- Moatasem Elsayed

Bio

- Embedded Linux Software Engineer
- Embedded Software Engineer
- Embedded Software Engineer
- Founder & CEO
- Mentoring For Graduation Project +40
- Instructor at Embedded Systems 75+ G

Eng.moatasem.9@gmail.com

01112932885

Containers

Content

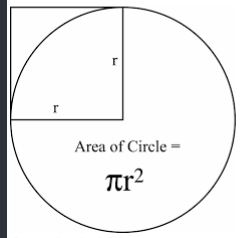
- Types of function
- Strings
- Help
- Function
- Global variable
- Lambda
- Quick task
- Lab
- Modules
- Garbage Collector
- List
- Tuple
- Set
- Tasks

Tasks(2)

Write a Python program which accepts the radius of a circle from the user and compute the a

Python: Area of a Circle

In geometry, the area enclosed by a circle of radius r is πr2. Here the Greek letter π represents a constant, approximately equal to 3.14159, which is equal to the ratio of the circumference of any circle to its diameter.




```
#second task
from math import pi
r = float(input ("Input the radius of the circle : "))
print ("The area of the circle with radius " + str(r) + " is: " + str(pi * r**2))
```

Tasks(3)

Print the calendar of a given month and year

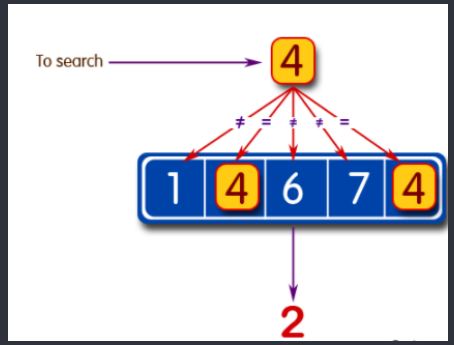
```
Input the year : 2017
Input the month : 04
     April 2017
Mo Tu We Th Fr Sa Su
                1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```

```python
#third task
import calendar
y = int(input("Input the year : "))
m = int(input("Input the month : "))
print(calendar.month(y, m))
```

Tasks

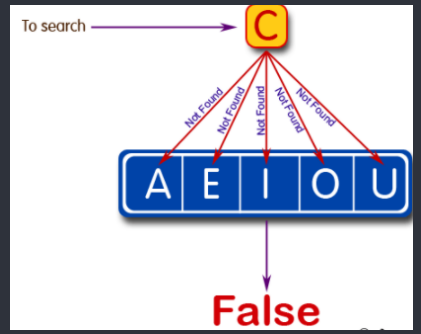
Write a Python program to count the number 4 in a given list.

```python
def list_count_4(nums):
  count = 0
  for num in nums:
    if num == 4:
      count = count + 1

  return count

print(list_count_4([1, 4, 6, 7, 4]))
print(list_count_4([1, 4, 6, 4, 7, 4]))
```



Write a Python program to test whether a passed letter is a vowel or not.

```python
def is_vowel(char):
    all_vowels = 'aeiou'
    return char in all_vowels
print(is_vowel('c'))
print(is_vowel('e'))
```



Write a python program to access environment variables.

PATH

os

```python
import os
# Access all environment variables
print('*---------------------------------*')
print(os.environ)
print('*---------------------------------*')
# Access a particular environment variable
print(os.environ['HOME'])
print('*---------------------------------*')
print(os.environ['PATH'])
print('*---------------------------------*')
```

Built-in Methods

String Methods

capitalize()	Converts the first character to upper case
casefold()	Converts string into lower case
center()	Returns a centered string
count()	Returns the number of times a specified value occurs in a string
encode()	Returns an encoded version of the string
endswith()	Returns true if the string ends with the specified value
expandtabs()	Sets the tab size of the string
find()	Searches the string for a specified value and returns the position of where it was found
format()	Formats specified values in a string
format_map()	Formats specified values in a string
index()	Searches the string for a specified value and returns the position of where it was found
isalnum()	Returns True if all characters in the string are alphanumeric
isalpha()	Returns True if all characters in the string are in the alphabet
isdecimal()	Returns True if all characters in the string are decimals

String Methods

```
#strip() method removes any whitespace from the beginning or the end:
a = " Hello, World! "
print(a.strip()) # returns "Hello, World!"
##############################
#The lower() method returns the string in lower case:
a = "Hello, World!"
print(a.lower())
##############################
#The upper() method returns the string in upper case:
a = "Hello, World!"
print(a.upper())
##############################
a = "Hello, World!"
print(a.replace("H", "J"))
##############################
a = "Hello, World!"
print(a.split(","))
print(type(a.split(",")))
#what's the type of the return ?!
##############################
txt = "The rain in Spain stays mainly in the plain"
x = "ain" in txt
print(x)
```

```
PS D:\Embedded System
Hello, World!
hello, world!
HELLO, WORLD!
Jello, World!
['Hello', ' World!']
<class 'list'>
True
HelloWorld
```

String Methods

```
32 #Use the format() method to insert numbers into strings:
33 age = 36
34 txt = "My name is John, and I am {}"
35 print(txt.format(age))
36 ################################
37 quantity = 3
38 itemno = 567
39 price = 49.95
40 myorder = "I want {} pieces of item {} for {} dollars."
41 print(myorder.format(quantity, itemno, price))
42 ################################
43 quantity = 3
44 itemno = 567
45 price = 49.95
46 myorder = "I want to pay {2} dollars for {0} pieces of item {1}."
47 print(myorder.format(quantity, itemno, price))
48 print(myorder) #take care
49 ################################
50 #Escape operator in python is \
51 # txt = "We are the so-called "Vikings" from the north." #ERROR
52 # print(txt)
53 txt = "We are the so-called \"Vikings\" from the north."
54 print(txt)
```

```
My name is John, and I am 36
I want 3 pieces of item 567 for 49.95 dollars.
I want to pay 49.95 dollars for 3 pieces of item 567.
I want to pay {2} dollars for {0} pieces of item {1}.
We are the so-called "Vikings" from the north.
Reverse is dlroWolleH
```

```
59  ##############################################3
60  #The join() method takes all items in an iterable an
61  a = ["How", "are", "you"]
62  print(" ana hena ".join(a))
63  ##############################
64  # multiply sting
65  n = 2
66  a = "HowAreYou"
67  print(a * n)
68  ##########################################################
69  txt = "Hello, welcome to my world."
70  x = txt.find("welcome")
71  print(x)
72  txt = "Hello, welcome to my world."
73  x = txt.find("e", 5, 10)
74  print(x)
75  ###########################################
76  # The count() method returns the number of times a s
77  txt = "I love apples, apple are my favorite fruit"
78  x = txt.count("apple")
79  print(x)
80  ########################################
81  # The isnumeric() method returns True if all the ch
82  txt = "565543"
83  x = txt.isnumeric()
84  print(x)
85  ########################################
86  # The isprintable() method returns True if all the
87  txt = "Hello! Are you #1?"
88  x = txt.isprintable()
89  print(x)
```

```
How ana hena are ana hena you
HowAreYouHowAreYou
7
8
2
True
True
PS D:\Embedded System\Embedded L
```

String substitution

```
#substitution
x=10
print("Welcome Num: "+str(x)+" hello")#+str()+
print("Welcome Num:{} hello".format(x))#.format(x,y,z,l)
print(f"Welcome Num:{x} hello") # {x}  {y}
print("Welcome Num: %d hello"% x)# % (x,y,z,l)
#in print only
print("Welcome Num:",x,"hello")
```

```
PS D:\Embedded System\Embe
Welcome Num: 10 hello
Welcome Num:10 hello
Welcome Num:10 hello
Welcome Num: 10 hello
Welcome Num: 10 hello
PS D:\Embedded System\Embe
```

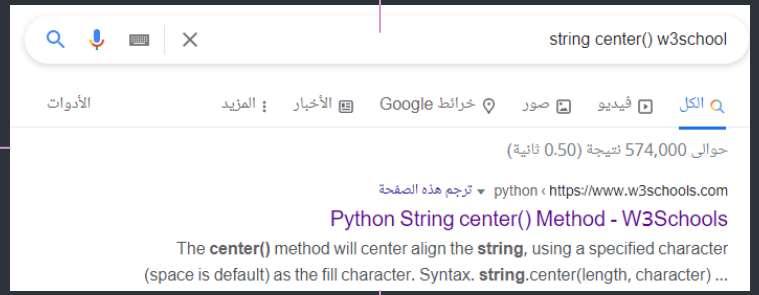
Raw String

```
raw_s = r'Hi\nHello'
print(raw_s)
```

```
PS D:\Embedd
Hi\nHello
```

Python is searchable language

encode()	Returns an encoded version of the string
endswith()	Returns true if the string ends with the specified value
expandtabs()	Sets the tab size of the string
find()	Searches the string for a specified value and returns the position of where it was found
format()	Formats specified values in a string
format_map()	Formats specified values in a string
index()	Searches the string for a specified value and returns the position of where it was found
isalnum()	Returns True if all characters in the string are alphanumeric
isalpha()	Returns True if all characters in the string are in the alphabet
isdecimal()	Returns True if all characters in the string are decimals
isdigit()	Returns True if all characters in the string are digits
isidentifier()	Returns True if the string is an identifier
islower()	Returns True if all characters in the string are lower case
isnumeric()	Returns True if all characters in the string are numeric
isprintable()	Returns True if all characters in the string are printable
isspace()	Returns True if all characters in the string are whitespaces
istitle()	Returns True if the string follows the rules of a title
isupper()	Returns True if all characters in the string are upper case
join()	Joins the elements of an iterable to the end of the string
ljust()	Returns a left justified version of the string
lower()	Converts a string into lower case
lstrip()	Returns a left trim version of the string
maketrans()	Returns a translation table to be used in translations
partition()	Returns a tuple where the string is parted into three parts
replace()	Returns a string where a specified value is replaced with a specified value
rfind()	Searches the string for a specified value and returns the last position of where it was foun
rindex()	Searches the string for a specified value and returns the last position of where it was foun
rjust()	Returns a right justified version of the string
rpartition()	Returns a tuple where the string is parted into three parts
rsplit()	Splits the string at the specified separator, and returns a list
rstrip()	Returns a right trim version of the string
split()	Splits the string at the specified separator, and returns a list
splitlines()	Splits the string at line breaks and returns a list
startswith()	Returns true if the string starts with the specified value
strip()	Returns a trimmed version of the string
swapcase()	Swaps cases, lower case becomes upper case and vice versa
title()	Converts the first character of each word to upper case
translate()	Returns a translated string
upper()	Converts a string into upper case
zfill()	Fills the string with a specified number of 0 values at the beginning



Example

Print the word "banana", taking

```
txt = "banana"

x = txt.center(20)

print(x)
```

Try it Yourself »

Syntax

```
string.center(length, character)
```

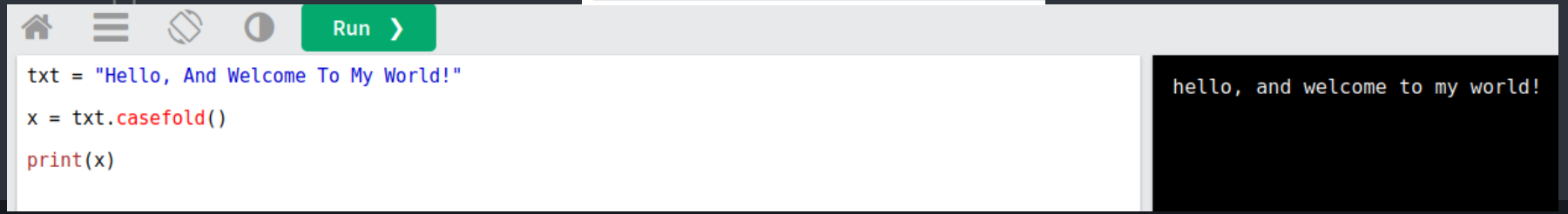
Parameter Values

Parameter	Description
length	Required. The length of the returned string
character	Optional. The character to fill the missing spac

Task

https://www.w3schools.com/python/python_ref_string.asp

Just check all methods of string and run all code on w3school to gain experience



Help

```
moatsem@moatsem-IdeaPad-Gaming-3-1
Python 3.10.6 (main, May 29 2023,
Type "help", "copyright", "credits
>>> help
Type help() for interactive help,
>>> help()

Welcome to Python 3.10's help util

If this is your first time using P
the tutorial on the internet at ht

Enter the name of any module, keyw
Python programs and using Python m
return to the interpreter, just type

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help> keywords

Here is a list of the Python keywords.  Enter any keyword to get more help.

False               class               from                or
None                continue            global              pass
True                def                 if                  raise
and                 del                 import              return
as                  elif                in                  try
assert              else                is                  while
async               except              lambda              with
await               finally             nonlocal            yield
break               for                 not

help>
```

```
help(str)

>>> help
Type help() for interactive help, or help(object) for help about object.
>>> help()

Welcome to Python 3.10's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the internet at https://docs.python.org/3.10/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help> str

help>
```

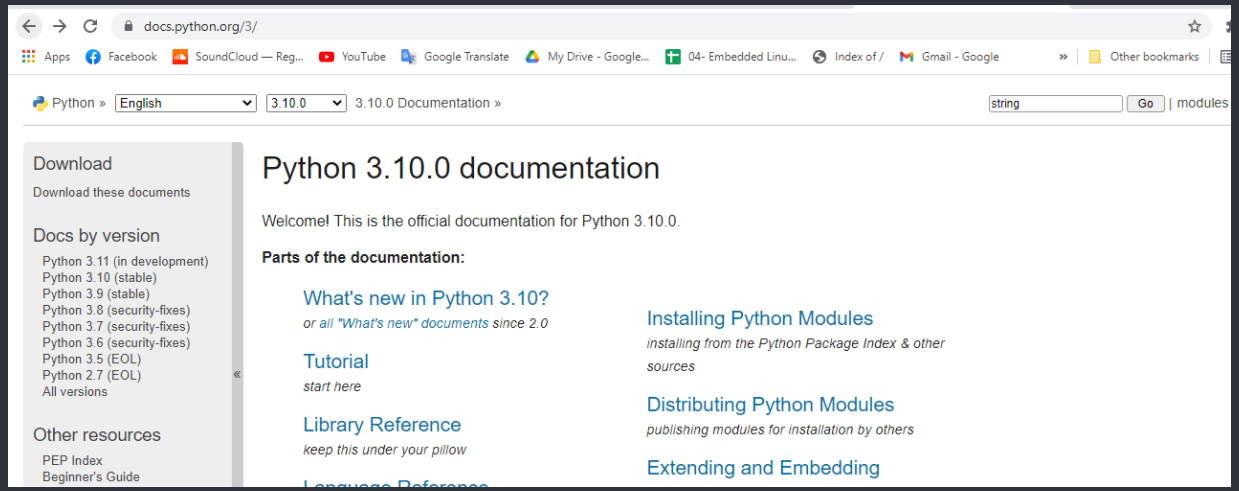
```
moatsem@moa

find(...)
    S.find(sub[, start[, end]]) -> int

    Return the lowest index in S where substring sub is found,
    such that sub is contained within S[start:end].  Optional
    arguments start and end are interpreted as in slice notation.

    Return -1 on failure.

format(...)
    S.format(*args, **kwargs) -> str

    Return a formatted version of S, using substitutions from args and kwargs.
    The substitutions are identified by braces ('{' and '}').

format_map(...)
    S.format_map(mapping) -> str

    Return a formatted version of S, using substitutions from mapping.
    The substitutions are identified by braces ('{' and '}').

index(...)
    S.index(sub[, start[, end]]) -> int

    Return the lowest index in S where substring sub is found,
    such that sub is contained within S[start:end].  Optional
    arguments start and end are interpreted as in slice notation.

    Raises ValueError when the substring is not found.

isalnum(self, /)
    Return True if the string is an alpha-numeric string, False otherwise.

    A string is alpha-numeric if all characters in the string are alpha-numeri
    there is at least one character in the string.

isalpha(self, /)
    Return True if the string is an alphabetic string, False otherwise.

    A string is alphabetic if all characters in the string are alphabetic and
    is at least one character in the string.

isascii(self, /)
    Return True if all characters in the string are ASCII, False otherwise.

    ASCII characters have code points in the range U+0000-U+007F.
    Empty string is ASCII too.

isdecimal(self, /)
    Return True if the string is a decimal string, False otherwise.

    A string is a decimal string if all characters in the string are decimal a
/find
```

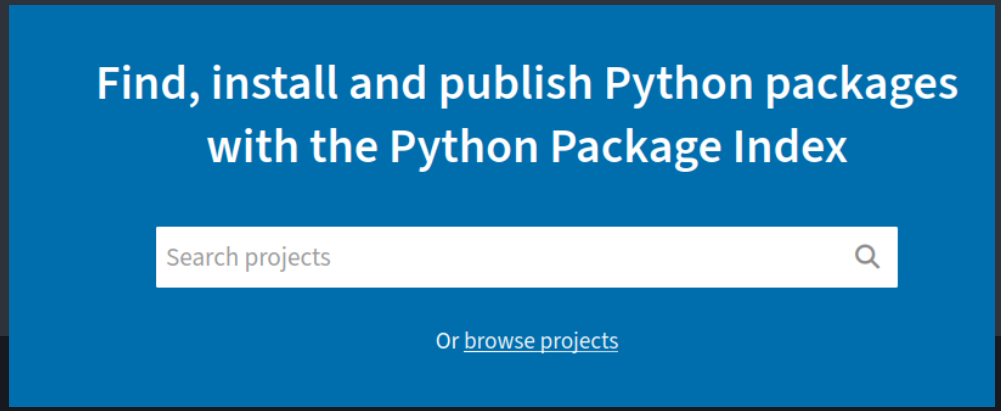
List all methods using dir

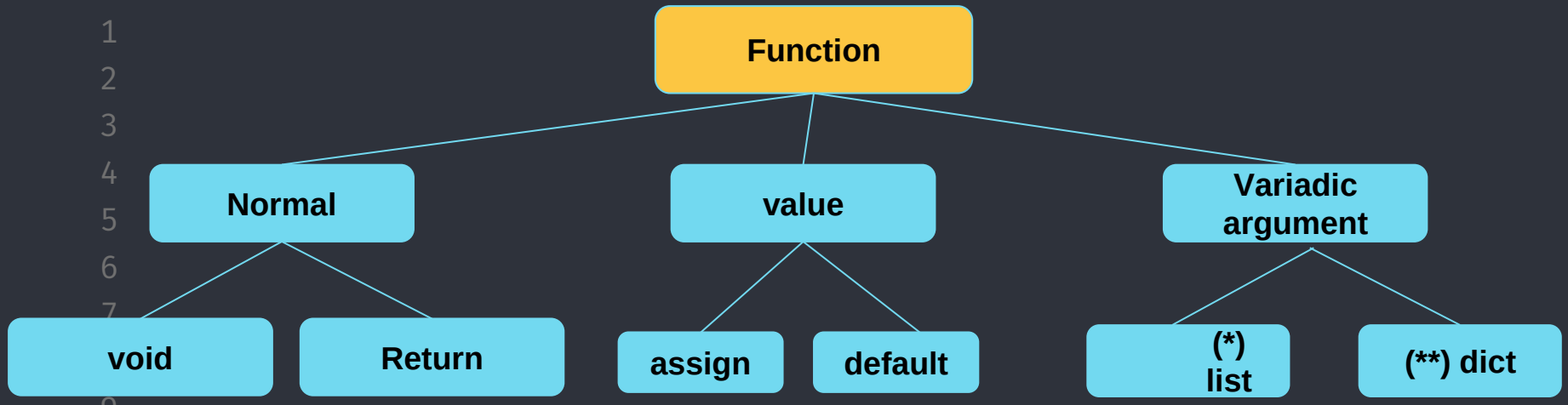
```
>>> dir(str)
['__add__', '__class__', '__contains__', '__delattr__', '__di
_eq__', '__format__', '__ge__', '__getattribute__', '__getite
__', '__gt__', '__hash__', '__init__', '__init_subclass__', '
, '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new
'__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setat
, '__str__', '__subclasshook__', 'capitalize', 'casefold', 'c
ncode', 'endswith', 'expandtabs', 'find', 'format', 'format_m
num', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidenti
isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', '
wer', 'lstrip', 'maketrans', 'partition', 'removeprefix', 're
ce', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rst
itlines', 'startswith', 'strip', 'swapcase', 'title', 'transl
```

```
>>> dir(str)
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewargs__', '__gt__', '__hash__', '__init__', '__i
nit_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__', '__sizeof__', '__s
tr__', '__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', 'removeprefix', 'removesuffix', 'replace', 'rfind'
, 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']
>>> help(str.find)

>>> help(str.find)
```

Documentation






Function
Normal
value
Variadic argument
void
Return
assign
default
(*) list
(**) dict

Functions

A function is a block of code which only runs when it is called.
You can pass data, known as parameters, into a function.
A function can return data as a result.

lab2.py > ...

```python
#Functions
#In Python a function is defined using the def keyword:
def my_function():
  print("Hello from a function")

#To call a function, use the function name followed by parenthesis:
my_function()

################################
def my_function(fname):
  print("My Name is "+fname)

my_function("Muhammad")
```

```
PS D:\Embedded System\Emb
Hello from a function
My Name is Muhammad
```

Return function

```
14  ########################################
15  ##################################
16  def my_function(x):
17      return 5 * x
18
19  print(my_function(3))
20  print(my_function(5))
21  ##################################
22  #function definitions cannot be empty
23  def myfunction():
24      pass
25
```

```
PS D:\Embedd
15
25
```

Assign value to param

```
1
2   ###################################
3   #You can also send arguments with the key = value syntax, order does not matter
4   def my_function(child3, child2, child1):
5     print("The youngest child is " + child3)
6
7   my_function(child1 = "Emil", child2 = "Tobias", child3 = "Linus")
8   ###################################
9
10
11
12
13
14
```

Default value function

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
```

```
57  def my_function(country = "Norway"):
58    print("I am from " + country)
59
60  my_function("Sweden")
61  my_function("India")
62  my_function()
63  ####################################
64  #Time saving, generic function could be used
65  def my_function(arg):
66    for x in arg:
67      print(x)
68
69  mylist = ["apple", "banana", "cherry"]
70  mytuple=(1,2,3)
71  my_function(mylist)
72  my_function(mytuple)
```

Veridic Function and assign parameter

```
1   #and can access the items accordingly:
2   def my_function(*kids):
3     print("The youngest child is " + kids[2])
4
5
6   my_function("Emil", "Tobias", "Linus")
7   ##########################################
8
9
10
11
12
13
14
```

Key and value (dictionary)

```
42  ###################################
43  #This way the function will receive a dictionary of arguments,
44  def my_function(**kid):
45    print("His last name is " + kid["lname"])
46
47  my_function(fname = "Muhammad", lname = "ALi")
48
```

```
49  def func(d):
50      for key in d:
51          print("key:", key, "Value:", d[key])
52
53  D = {'a':1, 'b':2, 'c':3}
54  func(D)
```

Global variable

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
```

```
x=555
def myfunc():
    x=111
    print(id(x))
    print(x)
myfunc()
print(id(x))
print(x)
```

```
x=555
def myfunc():
    global x
    x=111
    print(id(x))
    print(x)
myfunc()
print(id(x))
print(x)
```

Lambda Expression

```
89  ##############################
90  x = lambda a : a + 10
91  print(x(5))
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS D:\Embedded System\Embedded Linux\My
PS D:\Embedded System\Embedded Linux\My
15
PS D:\Embedded System\Embedded Linux\My
```

```
92
93  x = lambda a, b : a * b
94  print(x(5, 6))
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS D:\Embedded System\Embedded Linux\
30
PS D:\Embedded System\Embedded Linux\
```

```
96  print((lambda x, y: x + y)(2, 3))
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS D:\Embedded System\Embedded Linux\My presentatic
5
PS D:\Embedded System\Embedded Linux\My presentatic
```

rust

```
9
1  fn main() {
2      let callback = |x| {
3          println!("hello {}", x);
4      };
5      callback(2);
6  }
7
```

c++

```
11      int main()
12  int main() {
13    int x = 15;
14    auto t = [x](int y, int z) -> void {
15      std::cout << "y= " << y << "z= " << z << std::endl;
16      std::cout << "x= " << x << std::endl;
17    };
18    t(2, 3);
19  }
```

Quick Task

Find the largest item from a given list using loop

```
list1 = [10, 20, 4, 45, 99]
# sorting the list
list1.sort()
# printing the last element
print("Largest element is:", list1[-1])
list1 = [10, 20, 4, 45, 99]
# printing the maximum element
print("Largest element is:", max(list1))
```

```
x = [4, 6, 8, 24, 12, 2]
```

Expected Output:

```
24
```

Write code to find automatically bitcoin rate

Use this API and import requests

https://api.coindesk.com/v1/bpi/currentprice.json

```
99  ##################################################3
100 import requests
101
102 url=requests.get("https://api.coindesk.com/v1/bpi/currentprice.json")
103 print(url.json()['bpi']['USD'])
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS D:\Embedded System\Embedded Linux\My presentation\01python\Session 3> python .\lab2.py
{'code': 'USD', 'symbol': '&#36;', 'rate': '60,596.8850', 'description': 'United States Dollar', 'rate_float': 60596.885}
PS D:\Embedded System\Embedded Linux\My presentation\01python\Session 3>
```

Lab (Add shortcuts)

add_shortcut.py > ...

```python
import keyboard
import subprocess


def on_triggered():
    subprocess.run(['nautilus', '/home/moatsem/'])


def listen_for_shortcut():
    # Set the desired shortcut key combination (Example: Ctrl + Alt + S)
    shortcut = "ctrl + alt + s"

    # Register the callback function for the shortcut
    keyboard.add_hotkey(shortcut, on_triggered)

    # Continuously listen for keyboard events
    keyboard.wait()


# Start listening for the shortcut
listen_for_shortcut()
```

Modules

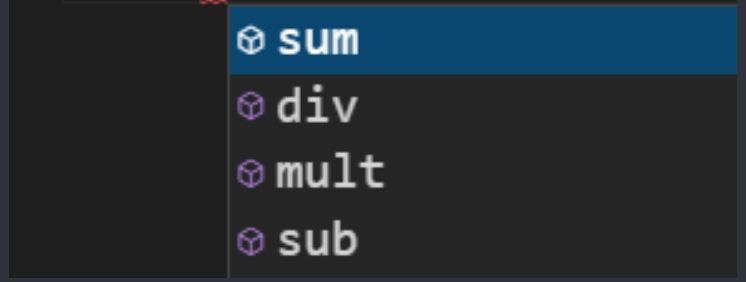
A file containing a set of functions you want to include in your application.

util.py

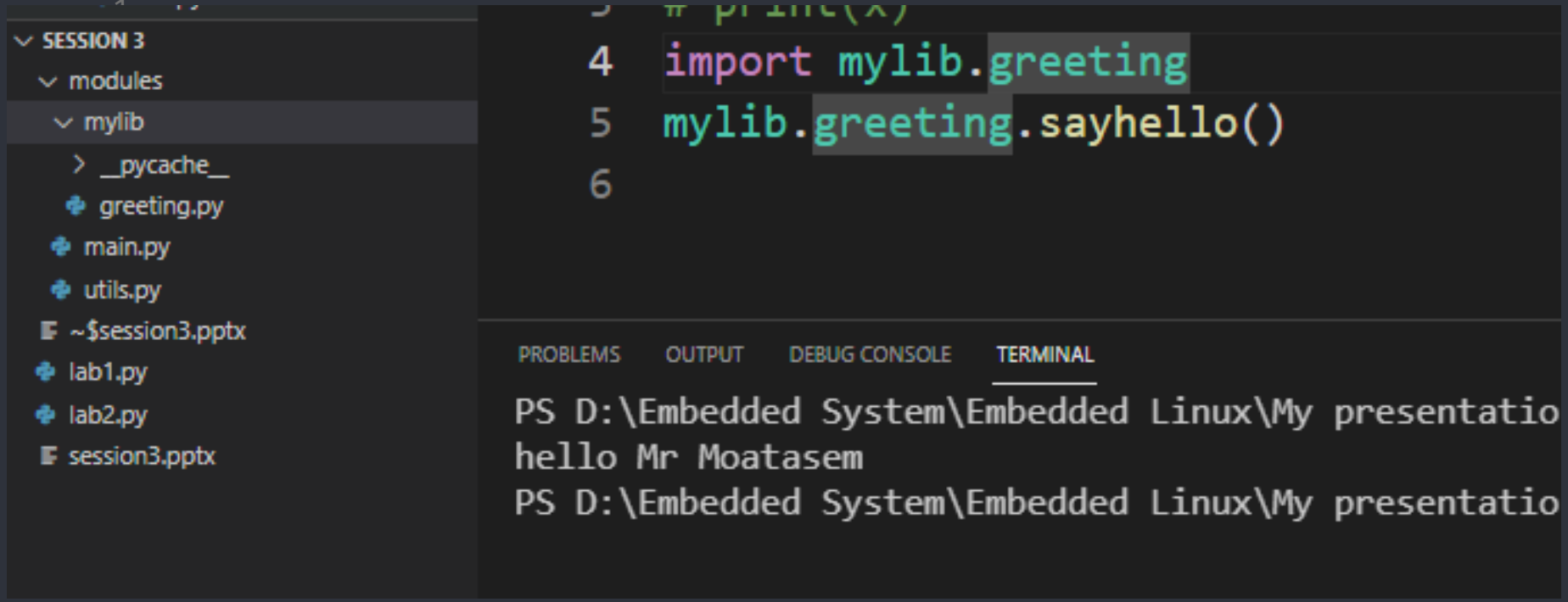
```python
def sum(x,y):
    return x+y
def sub(x,y):
    return x-y
def mult(x,y):
    return x*y
def div(x,y):
    return x/y
```

Main.py

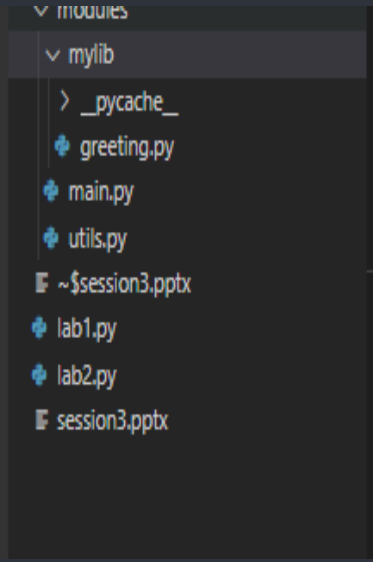
```python
import utils
x=utils.sum(2,3)
print(x)
utils.
```



Modules inside folder



Alias import



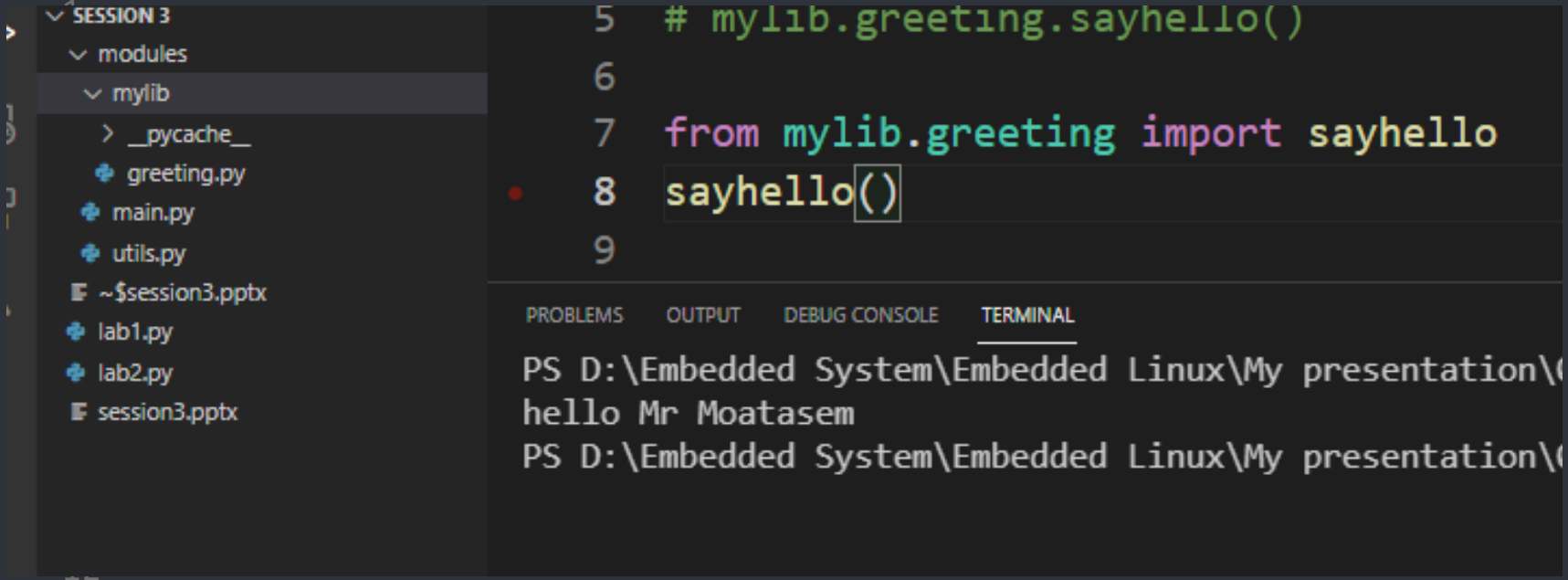

```
10  ##############################################3
11  import mylib.greeting as gt
12  gt.sayhello()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS D:\Embedded System\Embedded Linux\My presentation\01python\Session 3> python .\modules\main.py
hello Mr Moatasem
PS D:\Embedded System\Embedded Linux\My presentation\01python\Session 3>
```

```
11
12
13
14
```

From import



13

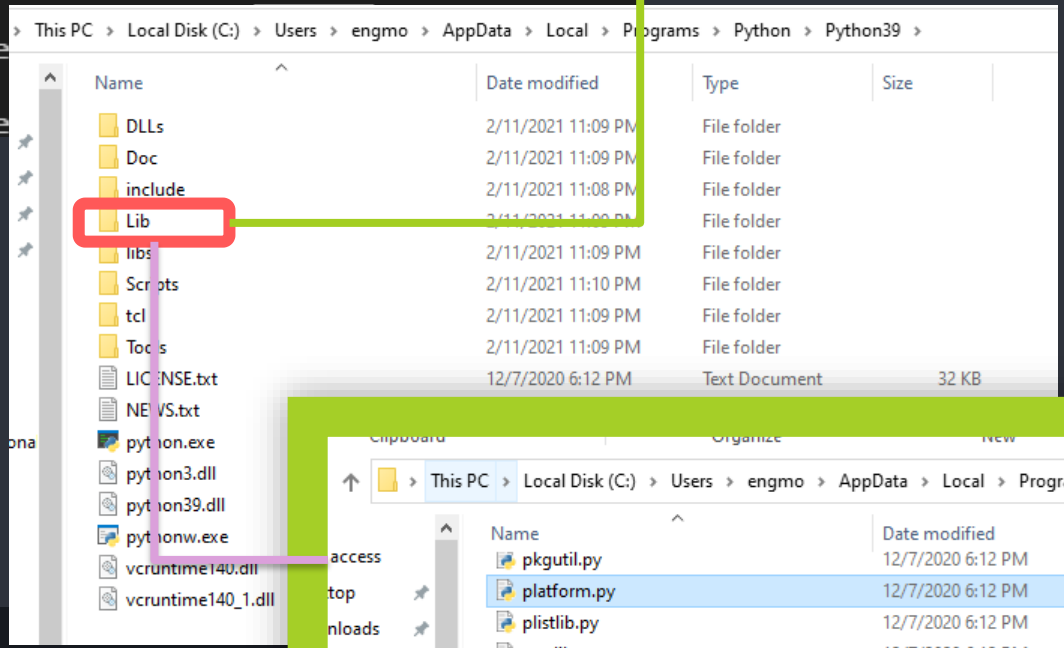
14

Modules built-in

```
14  import platform
15  print(platform.system())
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS D:\Embedde
Windows
PS D:\Embedde
```



Module path from code

```
17 import importlib
18 print("Location of Python os module sources:")
19 print(importlib.import_module('os'))
20
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL powershell

```
PS D:\Embedded System\Embedded Linux\My presentation\01python\Session 3> python .\modules\main.py
Location of Python os module sources:
<module 'os' from 'C:\\Program Files\\WindowsApps\\PythonSoftwareFoundation.Python.3.9_3.9.2032.0_x64__qbz5n2kfra8p0\\lib\\os.py'>
PS D:\Embedded System\Embedded Linux\My presentation\01python\Session 3>
```

```
>> import importlib
>> importlib.import_module("os")
module 'os' from '/usr/lib/python3.10/os.py'>
>>
1]+  Stopped                 python3
oatsem@noatsem-IdeaPad-Gaming-3-15IAH7:~/sigtest/philp_workspace/certs$ ls /usr/lib/python3.10/
bc.py              collections               dis.py                heapq.py          mimetypes.py       posixpath.py      shelve.py          subprocess.py
ifc.py             _collections_abc.py       distutils             hmac.py           modulefinder.py    pprint.py         shlex.py           sunau.py
aix_support.py     colorsys.py               _distutils_system_mod.py html           multiprocessing    profile.py        shutil.py          symtable.py
ntigravity.py      _compat_pickle.py         doctest.py            http              netrc.py           pstats.py         signal.py          _sysconfigdata__lin
rgparse.py         compileall.py             email                 imaplib.py        nntplib.py         pty.py            _sitebuiltins.py   _sysconfigdata__x86
st.py              _compression.py           encodings             imghdr.py         ntpath.py          _py_abc.py        sitecustomize.py   sysconfig.py
synchat.py         concurrent                ensurepip             importlib         nturl2path.py      __pycache__       site.py            tabnanny.py
syncio             config-3.10-x86_64-linux-gnu enum.py            imp.py            numbers.py         pyclbr.py         smtpd.py           tarfile.py
syncore.py         configparser.py           filecmp.py            inspect.py        opcode.py          py_compile.py     smtplib.py         telnetlib.py
ase64.py           contextlib.py             fileinput.py          io.py             operator.py        _pydecimal.py     sndhdr.py          tempfile.py
db.py              contextvars.py            fnmatch.py            ipaddress.py      optparse.py        pydoc_data        socket.py          test
inhex.py           copy.py                   fractions.py          json              os.py              pydoc.py          socketserver.py    textwrap.py
isect.py           copyreg.py                ftplib.py             keyword.py        _osx_support.py    _pyio.py          sqlite3            this.py
bootsubprocess.py  cProfile.py               functools.py          lib2to3           pathlib.py         queue.py          sre_compile.py     _threading_local.py
z2.py              crypt.py                  __future__.py         lib-dynload       pdb.py             quopri.py         sre_constants.py   threading.py
alendar.py         csv.py                    genericpath.py        LICENSE.txt       __phello__.foo.py  random.py         sre_parse.py       timeit.py
gi.py              ctypes                    getopt.py             linecache.py      pickle.py          reprlib.py        ssl.py             tokenize.py
gitb.py            curses                    getpass.py            locale.py         pickletools.py     re.py             statistics.py      token.py
hunk.py            dataclasses.py            gettext.py            logging           pipes.py           rlcompleter.py    stat.py            traceback.py
md.py              datetime.py               glob.py               lzma.py           pkgutil.py         runpy.py          stringprep.py      tracemalloc.py
odecs.py           dbm                       graphlib.py           mailbox.py        platform.py        sched.py          string.py          trace.py
odeop.py           decimal.py                gzip.py               mailcap.py        plistlib.py        secrets.py        _strptime.py       tty.py
ode.py             difflib.py                hashlib.py            _markupbase.py    poplib.py          selectors.py      struct.py          turtle.py
oatsem@noatsem-IdeaPad-Gaming-3-15IAH7:~/sigtest/philp_workspace/certs$
```

Garbage collector

```
11  #diff between list.append / list=
12  ls1=[1,2,3,4]
13  print(id(ls1))
14  ls1.append(5)
15  print(id(ls1))
16
17  ls2=[1,2,3,4]
18  print(id(ls2))
19  ls2=[1,2,3,4,5]
20  print(id(ls2))
```

```
PS D:\Embedded Syst
2810725851264
2810725851264
2810725853312
2810725844032
```

List methods

append()	Adds an element at the end of the list
clear()	Removes all the elements from the list
copy()	Returns a copy of the list
count()	Returns the number of elements with the specified value
extend()	Add the elements of a list (or any iterable), to the end of the current list
index()	Returns the index of the first element with the specified value
insert()	Adds an element at the specified position
pop()	Removes the element at the specified position
remove()	Removes the first item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list

Cont ..

```python
#Lists in python,[],ordered,mutable,changeable,allow duplica
mylist=[1,1.3,"Hi",1]

for i in mylist:
    print(i)
    print(type(i))
##########################
thislist = ["apple", "banana", "cherry", "apple", "cherry"]
print(thislist)
print(len(thislist))
##########################
# The list() Constructor
thislist = list(("apple", "banana", "cherry")) # note the do
print(thislist)
##########################
```

```
PS D:\Embedded System\Embedded Linux\My presentat
1
<class 'int'>
1.3
<class 'float'>
Hi
<class 'str'>
1
<class 'int'>
['apple', 'banana', 'cherry', 'apple', 'cherry']
5
['apple', 'banana', 'cherry']
```

Cont …

```
ab1.py > ...
37  mylist=[10,20,30,40,50,60,70]
38  print(mylist[:])
39  print(mylist[::])
40  print(mylist[::2])
41  print(mylist[1:1])
42  print(mylist[-1:])
43  print(mylist[-1::-1])
44  ########################
45  #check if item in the list
46  thislist = ["apple", "banana", "cherry"]
47  if "apple" in thislist:
48    print("Yes, 'apple' is in the fruits list")
49  ########################
50  thislist = ["apple", "banana", "cherry"]
51  thislist[0]= "x"
52  thislist[1:3] = ["y", "z"]
53  print(thislist)
54  print(len(thislist))
```

```
PS D:\Embedded System\Embedded Linux\
[10, 20, 30, 40, 50, 60, 70]
[10, 20, 30, 40, 50, 60, 70]
[10, 30, 50, 70]
[]
[70]
[70, 60, 50, 40, 30, 20, 10]
Yes, 'apple' is in the fruits list
['x', 'y', 'z']
3
```

Cont …

```
#Sort Ascending
thislist = [100, 50, 65, 82, 23]
thislist.sort()
print(thislist)
#Sort Descending
thislist = [100, 50, 65, 82, 23]
thislist.sort(reverse = True)
print(thislist)
##########################
# nested list
L = ['a', 'b', ['cc', 'dd', ['eee', 'fff']], 'g', 'h']
print(L[2])
print(L[2][2])
print(L[2][2][0])
##############################
L = ['a', ['bb', 'cc'], 'd']
L[1].insert(0,'xx')
print(L)
##############################
```

```
[23, 50, 65, 82, 100]
[100, 82, 65, 50, 23]
['cc', 'dd', ['eee', 'fff']]
['eee', 'fff']
eee
['a', ['xx', 'bb', 'cc'], 'd']
```

Cont …

```
75  L = ['a', ['bb', 'cc', 'dd'], 'e']
76  del L[1][1]
77  print(L)
78  ########################################3
79  L = ['a', ['bb', 'cc', 'dd'], 'e']
80  L[1].remove('cc')
81  print(L)
82  print(len(L))
83  ########################################3
84  # using lists as stack
85  stack = [3, 4, 5]
86  stack.append(6)
87  stack.append(7)
88  print(stack)
89  stack.pop()
90  stack.pop()
```

```
PS D:\Embedded System\Embedde
['a', ['bb', 'dd'], 'e']
['a', ['bb', 'dd'], 'e']
3
[3, 4, 5, 6, 7]
[3, 4, 5]
```

Cont …

```
94  fruits = ['apple', 'banana', 'cherry']
95  cars = ['Ford', 'BMW', 'Volvo']
96  fruits.extend(cars)
97  print(fruits)
98  ##########################################
99  fruits = ['apple', 'banana', 'cherry']
100 indx = fruits.index("cherry")
101 print(indx)
102 ##########################################
103 fruits = ['apple', 'banana', 'cherry']
104 x = fruits.count("cherry")
105 print(x)
106 ##########################################33
107 fruits = ['apple', 'banana', 'cherry', 'orange']
108 y=fruits
109 x = fruits.copy()
110 print(x)
111 print(id(x))
112 print(id(y))
113 print(id(fruits))
```

```
['apple', 'banana', 'cherry', 'Ford', 'BMW', 'Volvo']
2
1
['apple', 'banana', 'cherry', 'orange']
2122216956288
2122216948224
2122216948224
```

Tuple Method

```
121 # Tuples are unchangeable, meaning that we cannot change, add or remove items afte
122 thistuple = ("apple", "banana", "cherry")
123 print(len(thistuple))
124 # ##########################
125 thistuple = ("apple",)
126 print(type(thistuple))
127
128 thistuple = ("apple") #what's the differ
129 print(type(thistuple))
130
131 # ##########################
132 tuple1 = ("abc", 34, True, 40, "male")
133 print(tuple1)
134 # ##########################
135 # tuple() constructor to make a tuple.
136 thistuple = tuple(("apple", "banana", "cherry")) # note the double round-brackets
137 print(thistuple)
```

```
PS D:\Embedded System\Embedded Linux\My pre
3
<class 'tuple'>
<class 'str'>
('abc', 34, True, 40, 'male')
('apple', 'banana', 'cherry')
PS D:\Embedded System\Embedded Linux\My pre
```

cont..

```
140 thistuple = ("A", "B", "C", "D", "E", "F")
141 print(thistuple[-1])
142 print(thistuple[:])
143 print(thistuple[::])
144 print(thistuple[::2])
145 print(thistuple[1:1])
146 print(thistuple[-1:])
147 print(thistuple[-1::-1])
148 print(thistuple[-4:-1])
149
150 thistuple = ("apple", "banana", "cherry")
151 if "apple" in thistuple:
152     print("Yes, 'apple' is in the fruits tuple")
153
154 thistuple = ("apple", "banana", "cherry")
155 thistuple.append("orange") # This will raise an error
156 print(thistuple)
```

```
F
('A', 'B', 'C', 'D', 'E', 'F')
('A', 'B', 'C', 'D', 'E', 'F')
('A', 'C', 'E')
()
('F',)
('F', 'E', 'D', 'C', 'B', 'A')
('C', 'D', 'E')
Yes, 'apple' is in the fruits tuple
Traceback (most recent call last):
  File "D:\Embedded System\Embedded Linux\My presentation\01python\Session 4\lab
    thistuple.append("orange") # This will raise an error
AttributeError: 'tuple' object has no attribute 'append'
PS D:\Embedded System\Embedded Linux\My presentation\01python\Session 4>
```

cont..

```
60  fruits = ("apple", "banana", "cherry", "strawberry", "raspberry")
61  (green, yellow, *red) = fruits
62  print(green)
63  print(yellow)
64  print(type(yellow))
65  print(red)
66  print(type(red))
67  ########################
68  # If the asterix is added to another variable name thar
69  # Python will assign values to the variable until the number of va
70  fruits = ("apple", "mango", "papaya", "pineapple", "cherry")
71  (green, *tropic, red) = fruits
72  print(green)
73  print(tropic)
74  print(red)
```

```
apple
banana
<class 'str'>
['cherry', 'strawberry', 'raspberry']
<class 'list'>
apple
['mango', 'papaya', 'pineapple']
cherry
```

cont..

```python
#Join Two Tuples
tuple1 = ("a", "b" , "c",1.1,12,[1,23,9])
tuple2 = (1, 2, 3)
tuple3 = tuple1 + tuple2
print(tuple3)
##################################################
fruits = ("apple", "banana", "cherry")
mytuple = fruits * 2
print(mytuple)
##################################################
thistuple = (1, 3, 7, 8, 7, 5, 4, 6, 8, 5)
x = thistuple.count(5)
print(x)
##################################################
thistuple = (1, 3, 7, 8, 7, 5, 4, 6, 8, 5)
x = thistuple.index(8)
print(x)
```

Method	Description
count()	Returns the number of times a specified value occurs in a tuple
index()	Searches the tuple for a specified value and returns the position of where it was found

```
PS D:\Embedded System\Embedded Linux\My presentation\01python\Se
('a', 'b', 'c', 1.1, 12, [1, 23, 9], 1, 2, 3)
('apple', 'banana', 'cherry', 'apple', 'banana', 'cherry')
2
3
PS D:\Embedded System\Embedded Linux\My presentation\01python\Se
```

set Method

Method	Description
append()	Adds an element at the end of the list
clear()	Removes all the elements from the list
copy()	Returns a copy of the list
count()	Returns the number of elements with the specified value
extend()	Add the elements of a list (or any iterable), to the end of the current list
index()	Returns the index of the first element with the specified value
insert()	Adds an element at the specified position
pop()	Removes the element at the specified position
remove()	Removes the first item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list

cont..

```python
##############################################
thisset = {"apple", "banana", "cherry"}
print(thisset)
print(len(thisset))
##############################################
#loop in list
thisset = {"apple", "banana", "cherry"}
for x in thisset:
  print(x)
##############################################
#find element in set
thisset = {"apple", "banana", "cherry"}
print("banana" in thisset)
##############################################
#update list with only one element
thisset = {"apple", "banana", "cherry"}
thisset.add("orange")
print(thisset)
##############################################
#update list with more than one element
thisset = {"apple", "banana", "cherry"}
thisset.update(["orange", "mango", "grapes"])
print(thisset)
##############################################
```

```
PS D:\Embedded System\Embedded Linux\My presentation\01pyth
{'banana', 'cherry', 'apple'}
3
banana
cherry
apple
True
{'banana', 'cherry', 'apple', 'orange'}
{'banana', 'apple', 'cherry', 'grapes', 'mango', 'orange'}
```

Cont ..

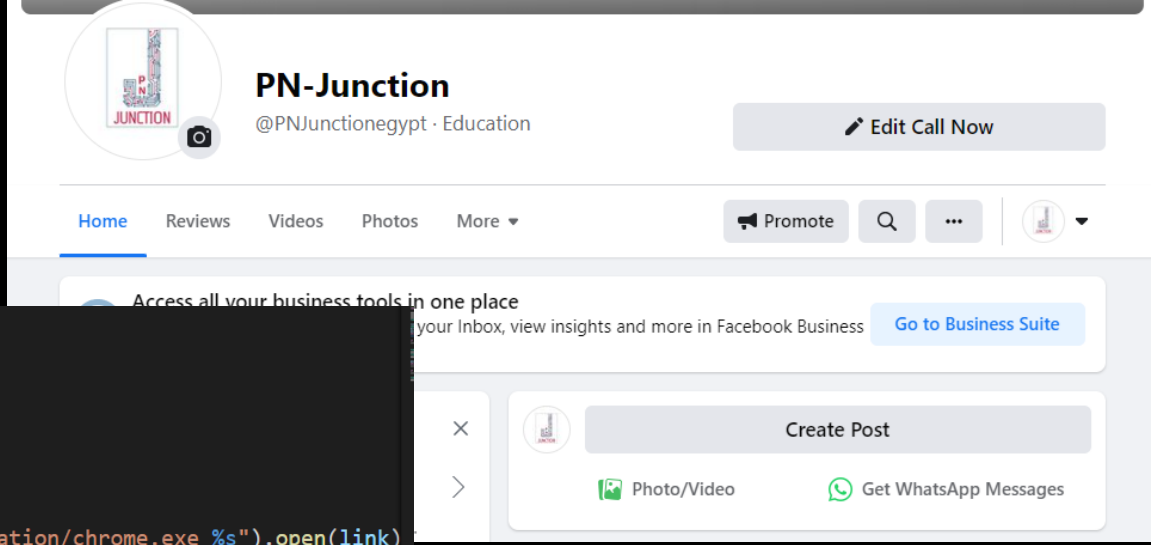
```
#remove element from list
thisset = {"apple", "banana", "cherry"}
thisset.remove("banana")
print(thisset)
##############################################
thisset = {"apple", "banana", "cherry"}
x = thisset.pop()
print(x)
print(thisset)
##############################################
# Return a set that contains the items that o
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
z = x.difference(y)
print(z)
##############################################
# The intersection() method returns a set tha
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
z = x.intersection(y)
print(z)
```

```
PS D:\Embedded Systen
{'apple', 'cherry'}
apple
{'cherry', 'banana'}
{'banana', 'cherry'}
{'apple'}
PS D:\Embedded Systen
```

pyautoGui

```
1
2
3
```

```
6
7  pyautogui.hotkey('win')
8  pyautogui.typewrite("goo")
9  pyautogui.hotkey('enter')
10 time.sleep(2)
11
12 link="https://www.facebook.com/PNJunctionegypt"
13 webbrowser.get("C:/Program Files/Google/Chrome/Application/chrome.exe %s").open(link)
14 dir_path = os.path.dirname(os.path.realpath(__file__))
15 pointxy=None
16 while pointxy is None  :
17     pointxy=pyautogui.locateOnScreen(dir_path+"\\createpost.png",confidence=.7)
18
19 if pointxy is not None:
20         pyautogui.moveTo(pointxy[0]+150,pointxy[1]+20,duration=0.5)
21         time.sleep(0.1)
22
23 time.sleep(2)
24 pyautogui.click()
25 time.sleep(4)
26 pyautogui.write("Embedded Diploma Soon stay Tunned..")
```



Task

1-Make your module that contain favourite websites and have function called Firefox take url and open website
2- then make main file and print menu of sites for user and let him choice

```python
import firelink

firelink.firefox(firelink.facebook_link)
while True:
    pass
```

Hint : use import webbrowser

Tasks

https://www.boredapi.com/api/activity

Write a code to suggest automatically activates for you

Get your public IP https://api.ipify.org/?format=json

Task PyAutoGUI

```
1
2   Using PyAutoGUI
3   - To open vscode
4   - install clangd from extension
5   - install c++ testmate  from extension
6   - install c++ helper  from extension
7   - install cmake  from extension
8   - install cmake tools  from extension
9
10
11
12
13
14
```