 minutes ago (unsaved changes)'. The menu bar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The toolbar has icons for file operations and execution. The code cell contains the following Python code:

```
In [9]: a = "hello this side regex"
        vowel_counting = {}

        for vowel in "aeiou":
            count = a.count(vowel)

            vowel_counting[vowel] = count

        print(vowel_counting)
        counts = vowel_counting.values()

        total_vowels = sum(counts)

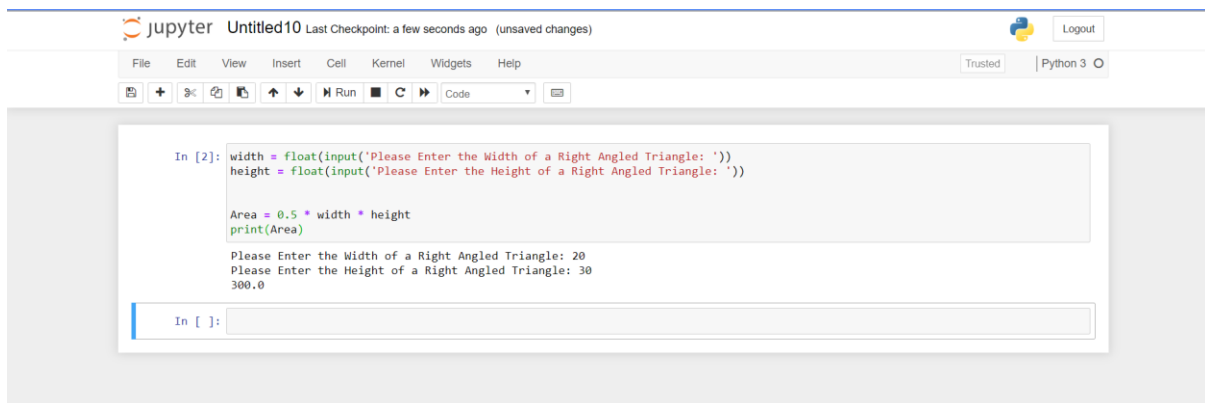
        print(total_vowels)
        {'a': 0, 'e': 4, 'i': 2, 'o': 1, 'u': 0}
        7
```

The output of the code is a dictionary showing the count of each vowel: {'a': 0, 'e': 4, 'i': 2, 'o': 1, 'u': 0} and the total count of vowels is 7.

Q3. Find out the area of triangle

- $\frac{1}{2} * b * h$ (formula of area)

- You have to take value from user about the base, & the height



The image shows a Jupyter Notebook interface with a code cell. The code prompts the user to enter the width and height of a right-angled triangle, calculates the area using the formula $\text{Area} = 0.5 * \text{width} * \text{height}$, and prints the result. The output shows the user entered a width of 20 and a height of 30, resulting in an area of 300.0.

```
In [2]: width = float(input('Please Enter the Width of a Right Angled Triangle: '))
height = float(input('Please Enter the Height of a Right Angled Triangle: '))

Area = 0.5 * width * height
print(Area)

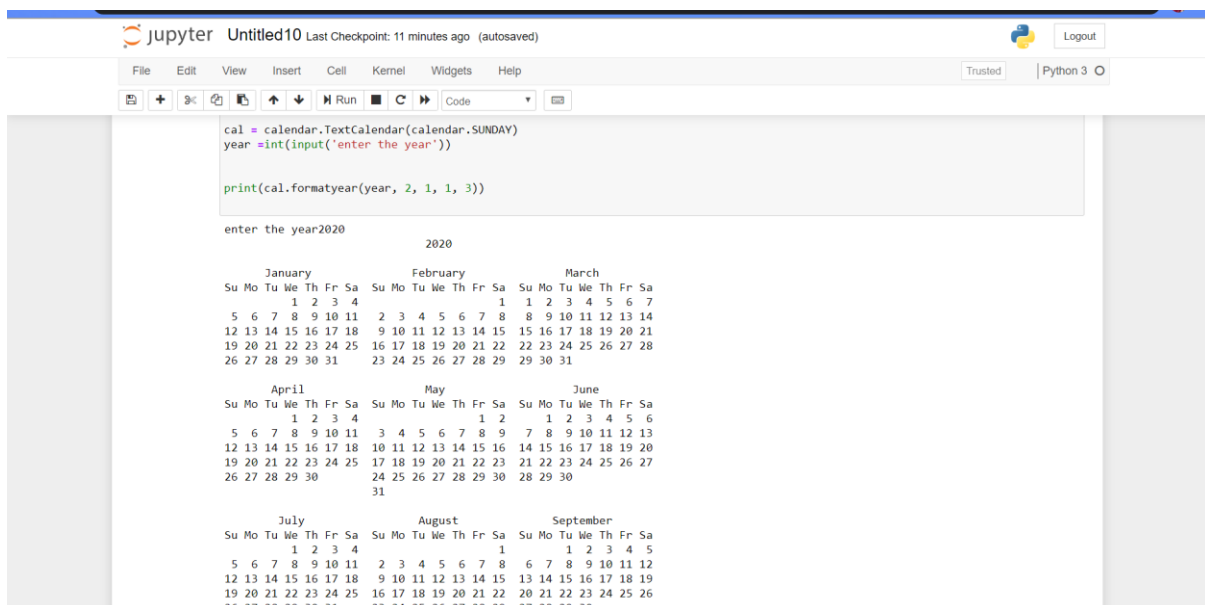
Please Enter the Width of a Right Angled Triangle: 20
Please Enter the Height of a Right Angled Triangle: 30
300.0

In [ ]:
```

Q4. Print the calendar on the terminal. If you give the year.

- Allow the user to input the year.

- Then should that calendar of that year



The image shows a Jupyter Notebook interface with a code cell. The code uses the `calendar` module to create a text calendar for a given year. The user enters the year 2020, and the output displays a detailed calendar for 2020, showing the days of the week and the dates for each month from January to September.

```
cal = calendar.TextCalendar(calendar.SUNDAY)
year = int(input('enter the year'))

print(cal.formatyear(year, 2, 1, 1, 3))

enter the year2020

                2020

    January                February                March
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
   1  2  3  4              1  2  3  4  5  6  7  1  2  3  4  5  6  7
  5  6  7  8  9 10 11    2  3  4  5  6  7  8    8  9 10 11 12 13 14
 12 13 14 15 16 17 18    9 10 11 12 13 14 15    15 16 17 18 19 20 21
 19 20 21 22 23 24 25    16 17 18 19 20 21 22    22 23 24 25 26 27 28
 26 27 28 29 30 31      23 24 25 26 27 28 29    29 30 31

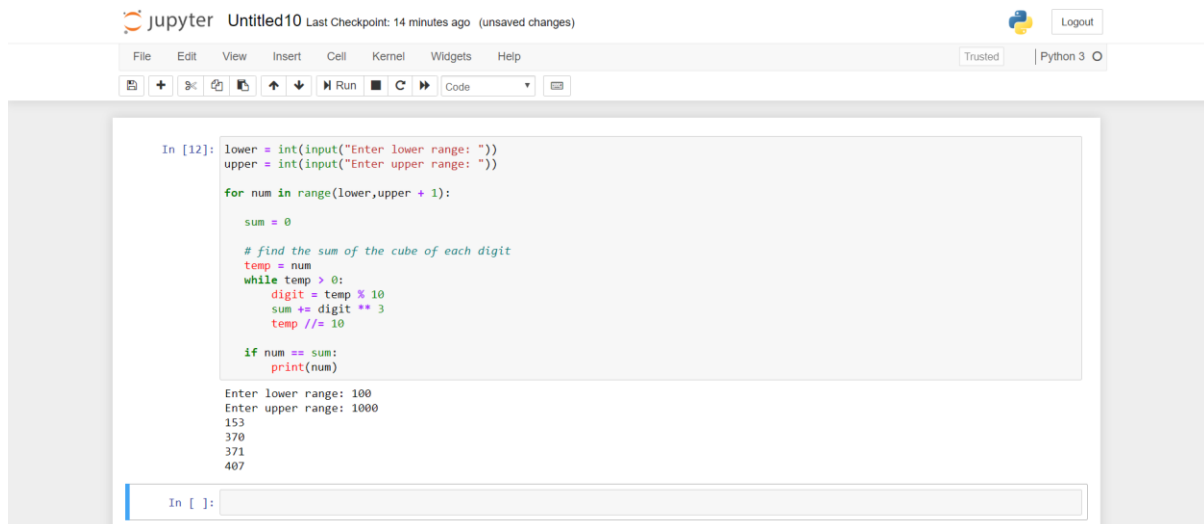
    April                May                June
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
   1  2  3  4              1  2              1  2  3  4  5  6
  5  6  7  8  9 10 11    3  4  5  6  7  8  9    7  8  9 10 11 12 13
 12 13 14 15 16 17 18    10 11 12 13 14 15 16    14 15 16 17 18 19 20
 19 20 21 22 23 24 25    17 18 19 20 21 22 23    21 22 23 24 25 26 27
 26 27 28 29 30          24 25 26 27 28 29 30    28 29 30
                        31

    July                August                September
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
   1  2  3  4              1  2  3  4  5              1  2  3  4  5
  5  6  7  8  9 10 11    2  3  4  5  6  7  8    6  7  8  9 10 11 12
 12 13 14 15 16 17 18    9 10 11 12 13 14 15    13 14 15 16 17 18 19
 19 20 21 22 23 24 25    16 17 18 19 20 21 22    20 21 22 23 24 25 26
 26 27 28 29 30 31      23 24 25 26 27 28 29    27 28 29 30
```

Python Assignment 3

Q1. Find the Armstrong Number between the two numbers which are input by user

- Armstrong number : 153 -> $1*1*1 + 5*5*5 + 3*3*3$



A screenshot of a Jupyter Notebook interface. The top bar shows 'jupyter Untitled10' and 'Last Checkpoint: 14 minutes ago (unsaved changes)'. The menu bar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The toolbar has icons for file operations, running, and code execution. The code cell contains the following Python code:

```
In [12]: lower = int(input("Enter lower range: "))
upper = int(input("Enter upper range: "))

for num in range(lower, upper + 1):
    sum = 0
    # find the sum of the cube of each digit
    temp = num
    while temp > 0:
        digit = temp % 10
        sum += digit ** 3
        temp //= 10
    if num == sum:
        print(num)
```

The output of the code is:

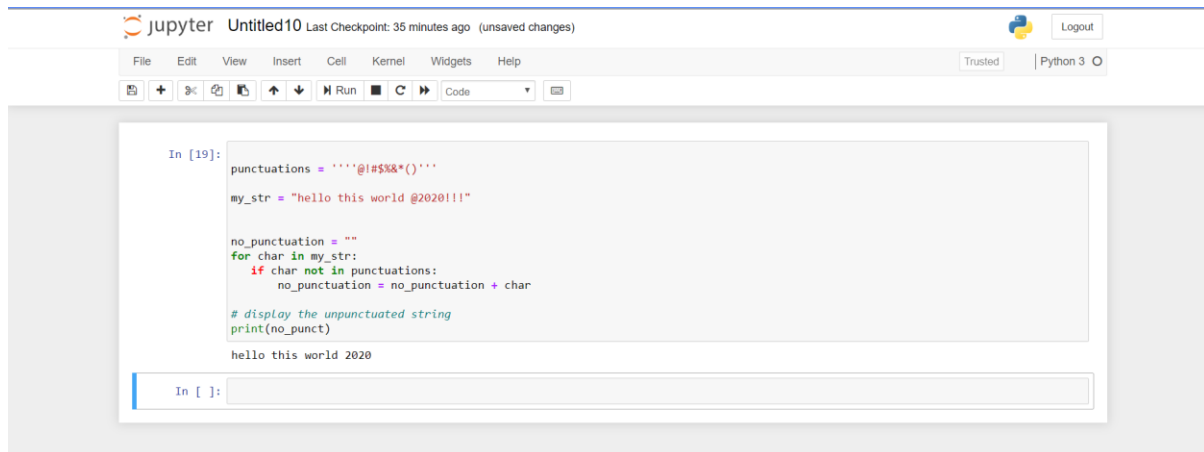
```
Enter lower range: 100
Enter upper range: 1000
153
370
371
407
```

Q2. Let's say you have a string "hello this world @2020!!! "

- Remove the punctuation like ["@!#\$%&*()"] from the string

■ Final output should be without the punctuation

- "hello this world 2020"



A screenshot of a Jupyter Notebook interface. The top bar shows 'jupyter Untitled10' and 'Last Checkpoint: 35 minutes ago (unsaved changes)'. The menu bar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The toolbar has icons for file operations, running, and code execution. The code cell contains the following Python code:

```
In [19]: punctuations = '''!@#$%^&*()'''
my_str = "hello this world @2020!!!"

no_punctuation = ""
for char in my_str:
    if char not in punctuations:
        no_punctuation = no_punctuation + char

# display the unpunctuated string
print(no_punct)
```

The output of the code is:

```
hello this world 2020
```