jupyter day-5 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3.6.6 64-bit

PRODDEC PYTHON DAY 5

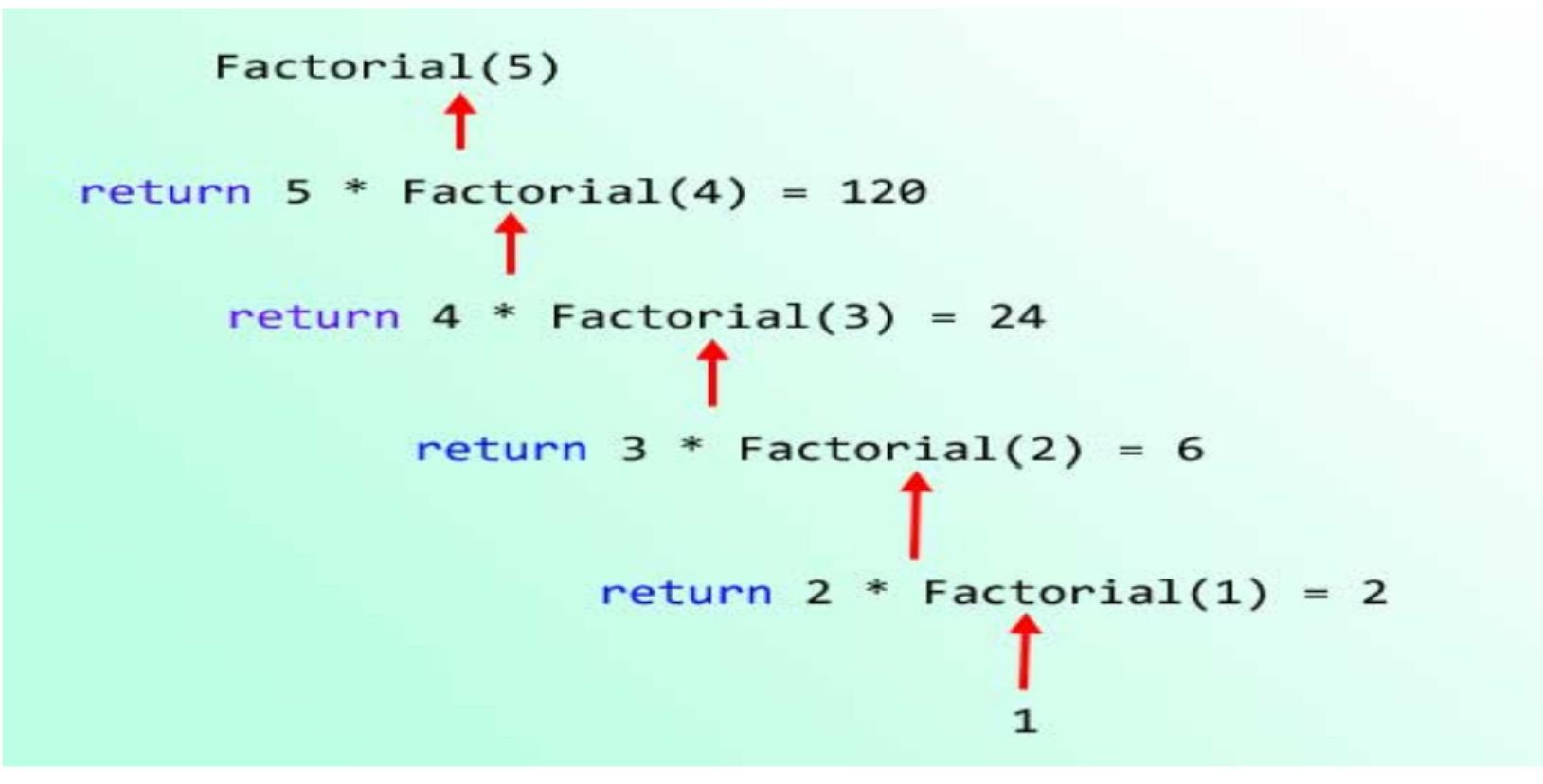
Function, Recursion and Return - Working

In [42]:
```python
# To find the factorial of the 5.
def Factorial(n):
    if n==1:
        return 1
    else:
        return n*Factorial(n-1)

print(Factorial(5))
```

```
120
```

Working of recursion.



List comprehension

In [19]:
```python
# List comprehensions provide a concise way to create lists.
# Little line of code.
# Complicated to understand.
```

Without List comprehension

In [43]:
```python
# For adding all natural number (1-20) in a list.
naturalNumber=[]
for i in range(1,21):
    naturalNumber.append(i)
print(naturalNumber)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
```

With List comprehension

In [44]:
```python
naturalNum=[i for i in range(1,21)] # List comprehension.
print(naturalNum)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
```

Using if condition in List comprehension

In [45]:
```python
evenList=[i for i in range(2,20) if i%2==0] # List comprehension.
print(evenList)
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18]
```

Using if and else in List comprehension

In [46]:
```python
# I want to create a new list based on the following criteria.
# if the value in the myList is less than 40 -> add 0
# if the value in the myList is greater than 40 -> add 1

myList=[12,50,48,5,4,6,2,78,45,13,74]
newList=[0 if i<40 else 1 for i in myList ]
print(newList)
```

```
[0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1]
```

Path

In [ ]:
```python
# My Definition.
# Telling someone, where to go.
# Relative path -> where to go from your current position.
# Absolute path -> where to go from starting position.
```

Relative path and Absolute path

In [ ]:
```python
# I am currently in the bacon folder.
# Actual Starting position -> C:\
# dot -> current folder.
# dot dot -> previous folder.
```

Import statement

In [ ]:
```python
"""
Python import statement enables the user to import
particular modules in the corresponding program.
"""
```

In [47]:
```python
# my_functions.py -> This is the name of this file.
# These are the content in this file.
def sayHello():
    return "Hello Sir"
def cube(num):
    return num*num*num
def root(number):
    return number**0.5

# The below codes run when this module is imported.

print(sayHello())

print("This is my program.")
print("Dont copy my function.")
print("you can import it.")

print(cube(3))
print(root(2))
```

```
Hello Sir
This is my program.
Dont copy my function.
you can import it.
27
1.4142135623730951
```

name variable

In [ ]:
```python
"""
I have two python file -> first.py and second.py

when first.py is executed,
the value of the __name__ variables in first.py is  "__main__"

when the first.py is imported into second.py,
the value of the __name__ in first.py is "first".
"""
```

Best method to create a module

In [ ]:
```python
# my_function.py -> This is the name of this file.
# These are the content in this file.
def sayHello():
    return "Hello Sir"
def cube(num):
    return num*num*num
def root(number):
    return number**0.5

# The below codes doesn't run when this module is imported.
# But run, when this file is executed.

if __name__=="__main__":
    print(sayHello())

    print("This is my program.")
    print("Dont copy my function.")
    print("you can import it.")

    print(cube(3))
    print(root(2))
```

Simple import statement

In [1]:
```python
import my_function

print(my_function.sayHello())
print(my_function.cube(20))
print(my_function.root(50))
```

```
Hello Sir
This is my program.
Dont copy my function.
you can import it.
27
1.4142135623730951
Hello Sir
8000
7.0710678118654755
```

as keyword

In [48]:
```python
# as keyword help to use simple word instead of long name.
import my_function as mf

print(mf.sayHello())
print(mf.cube(20))
print(mf.root(50))
```

```
Hello Sir
8000
7.0710678118654755
```

from and * keyword

In [49]:
```python
# * means all
from my_function import *

print(sayHello())
print(cube(20))
print(root(50))
```

```
Hello Sir
8000
7.0710678118654755
```

Importing only necessary things

In [50]:
```python
# I didn't import the sayHello() function.
# I don't need that fuction here.

from my_function import root,cube
print(cube(20))
print(root(50))
```

```
8000
7.0710678118654755
```

Importing only necessary things and as keyword

In [51]:
```python
from my_function import root as rt,cube as cb
print(cb(20))
print(rt(50))
```

```
8000
7.0710678118654755
```

module

In [ ]:
```python
# A python script can be called as a module.
```

user defined modules

In [ ]:
```python
# module created by user.
```

bultin module

In [ ]:
```python
# all modules that comes with the python installation.
```

Some builtins module examples

random module

In [52]:
```python
# Printing all functions in the random module.
# Dont think too much! Search google for more details.
import random
print(dir(random))
```

```
['BPF', 'LOG4', 'NV_MAGICCONST', 'RECIP_BPF', 'Random', 'SG_MAGICCONST', 'Syste
mRandom', 'TWOPI', '_BuiltinMethodType', '_MethodType', '_Sequence', '_Set', '_
_all__', '__builtins__', '__cached__', '__doc__', '__file__', '__loader__', '__
name__', '__package__', '__spec__', '_acos', '_bisect', '_ceil', '_cos', '_e',
'_exp', '_inst', '_itertools', '_log', '_pi', '_random', '_sha512', '_sin', '_s
qrt', '_test', '_test_generator', '_urandom', '_warn', 'betavariate', 'choice',
'choices', 'expovariate', 'gammavariate', 'gauss', 'getrandbits', 'getstate',
'lognormvariate', 'normalvariate', 'paretovariate', 'randint', 'random', 'randr
ange', 'sample', 'seed', 'setstate', 'shuffle', 'triangular', 'uniform', 'vonmi
sesvariate', 'weibullvariate']
```

In [53]:
```python
# This module is related with random number.
import random

# Generating 5 random number
for i in range(5):
    print(random.randint(1,101))
```

```
94
3
79
54
16
```

In [56]:
```python
# Getting a random item from the list.
import random
car=["maruti","honda","fiat","bmw","jaguar","suzuki","alto","nano"]
for i in range(5):
    print(car[random.randint(0,7)]) # using slice and randint
```

```
suzuki
suzuki
nano
bmw
bmw
```

In [57]:
```python
# Selecting 5 random item from the list
car=["maruti","honda","fiat","bmw","jaguar","suzuki","alto","nano"]

for i in range(5):
    print(random.choice(car))
```

```
nano
nano
suzuki
maruti
jaguar
```

In [59]:
```python
# Shuffling a list 5 times and printing the list.

car=["maruti","honda","fiat","bmw","jaguar","suzuki","alto","nano"]
for i in range(1,6):
    random.shuffle(car)
    print(i,car)
```

```
1 ['nano', 'honda', 'fiat', 'alto', 'suzuki', 'jaguar', 'bmw', 'maruti']
2 ['bmw', 'fiat', 'nano', 'jaguar', 'maruti', 'honda', 'suzuki', 'alto']
3 ['nano', 'honda', 'suzuki', 'fiat', 'jaguar', 'maruti', 'bmw', 'alto']
4 ['bmw', 'suzuki', 'maruti', 'nano', 'alto', 'fiat', 'jaguar', 'honda']
5 ['bmw', 'nano', 'alto', 'jaguar', 'suzuki', 'maruti', 'fiat', 'honda']
```

math module

In [11]:
```python
# printing all function in the math module
import math
print(dir(math))
```

```
['__doc__', '__loader__', '__name__', '__package__', '__spec__', 'acos', 'acos
h', 'asin', 'asinh', 'atan', 'atan2', 'atanh', 'ceil', 'copysign', 'cos', 'cos
h', 'degrees', 'e', 'erf', 'erfc', 'exp', 'expm1', 'fabs', 'factorial', 'floo
r', 'fmod', 'frexp', 'fsum', 'gamma', 'gcd', 'hypot', 'inf', 'isclose', 'isfini
te', 'isinf', 'isnan', 'ldexp', 'lgamma', 'log', 'log10', 'log1p', 'log2', 'mod
f', 'nan', 'pi', 'pow', 'radians', 'sin', 'sinh', 'sqrt', 'tan', 'tanh', 'tau',
'trunc']
```

Some functions in the math module

In [60]:
```python
print(math.sqrt(36)) # printing the root of 36.
print(math.tan(55)) # Printing the tan value of 55.
print(math.pow(2,3)) # Finding the 2^3
print(math.factorial(20)) # Print the factorial of 100
```

```
6.0
-45.18308791052113
8.0
2432902008176640000
```

Some variables in the math module

In [61]:
```python
# These are not functions, these are variables in the math module
# NOTE -> () is not added at the end.
print(math.pi) # print the value of the pi.
print(math.e)  # print the value of e
```

```
3.141592653589793
2.718281828459045
```

Using the help function

In [62]:
```python
# I want to know about the gcd function in the math module.

print(help(math.gcd))  # To know how gcd works

# Same as finding the HCF of 2 numbers
# HCF -> Highest common factor.
```

```
Help on built-in function gcd in module math:

gcd(...)
    gcd(x, y) -> int
    greatest common divisor of x and y

None
```

In [18]:
```python
# Example
import math
print(math.gcd(12,9)) # Find the HCF of 9 and 12
```

```
3
```

Builtin modules and user defined modules

In [30]:
```python
# Modules available with python installations-> Builtin module
# Modules created by user -> user defined modules.
```

time module

In [33]:
```python
# Printing all function in the time module.
import time
print(dir(time))
```

```
['_STRUCT_TM_ITEMS', '__doc__', '__loader__', '__name__', '__package__', '__spe
c__', 'altzone', 'asctime', 'clock', 'ctime', 'daylight', 'get_clock_info', 'gm
time', 'localtime', 'mktime', 'monotonic', 'perf_counter', 'process_time', 'sle
ep', 'strftime', 'strptime', 'struct_time', 'time', 'timezone', 'tzname']
```

time.time()

In [ ]:
```python
"""
time.time() method of Time module is used to get the time in seconds
since epoch. The handling of leap seconds is platform dependent.

Note: The epoch is the point where the time starts, and is platform
dependent. On Windows and most Unix systems,
the epoch is January 1, 1970, 00:00:00 (UTC)
and leap seconds are not counted towards the time in seconds
since the epoch. """
```

In [63]:
```python
print(time.time()) # total time elapsed in float, after January 1, 1970
print("Waiting")
time.sleep(5) # Program sleep for 5 second.
print(time.time())
```

```
1595256277.3167164
Waiting
1595256282.3168912
```

Calculating time taken to execute a program

In [64]:
```python
import time
starting_time=time.time()
for i in range(20):
    print("HELLO".title())
ending_time=time.time()

print("Total time taken:",ending_time-starting_time)
```

```
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Total time taken: 0.0019979476928710938
```

builtins modules

In [32]:
```python
# Ready to use functions are available from builtins module.
# This module is already imported when you execute a python script.
# Printing functions in the builtins module.
import builtins

print(dir(builtins))
# note down->len,print,pow,max,str,int,list,min,range...
```

```
['ArithmeticError', 'AssertionError', 'AttributeError', 'BaseException', 'Block
ingIOError', 'BrokenPipeError', 'BufferError', 'BytesWarning', 'ChildProcessErr
or', 'ConnectionAbortedError', 'ConnectionError', 'ConnectionRefusedError', 'Co
nnectionResetError', 'DeprecationWarning', 'EOFError', 'Ellipsis', 'Environment
Error', 'Exception', 'False', 'FileExistsError', 'FileNotFoundError', 'Floating
PointError', 'FutureWarning', 'GeneratorExit', 'IOError', 'ImportError', 'Impor
tWarning', 'IndentationError', 'IndexError', 'InterruptedError', 'IsADirectoryE
rror', 'KeyError', 'KeyboardInterrupt', 'LookupError', 'MemoryError', 'ModuleNo
tFoundError', 'NameError', 'None', 'NotADirectoryError', 'NotImplemented', 'Not
ImplementedError', 'OSError', 'OverflowError', 'PendingDeprecationWarning', 'Pe
rmissionError', 'ProcessLookupError', 'RecursionError', 'ReferenceError', 'Reso
urceWarning', 'RuntimeError', 'RuntimeWarning', 'StopAsyncIteration', 'StopIter
ation', 'SyntaxError', 'SyntaxWarning', 'SystemError', 'SystemExit', 'TabErro
r', 'TimeoutError', 'True', 'TypeError', 'UnboundLocalError', 'UnicodeDecodeErr
or', 'UnicodeEncodeError', 'UnicodeError', 'UnicodeTranslateError', 'UnicodeWar
ning', 'UserWarning', 'ValueError', 'Warning', 'WindowsError', 'ZeroDivisionErr
or', '__IPYTHON__', '__build_class__', '__debug__', '__doc__', '__import__', '_
_loader__', '__name__', '__package__', '__spec__', 'abs', 'all', 'any', 'asci
i', 'bin', 'bool', 'bytearray', 'bytes', 'callable', 'chr', 'classmethod', 'com
pile', 'complex', 'copyright', 'credits', 'delattr', 'dict', 'dir', 'display',
'divmod', 'enumerate', 'eval', 'exec', 'filter', 'float', 'format', 'frozense
t', 'get_ipython', 'getattr', 'globals', 'hasattr', 'hash', 'help', 'hex', 'i
d', 'input', 'int', 'isinstance', 'issubclass', 'iter', 'len', 'license', 'lis
t', 'locals', 'map', 'max', 'memoryview', 'min', 'next', 'object', 'oct', 'ope
n', 'ord', 'pow', 'print', 'property', 'range', 'repr', 'reversed', 'round', 's
et', 'setattr', 'slice', 'sorted', 'staticmethod', 'str', 'sum', 'super', 'tupl
e', 'type', 'vars', 'zip']
```

In [ ]:

In [ ]: