PythonPracticalWork

April 6, 2020

https://www.youtube.com/watch?v=M5ILgNI0iXw&t=272s

[3]:
```
print("Hello World")
name = input("Enter your name: ")
print(f"Welcome user {name.upper()}".center(100, "_"))
# shift + enter
```

```
Hello World
Enter your name: sachin yadav
_____________________________________Welcome user SACHIN
YADAV_____________________________________
```

[19]:
```
country = [ 'India', 'US', 'UK', 'CHINA', 'Itely'] # list
corona =  [ 1500, 2500, 2350, 3500, 3300 ]
color =   [ 'blue', 'orange', 'magenta', 'red', 'cyan']
```

[20]:
```
print(country)
```

```
['India', 'US', 'UK', 'CHINA', 'Itely']
```

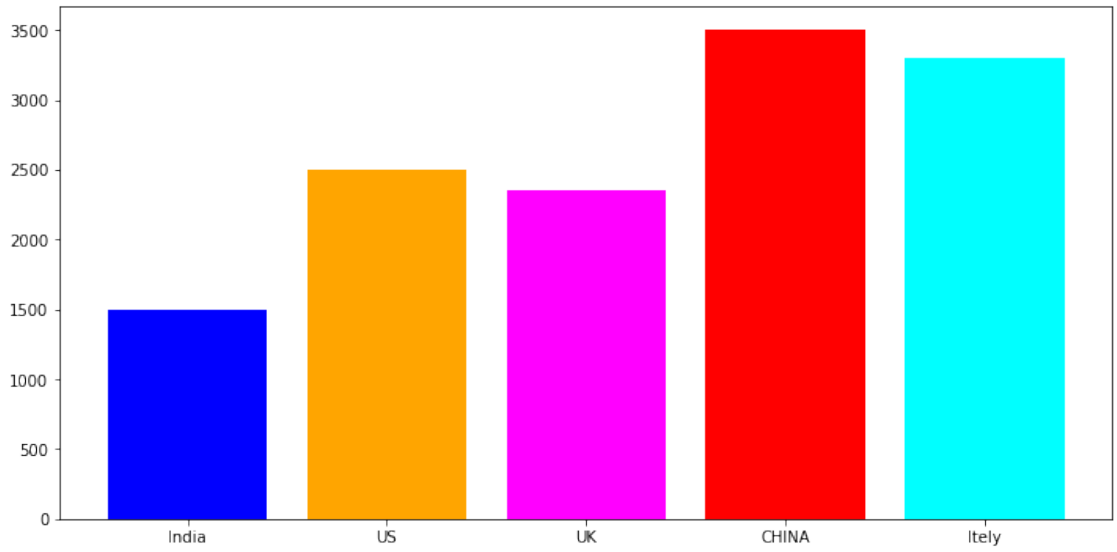
[21]:
```
country[3]
```

[21]:
```
'CHINA'
```

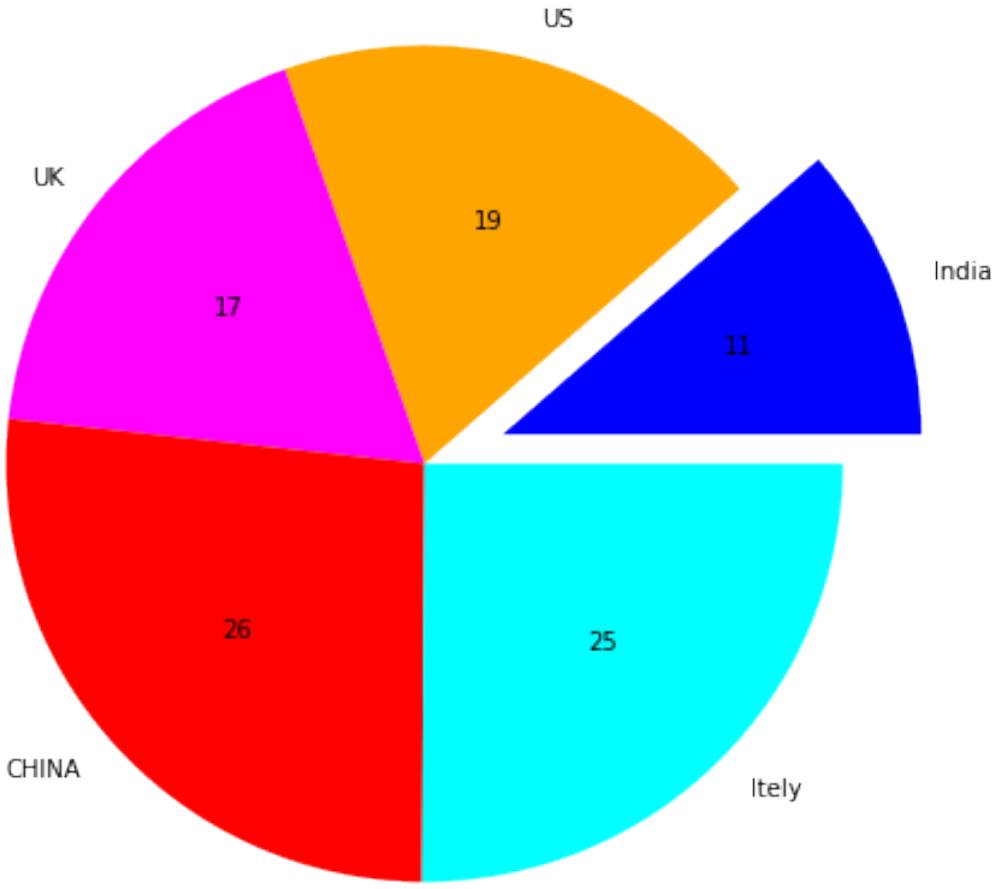
[22]:
```
import matplotlib.pyplot as plt
```

[23]:
```
%matplotlib inline
```

[24]:
```
plt.figure(figsize=(12, 6))
plt.bar(country, corona, color=color)
plt.show()
```



[31]:
```
import matplotlib.pyplot as plt
country = [ 'India', 'US', 'UK', 'CHINA', 'Itely'] # list
corona =  [ 1500, 2500, 2350, 3500, 3300 ]
color =   [ 'blue', 'orange', 'magenta', 'red', 'cyan']
plt.figure(figsize=(8, 8))
plt.pie(corona, labels=country, colors=color, autopct="%d",
       explode=[  0.2, 0, 0, 0, 0])
plt.show()
```



[35]:
```
url = "https://www.technipages.com/wp-content/uploads/2019/12/File-Header.jpg"
```

[36]:
```
import requests
```

[37]:
```
page = requests.get(url)
```

[39]:
```
page.headers['Content-type']
```

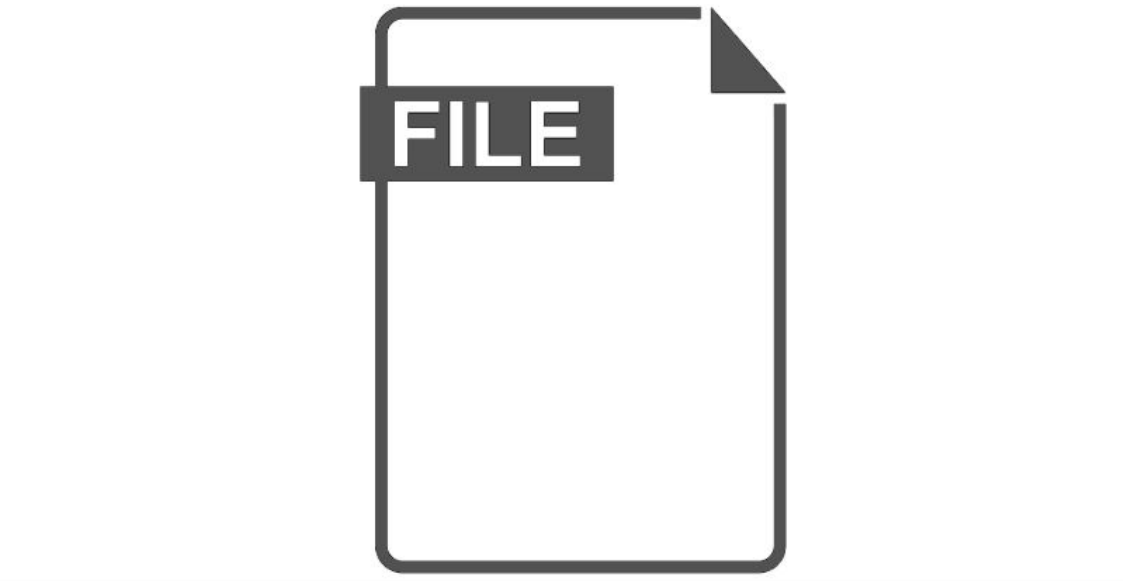
[39]: 'image/jpeg'

[40]:
```
fp = open('file_header.jpeg', 'wb')
fp.write(page.content)
fp.close()
```

[41]:
```
from PIL import Image
```

[42]:
```
Image.open('file_header.jpeg')
```

[42]:



[43]:
```
# Data type
# if - else
# loop
# function

# web / gui app
```

[45]:
```
%%writefile break_timer.py
import os
import time
seconds = 10*60
for i in range(seconds+1):
    os.system('cls')
    print("\n\n\n\n\n\n")
    print(f"Time Left in Break: {seconds-i} seconds".center(100))
    time.sleep(1)
```

```
Writing break_timer.py
```

[46]:
```
pwd
```

[46]:
```
'C:\\Users\\sachin\\Desktop\\python_Webinar'
```

Data Structures

```
1. Numbers --> int, float, complex
2. Strings --> single line, multi-line
3. List --> collection homogenous or hetrogenous data type
4. Dictinory --> map type object key-value pair store
```

[47]:
```python
x = 5
```

[48]:
```python
print(type(x))
```

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

[49]:
```python
x = ␣
↪2993274732472849327473247237437249732984789324798237497239479324789237842394798237489327894
```

[50]:
```python
print(x)
```

```
29932747324728493274732472374372497329847893247982374972394793247892378423947982
37489327894793284789324789234798
```

[51]:
```python
print(type(x))
```

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

[52]:
```python
print(dir(x))
```

```
['__abs__', '__add__', '__and__', '__bool__', '__ceil__', '__class__',
'__delattr__', '__dir__', '__divmod__', '__doc__', '__eq__', '__float__',
'__floor__', '__floordiv__', '__format__', '__ge__', '__getattribute__',
'__getnewargs__', '__gt__', '__hash__', '__index__', '__init__',
'__init_subclass__', '__int__', '__invert__', '__le__', '__lshift__', '__lt__',
'__mod__', '__mul__', '__ne__', '__neg__', '__new__', '__or__', '__pos__',
'__pow__', '__radd__', '__rand__', '__rdivmod__', '__reduce__', '__reduce_ex__',
'__repr__', '__rfloordiv__', '__rlshift__', '__rmod__', '__rmul__', '__ror__',
'__round__', '__rpow__', '__rrshift__', '__rshift__', '__rsub__',
'__rtruediv__', '__rxor__', '__setattr__', '__sizeof__', '__str__', '__sub__',
'__subclasshook__', '__truediv__', '__trunc__', '__xor__', 'bit_length',
'conjugate', 'denominator', 'from_bytes', 'imag', 'numerator', 'real',
'to_bytes']
```

[53]:
```python
x.bit_length()
```

[53]: 371

[54]:
```python
x = 5.6
```

[55]:
```python
print(type(x))
```

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

[56]:
```
print(dir(x))
```

```
['__abs__', '__add__', '__bool__', '__class__', '__delattr__', '__dir__',
'__divmod__', '__doc__', '__eq__', '__float__', '__floordiv__', '__format__',
'__ge__', '__getattribute__', '__getformat__', '__getnewargs__', '__gt__',
'__hash__', '__init__', '__init_subclass__', '__int__', '__le__', '__lt__',
'__mod__', '__mul__', '__ne__', '__neg__', '__new__', '__pos__', '__pow__',
'__radd__', '__rdivmod__', '__reduce__', '__reduce_ex__', '__repr__',
'__rfloordiv__', '__rmod__', '__rmul__', '__round__', '__rpow__', '__rsub__',
'__rtruediv__', '__set_format__', '__setattr__', '__sizeof__', '__str__',
'__sub__', '__subclasshook__', '__truediv__', '__trunc__', 'as_integer_ratio',
'conjugate', 'fromhex', 'hex', 'imag', 'is_integer', 'real']
```

[57]:
```
x = 15 + 5j # imag --> j
```

[58]:
```
print(type(x))
```

```
<class 'complex'>
```

[59]:
```
5 + 6
```

[59]: 11

[60]:
```
5 * 6
```

[60]: 30

[61]:
```
5 / 6
```

[61]: 0.8333333333333334

[62]:
```
5 % 6
```

[62]: 5

String

[63]:
```
s = "Hello World"
```

[64]:
```
print(type(s))
```

```
<class 'str'>
```

[65]:
```
print(dir(s))
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__',
'__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__',
'__getnewargs__', '__gt__', '__hash__', '__init__', '__init_subclass__',
```

```
'__iter__', '__le__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__',
'__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmod__', '__rmul__',
'__setattr__', '__sizeof__', '__str__', '__subclasshook__', 'capitalize',
'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find',
'format', 'format_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal',
'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace',
'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans',
'partition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit',
'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title',
'translate', 'upper', 'zfill']
```

[66]:
```
s.swapcase()
```

[66]:
```
'hELLO wORLD'
```

[67]:
```
s.title()
```

[67]:
```
'Hello World'
```

[68]:
```
s.lower()
```

[68]:
```
'hello world'
```

[69]:
```
s.upper()
```

[69]:
```
'HELLO WORLD'
```

```
__ --> operators
```

[70]:
```
5 + 6
```

[70]:
```
11
```

[71]:
```
int.__add__(5, 6)
```

[71]:
```
11
```

[72]:
```
5 + 6 # int.__add__(5, 6)
```

[72]:
```
11
```

[73]:
```
"Hello " + "World"
```

[73]:
```
'Hello World'
```

[74]:
```
str.__add__("Hello ", "World")
```

[74]:
```
'Hello World'
```

[75]:
```
print(dir(s))
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__',
'__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__',
'__getnewargs__', '__gt__', '__hash__', '__init__', '__init_subclass__',
'__iter__', '__le__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__',
'__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmod__', '__rmul__',
'__setattr__', '__sizeof__', '__str__', '__subclasshook__', 'capitalize',
'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find',
'format', 'format_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal',
'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace',
'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans',
'partition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit',
'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title',
'translate', 'upper', 'zfill']
```

[76]:
```
help(str.center)
```

```
Help on method_descriptor:

center(self, width, fillchar=' ', /)
    Return a centered string of length width.

    Padding is done using the specified fill character (default is a space).
```

[77]:
```
s = "Sachin Yadav"
```

[78]:
```
s.center(100, "*")
```

[78]:
```
'**********************************************Sachin
Yadav**********************************************'
```

[79]:
```
# list
```

[80]:
```
lang = [ 'java', 'c', 'c++', 'ruby', 'perl']
```

[81]:
```
print(type(lang))
```

```
<class 'list'>
```

[82]:
```
print(dir(lang))
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__delitem__',
'__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__',
'__getitem__', '__gt__', '__hash__', '__iadd__', '__imul__', '__init__',
'__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__mul__',
'__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__reversed__',
```

```
'__rmul__', '__setattr__', '__setitem__', '__sizeof__', '__str__',
'__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index',
'insert', 'pop', 'remove', 'reverse', 'sort']
```

[83]:
```
lang.append('python')
```

[84]:
```
print(lang)
```

```
['java', 'c', 'c++', 'ruby', 'perl', 'python']
```

[85]:
```
lang.insert(1, "php" )
```

[86]:
```
print(lang)
```

```
['java', 'php', 'c', 'c++', 'ruby', 'perl', 'python']
```

[87]:
```
lang.remove('ruby')
```

[89]:
```
lang
```

[89]:
```
['java', 'php', 'c', 'c++', 'perl', 'python']
```

[90]:
```
lang.pop(1)
```

[90]:
```
'php'
```

[91]:
```
lang
```

[91]:
```
['java', 'c', 'c++', 'perl', 'python']
```

Real Example

[92]:
```
marksheet = [
    [ 'sachin', 67],
    [ 'astik', 90],
    [ 'avik', 78],
    [ 'divya', 90],
    [ 'mohit', 70]
]
```

[93]:
```
print(marksheet)
```

```
[['sachin', 67], ['astik', 90], ['avik', 78], ['divya', 90], ['mohit', 70]]
```

[98]:
```
marksheet.sort(reverse=True)
```

[99]:
```
marksheet
```

[99]:
```
[['sachin', 67], ['mohit', 70], ['divya', 90], ['avik', 78], ['astik', 90]]
```

[97]:
```
from operator import itemgetter
```

[100]:
```
marksheet.sort(key=itemgetter(1), reverse=True)
```

[101]:
```
marksheet
```

[101]:
```
[['divya', 90], ['astik', 90], ['avik', 78], ['mohit', 70], ['sachin', 67]]
```

dict

[102]:
```
bank = {

    1001: {'name': 'sachin', 'balance': 10000, 'password': 'redhat'},
    1002: {'name': 'rajat', 'balance': 25000,  'password': 'rajat@123'},
    1003: {'name': 'nidhi', 'balance': 15000, 'password': 'nidhi123'}
}
```

[103]:
```
bank[1001]
```

[103]:
```
{'name': 'sachin', 'balance': 10000, 'password': 'redhat'}
```

[104]:
```
bank[1002]
```

[104]:
```
{'name': 'rajat', 'balance': 25000, 'password': 'rajat@123'}
```

[106]:
```
bank[1001]['name']
```

[106]:
```
'sachin'
```

[107]:
```
bank[1002]['password']
```

[107]:
```
'rajat@123'
```

[108]:
```
bank[1003]['balance']
```

[108]:
```
15000
```

Greatest among three number

[114]:
```
x, y, z = list(map(int, input().split()))
print(x, y, z, sep='\n')
```

```
10 35 10
10
35
10
```

[117]:
```
x,y,z = map(int, input("x y z").split())
print("Value of x is : ", x)
print("Value of y is : ", y)
print("Value of z is : ", z)

if x >= y and x >= z:
    print(f"{x} is greatest")
elif y >= z:
    print(f"{y} is greatest")
else:
    print(f"{z} is greatest")
```

```
x y z1 2 3
Value of x is :  1
Value of y is :  2
Value of z is :  3
3 is greatest
```

```
loop
```

```
intilization

condition

increment/ decrement
```

[119]:
```
c = 1

while c <= 10:
    print(f"Hello World Times {c}")
    c += 1
else:
    print("else will run if while completes it's condtion")
```

```
Hello World Times 1
Hello World Times 2
Hello World Times 3
Hello World Times 4
Hello World Times 5
Hello World Times 6
Hello World Times 7
Hello World Times 8
Hello World Times 9
Hello World Times 10
else will run if while completes it's condtion
```

[120]:
```
c = 1

while c <= 10:
```

```
    print(f"Hello World Times {c}")
    if c == 5:
        break
    c += 1
else:
    print("else will run if while completes it's condtion")
```

```
Hello World Times 1
Hello World Times 2
Hello World Times 3
Hello World Times 4
Hello World Times 5
```

[121]:
```
c = 1
while c <= 10:
    c += 1
    if  c % 2:
        continue
    print("Hello World Times", c)
else:
    print("Do you get it ?")
```

```
Hello World Times 2
Hello World Times 4
Hello World Times 6
Hello World Times 8
Hello World Times 10
Do you get it ?
```

Guess Game

[122]:
```
import random
```

[126]:
```
random.randint(10, 50)
```

[126]: 23

[127]:
```
# computer 1 random number guess 1-50
# user --> 5 chances to guesss
# each invalid guess user hint --> low, high
```

[129]:
```
comGuess = random.randint(1, 50)
chances = 1
while chances <= 5:
    print(f"You Have Left {6-chances} to Guess")
    userGuess = int(input("User Guess(1, 50) : "))
    if userGuess >= 1 and userGuess <= 50:
        if userGuess > comGuess:
```

```
            print("Hint: Your Guess is High.")
        elif userGuess < comGuess:
            print("Hint: Your Guess is Low")
        else:
            print("Whoooo!! Genius!!!!You have won the game")
            break
        chances += 1
    else:
        print("Invalid Guess Guess Between 1-50")
else:
    print("You Such a Looser")
    print("Computer Guess Was: ", comGuess)
```

```
You Have Left 5 to Guess
User Guess(1, 50) : 1000
Invalid Guess Guess Between 1-50
You Have Left 5 to Guess
User Guess(1, 50) : 25
Hint: Your Guess is Low
You Have Left 4 to Guess
User Guess(1, 50) : 40
Hint: Your Guess is Low
You Have Left 3 to Guess
User Guess(1, 50) : 45
Hint: Your Guess is Low
You Have Left 2 to Guess
User Guess(1, 50) : 48
Hint: Your Guess is High.
You Have Left 1 to Guess
User Guess(1, 50) : 46
Whoooo!! Genius!!!!You have won the game
```

[133]:
```
data = [ 'java', 'c', 'c++', 'ruby', 'perl', 'python']
```

[135]:
```
for item in data:
    if item != 'python':
        print(f"{item} is very boring")
    else:
        print(f"{item} is Awesome".upper().center(100, '_'))
```

```
java is very boring
c is very boring
c++ is very boring
ruby is very boring
perl is very boring
_____________________________________________PYTHON IS
AWESOME_____________________________________________
```

[136]:
```
def hello(name):
    print(f"Hello {name} this is a function.")
```

[137]:
```
hello('sachin')
```

```
Hello sachin this is a function.
```

[141]:
```
def prime(number):
    if number <= 1:
        return False
    elif number <= 3:
        return True
    else:
        for check in range(2, number //2):
            if number % check == 0:
                return False
        return True
```

[142]:
```
prime(127)
```

[142]:
```
True
```

[143]:
```
prime(121)
```

[143]:
```
False
```

[144]:
```
prime(13)
```

[144]:
```
True
```

[145]:
```
prime(100)
```

[145]:
```
False
```

[146]:
```
def prime_range(start, end):
    counter = 1
    for number in range(start, end+1):
        if prime(number):
            counter += 1
            print(number, end=', ')
    else:
        print(f"\n\ntotal Prime nubmers in Given range({start}, {end}) are : ",␣
↪counter)
```

[147]:
```
prime_range(1, 100)
```

```
2, 3, 4, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97,
```

```
total Prime nubmers in Given range(1, 100) are :  27
```

[148]:
```
even = lambda number: True if number % 2 == 0 else False
```

[149]:
```
even(12)
```

[149]: True

[150]:
```
even(13)
```

[150]: False

[151]:
```
2**8
```

[151]: 256

[162]:
```
prime = lambda number, check=2: True if check > int(number**0.5) else False if␣
↪number % check == 0 else prime(number, check+1)
```

[163]:
```
prime(127)
```

[163]: True

[164]:
```
prime(121)
```

[164]: False

[165]:
```
prime_range(50, 100)
```

```
53, 59, 61, 67, 71, 73, 79, 83, 89, 97,

total Prime nubmers in Given range(50, 100) are :  11
```

[166]:
```
prime_range(1, 100)
```

```
1, 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97,

total Prime nubmers in Given range(1, 100) are :  27
```

[201]:
```
%%writefile calc.py

import tkinter
import random

colors = [ 'red', 'green', 'blue', 'cyan', 'magenta', '#123456', 'brown',
         'black']
```

```
root = tkinter.Tk()
l1 = tkinter.Label(root, text="My Personal Calculator",
                   font=('Times', 20, 'bold'), fg='red')
l1.pack()

s = tkinter.StringVar()
e = tkinter.Entry(root, textvariable=s, font=('Times', 30, 'bold'),
                  bg="#123456", fg='white')
e.pack()


def add_label():
    color = random.choice(colors)
    ans = eval(s.get())
    text = f"{s.get()} = {ans:.2f}"
    label = tkinter.Label(root, text=text,  font=('Times', 20, 'bold'),␣
 ↪fg=color)
    s.set('')
    label.pack()


b = tkinter.Button(root, text="!Solve Me!", command=add_label,
                   font=15, bg='#123456', fg='white', height=1, width=25)
b.pack()

exit_button = tkinter.Button(root, text="!EXIT!", command=root.destroy,
                             font=15, bg='#123456', fg='white', height=1,␣
 ↪width=25)
exit_button.pack()

root.title('Calculator')
root.wm_minsize(800, 600)
root.mainloop()
```

```
Overwriting calc.py
```

[202]:
```
eval('12*5/6')
```

[202]:
```
10.0
```

[203]:
```
!python calc.py
```

```
https://github.com/sachinyadav3496
```

[ ]: