Hello World!

print Hello World!	`print("Hello World!")`

print with a variable	`x= "Hello World!"`
	`print(x)`
	Output:
	`Hello World!`

Python Variables

local scope	contains inside of a function
	example: **x, y**
	`def newfunc():`
	`  x = 5`
	`  y = 10`
	`  print(x+y)`
	`newfunc()`

global scope	contains all python code
	examle: **z**
	`z = 50`
	`def myfunc():`
	`  x = 300`

Python Comments

using #	`print("Hello!")  #This is print function`
	`a= 5*5  #This is a calculation`
	`print(a)  #This is print function`
	Outout:
	`Hello!`
	`25`

using ""”	`"""`
	`This is only a comment!`
	`"""`
	`print("Hello!")`
	Output:
	`Hello!`

User Input

user input	`user = input("What is Your Username:")`
	`pwd = input("What is Your Password:")`
	`print("Username is: " + user)`
	`print("Password is: " + pwd)`
	Output:
	`Username is: IPCisco`
	`Password is: xyz`

```
myfunc()
print(x+z)
```

printing variables

```
a = 5
b = "John"
abc = {1,2,3,4,5}
mylist = ["x","yy","zzz"]

print(a)
print(b)
print(abc)
print(mylist)
```

Output:

```
5
John
{1, 2, 3, 4, 5}
['x', 'yy', 'zzz']
```

variable class

```
a = 5
b = "Cisco"
abc = {1,2,3,4,5}
mylist = ["router","switch","firewall"]

print(type(a))
print(type(b))
print(type(abc))
print(type(mylist))
```

Output:

```
<class 'int'>
<class 'str'>
<class 'set'>
<class 'list'>
```

user input and calculation

```
number = input("Please Enter A Number:")
number=int(number)
Square = number * number
print("The square of number:{}".format(Square))
```

Output:

```
Please Enter A Number: 5
The square of number:25
```


Python Strings

python string

```
"Hello"
```

string length

```
message = "hello,how are you?"
print(len(message))
```

Output:

```
18
```

using format

```
number1 = 2
number2 = 1
message = "I have {} cats and {} dog."
print(message.format(number1, number2))
```

Output:

```
I have 2 cats and 1 dog.
```

string split

```
message = 'life is good'
print(message.split())
```

Output:

```
['life', 'is', 'good']
```

string replace

```
msg = "I have two dogs, two cats."
newmsg = msg.replace("three", "two")
print(newmsg)
```

Output:

Python Lists

number list

```
numbers = [1,10,8,3,15]
```

string list

```
cars = ["Porche", "Aston Martin", "Ferrari"]
```

list length

```
cars = ["Porche", "Aston Martin", "Ferrari"]
length = len(cars)
print(length)
```

Output:

```
3
```

adding item

```
elves = ["Legolas", "Elrond"]
elves.append("Arwen")
print(elves)
```

Output:

```
["Legolas", "Elrond", "Arwen"]
```

removing item

```
list = [6, 15, 24, 36, 55]
list.remove(24)
print(list)
```

Output:

```
[6, 15, 36, 55]
```

sorting list

```
list = [85, 9, 3, 15, 75, 23]
list.sort()
print(list)
```

Output:

```
[3,9,15,23,75,85]
```

reverse sort

```
list = [85, 9, 3, 15, 75, 23]
list.sort(reverse=True)
print(list)
```

Output:

```
[85,75,23,15,9,3]
```

pop item

```
numbers = [14,26,45,76,95]
numbers.pop(3)
print(numbers)
```

Output:

```
[14, 26, 45, 95]
```

reverse list

```
numbers = [1,2,3,4,5]
numbers.reverse()
print(numbers)
```

```
I have three dogs, three cats.
```

string contains

```
msg = "Istanbul is in Turkey."
print(msg.find("Turkey"))
```

Output:

```
15
```

string find

```
msg = "I like cats."
print(msg.find("cats"))
```

Output:

```
7
```

string slice

```
x = "Hello World!"
print(x[2])
```

Output:

```
l
```

string slice

```
x = "Hello World!"
print(x[3:5])
```

Output:

```
lo
```

concatenation

```
print ("Hello" + "Python" + "World")
```

Output:

```
HelloPythonWorld
```

capitalize

```
msg = "welcome to python course!"
x = msg.capitalize()
print (x)
```

Output:

```
Welcome to python course!
```

	Output: `[5, 4, 3, 2, 1]`

comprehension	`list1 = ["152", "4", "98", "169", "16"]`
	`list2 = [x for x in list1 if "9" in x]`
	`print(list2)`
	Output:
	`['98', '169']`

finding Index	`numbers = [5,9,45,7,126,4]`
	`x = numbers.index(45)`
	`print(x)`
	Output:
	`2`

Python Tuples

number tuple	`tuple1 = (1, 2, 3)`
string tuple	`tuple2 = ("John", "Elena", "Albert")`

finding tuple member	`tuple1 = ("John", "Elena", "Albert")`
	`print(tuple1[1])`
	Output:
	`Elena`

printing tuple isle	`tuple1 = (1, 2, 3, 4, 5, 6)`
	`print(tuple1[2:4])`
	Output:
	`(3, 4)`

title	`msg = "welcome to python course!"`
	`x = msg.title()`
	`print (x)`
	Output:
	`Welcome To Python Course!`

lower	`msg = "weLcoMe to PytHon CouRse!"`
	`x = msg.lower()`
	`print (x)`
	Output:
	`welcome to python course!`

upper	`msg = "weLcoMe to PytHon CouRse!"`
	`x = msg.upper()`
	`print (x)`
	Output:
	`WELCOME TO PYTHON COURSE!`

count	`msg = "I like cats. Because cats are cute."`
	`x = msg.count("cats")`
	`print(x)`
	Output:
	`2`

join	`characters = ["Aragorn", "Arwen", "Legolas"]`
	`print ("-".join(characters))`
	Output:
	`Aragorn-Arwen-Legolas`

partition	`msg = "welcome to python course"`
	`x = msg.partition("to")`
	`print(x)`
	Output:

Counting members	`numbers = (10,15,24,10,29,34,45,10)`
	`x = numbers.count(10)`
	`print(x)`
	Output:
	`3`

Tuple length	`numbers = (10,15,24,10,29,34,45,10)`
	`x =len(numbers)`
	`print(x)`
	Output:
	`8`

Couting an item	`numbers = (10,5,14,25,38,10,62,47,10)`
	`x = numbers.count(10)`
	`print(x)`
	Output:
	`3`

for loop with tuple	`cars = ("bmw", "audi", "volvo", "mercedes")`
	`for x in cars:`
	`  print(x)`
	Output:
	`bmw`
	`audi`
	`volvo`
	`mercedes`

Python Sets

string set	`dwarves = {"Thorin", "Balin", "Dwalin"}`
number set	`numbers={23,5,2,14,32,18,45}`
mixed set	

	`('welcome ', 'to', ' python course')`

endswith	`msg = "weLcoMe to PytHon CouRse!"`
	`x = msg.endswith("!")`
	Output:
	`True`

Python Dictionaries

python dictionary	`character ={`
	`  "race": "wizard",`
	`  "name": "Gandalf",`
	`  "color": ["grey", "white"]`
	`  }`

dictionary length	`character ={`
	`  "race":"wizard",`
	`  "name":"Gandalf",`
	`  "color":["grey", "white"]`
	`}`
	`print(len(character))`
	Output:
	`3`

access item	`character ={`
	`  "race":"wizard",`
	`  "name":"Gandalf",`
	`  "color":"grey",`
	`}`
	`print(character["name"])`
	Output:
	`Gandalf`

access item (with get)	`device ={`
	`  "vendor": "Cisco",`
	`  "model": "9000",`
	`  "RU": 44`
	`}`
	`print(device.get("model"))`

	items={True, 5, 2.7, "Gimli", False, 34}
union set	dwarves1 = {"Thorin", "Balin", "Dwalin"} dwarves2 = {"Gimli", "Bofur", "Thorin"} print(dwarves1 | dwarves2) **Output:** {'Balin', 'Dwalin', 'Bofur', 'Thorin', 'Gimli'}
adding item	dwarves = {"Thorin", "Balin", "Dwalin"} dwarves.add("Gimli") print(dwarves) **Output:** {'Balin', 'Gimli', 'Thorin', 'Dwalin'}
removing item	dwarves = {"Thorin", "Balin", "Dwalin"} dwarves.remove("Thorin") print(dwarves) **Output:** {'Balin', 'Dwalin'}
clear set	dwarves = {"Thorin", "Balin", "Dwalin"} dwarves.clear() print(dwarves) **Output:** set()
finding difference	dwarves1 = {"Thorin", "Balin", "Dwalin"} dwarves2 = {"Gimli", "Balin", "Thorin"} dwarves3 = dwarves1.difference(dwarves2) print(dwarves3) **Output:** {'Dwalin'}

If Statements

	Output: 9000

adding items	device ={
	"brand": "Cisco",
	"model": "9000",
	"RU": 44
	}
	device["software"] = "Cisco IOS XE"
	print(device)
	Output:
	{'brand': 'Cisco', 'model': '9000',
	'RU': 44, 'software': 'Cisco IOS XE'}

changing value	device ={
	"brand": "Cisco",
	"model": "9000",
	"RU": 44
	}
	device["RU"] = 30
	print(device)
	Output:
	{'brand': 'Cisco', 'model': '9000', 'RU': 30}

changing value (with update)	device ={
	"brand": "Cisco",
	"model": "9000",
	"RU": 44
	}
	device.update({"RU": 30})
	print(device)
	Output:
	{'brand': 'Cisco', 'model': '9000', 'RU': 30}

removing an item	device ={
	"vendor": "Cisco",
	"model": "9000",
	"RU": 44
	}
	device.pop("model")
	print(device)
	Output:
	{'vendor': 'Cisco', 'RU': 44}

if statement

```
if (condition):
    body
```

If, else statement

```
if (condition):
    body
else:
    body
```

if, elif,else statement

```
if (condition)
    body
elif (condition):
    body
else:
    body
```

finding even/odd

```
x = 5
if (x % 2 == 0):
    print("x is an even number")
else:
    print("x is an odd number")
```

Output:

```
x is an odd number
```

continue statement

```
numbers=(5,12,25,33,48,50,61)
for x in numbers:
    if x % 5 ==0:
        print("{} divided by five.".format(x))
    else:
        continue
```

Output:

```
5 divided by five.
25 divided by five.
50 divided by five.
```

nested if

```
numbers=(30,44,54,63,78,96)
for x in numbers:
    if x % 3 == 0:
        if x % 9 == 0:
            print ("{} divided 3&9".format(x))
```

Output:

```
54 divided 3&9
63 divided 3&9
```

removing last item

```
device ={
    "vendor": "Nokia",
    "model": "7950 XRS",
    "RU": 39
}
device.popitem()
print(device)
```

Output:

```
{'vendor': 'Nokia', 'model': '7950 XRS'}
```

concatenation

```
print ("Hello" + "Python" + "World")
```

Output:

```
HelloPythonWorld
```

capitalize

```
msg = "welcome to python course!"
x = msg.capitalize()
print (x)
```

Output:

```
Welcome to python course!
```

title

```
msg = "welcome to python course!"
x = msg.title()
print (x)
```

Output:

```
Welcome To Python Course!
```

lower

```
msg = "weLcoMe to PytHon CouRse!"
x = msg.lower()
print (x)
```

Output:

```
welcome to python course!
```

upper

```
msg = "weLcoMe to PytHon CouRse!"
x = msg.upper()
print (x)
```

Output:

if, elif,else	`x = 10`
	`if x > 10:`
	`  print("greater")`
	`elif x < 10:`
	`  print("smaller")`
	`else:`
	`  print("equal")`
	
	Output:
	
	`equal`

if, elif,else with list	`list1=[30, 21, 11, 16, 82, 47, 28, 79]`
	`for x in list1:`
	`  if x < 21:`
	`  print("{} is smaller than 21.".format(x))`
	`  elif x > 21:`
	`  print("{} is greater than 21.".format(x))`
	`  else :`
	`  print("{} is equal to 21.".format(x))`
	
	Output:
	
	`30 is greater than 21.`
	`21 is equal to 21.`
	`11 is smaller than 21.`
	`16 is smaller than 21.`
	`82 is greater than 21.`
	`47 is greater than 21.`
	`28 is greater than 21.`
	`79 is greater than 21.`

Python While Loop

python while loop	`while statement:` `  body`

printing numbers	`x = 0`
	`while x < 5:`
	`  print(x)`
	`  x += 1`
	
	Output:
	
	`0`
	`1`
	`2`
	`3`
	`4`

break statement	`x = 0`

	`WELCOME TO PYTHON COURSE!`

count	`msg = "I like cats. Because cats are cute."`
	`x = msg.count("cats")`
	`print(x)`
	
	Output:
	
	`2`

join	`characters = ["Aragorn", "Arwen", "Legolas"]`
	`print ("-".join(characters))`
	
	Output:
	
	`Aragorn-Arwen-Legolas`

partition	`msg = "welcome to python course"`
	`x = msg.partition("to")`
	`print(x)`
	
	Output:
	
	`('welcome ', 'to', ' python course')`

endswith	`msg = "welcOMe to PytHon CouRse!"`
	`x = msg.endswith("!")`
	
	Output:
	
	`True`

Python For Loops

python for loop	**for** *value* **in** *sequence* *body (any action)*

for loop with lists	`numbers = [1, 2, 3]`
	`for x in numbers:`
	`  print(x*x)`
	
	Output:

```
while x < 10:
  x += 1
  if x == 5:
    break
  print(x)
```

Output:

```
1
2
3
4
```

continue statement

```
x = 0
while x < 10:
  x += 1
  if x == 3:
    continue
  if x == 5:
    continue
  print(x)
```

Output:

```
1
2
4
6
7
8
9
10
```


Python File Operations

open function modes	"**r**" To **read** the file. "**a**" To **append** to the end of the file. "**w**" To **write** to the file. "**x**" To **create** the file.
content of filexyz.txt	Weak people revenge. Strong people forgive. Intelligent people ignore.

file read and print	`x = open("filexyz.txt", "r")`
	`print(x.read())`
	
	Output:
	
	`Weak people revenge.`
	`Strong people forgive.`
	`Intelligent people ignore.`

```
1
4
9
```

for loop with strings

```
for x in "Kate":
   print(x)
```

Output:

```
K
a
t
e
```

for loop with if/else

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8]
even = []
odd = []
for x in numbers:
    if (x % 2 == 0):
        even.append(x)
    else:
        odd.append(x)
print("Even numbers:", even)
print("Odd numbers:", odd)
```

Output:

```
Even numbers: [2, 4, 6, 8]
Odd numbers: [1, 3, 5, 7]
```

break statement

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8]
for x in numbers:
    if (x == 5):
        break
    print(x*x)
```

Output:

```
1
4
9
16
```

continue statement

```
numbers = [1, 4, 5, 6, 7]
for x in numbers:
    if (x == 5):
        continue
    if (x == 7):
        continue
    print(x*x)
```

Output:

```
1
16
```

file write	`x = open("fileabc.txt", "w")`
	`x.write("Good things take time.")`
	`x.close()`
	
	`Good things take time.`

file append	`x = open("fileabc.txt", "a")`
	`x.write("Nice!")`
	`x.close()`
	
	`x = open("fileabc.txt", "r")`
	`print(x.read())`
	
	Output:
	
	`Good things take time.Nice!`

file create	`a = open("Ourfile.txt", "x")`
file remove	`import os` `os.remove("myfile.txt")`

Pyhton Modules

Python modules	https://docs.python.org/3/py-modindex.html
extention	.py
using a module	`import math`
using part of a module	`from math import sqrt` `print(sqrt(25))`

creating a module	`def square(x):`
	`  print(x*x)`
	
	`def hello(x):`
	`  print("Hello ",x)`
	
	record this file as "**ourmodule.py**".

	`36`

nested for loops	`cars= ["Audi", "Aston Martin"]`
	`colors = ["White", "Black"]`
	`for x in cars:`
	`  for y in colors:`
	`  print(x, y)`
	
	Output:
	
	`Audi White`
	`Audi Black`
	`Aston Martin White`
	`Aston Martin Black`

Python Functions

creating function	`def first_function():` `  print("This is Python!")`

calling function	`def first_function():`
	`  print("This is Python Function!")`
	`first_function()`
	
	Output:
	
	`"This is Python Function!`

function with numbers	`def our_function(x,y):`
	`  print(x * y)`
	
	`our_function(3,10)`
	`our_function(5,11)`
	
	Output:
	
	`30`
	`55`

function with strings	`def data(name,surname,no):`
	` print("Name:", name, "Surname:",surname,"No:",`
	
	`data("Gokhan","Kosem",1234)`
	
	Output:

	`import ourmodule`
	`ourmodule.square(5)`
	`ourmodule.hello("Gokhan")`
	
	Output:
	
	`25`
	`Hello Gokhan`

module alias	`import ourmodule as our`

listing module functions	`import math`
	`print(dir(math))`
	
	Output:
	
	`['__doc__', '__loader__', '__name__', ...`

Python PIP Install

checking PIP version	`C:\Users\ipcisco>pip --version`
installing PIP	`C:\Users\ipcisco>pip install xyz`
listing installed packages	`C:\Users\ipcisco>pip list`
unistall a package (xyz)	`C:\Users\ipcisco>pip uninstall xyz`
using a package (xyz)	`import xyz`

Python Math

math operators	**+ (addition)**

	`Name: Gokhan Surname: Kosem No: 1234`

default value	`def newfunc(name = "Legolas"):`
	`  print(name + " is an Elf." )`
	
	`newfunc("Elrond")`
	`newfunc("Galadriel")`
	`newfunc()`
	
	Output:
	
	`Elrond is an Elf.`
	`Galadriel is an Elf.`
	`Legolas is an Elf.`

using asterisk (*)	`def new_func(*elves):`
	`  print("The strongest Elf is " + elves[1])`
	`new_func("Elrond", "Galadriel", "Legolas")`
	
	Output:
	
	`The strongest Elf is Galadriel`

using asterisk (*)	`def letsadd(*args):`
	`  return sum(args)`
	`print(letsadd(15,25,50))`
	
	Output:
	
	`90`

return statement	`def newfunction(a):`
	`  return a * a`
	`print(newfunction(5))`
	
	Output:
	
	`25`

pass statement	`def newfunc():`
	`  pass`
	
	Used to get no error in empty function.

recursive function	`def findfactorial(a):`
	`  if a == 1:`
	`  return 1`
	`  else:`

	– (subtraction) *** (multiplication)** **/ (divide)** **% (difference)** **** (power)**

math operations	`a = 16`
	`b = 8`
	`c = 3`
	`print(a - b + c)`
	Output:
	`11`

finding odd number	`a = [1,2,3,4,5,6]`
	`for x in a:`
	`  if x%2==0:`
	`  print(x)`
	Output:
	`2`
	`4`
	`6`

min, max functions	`a = min(7,15,2)`
	`b = max(8,28,12)`
	`print(a)`
	`print(b)`
	Output:
	`2`
	`28`

pow function	`a = 2`
	`b = 5`
	`print(pow (a,b))`
	Output:
	`32`

sqrt function	`import math`
	`x = math.sqrt(100)`
	`print(x)`
	Output:

	`  return (a * findfactorial(a-1))`
	`print(findfactorial(5))`
	Output:
	`120`

lambda function	`x = lambda y: y * 2`
	`print(x(3))`
	Output:
	`6`

lambda with filter func.	`list1 = [7,12,4,15,3,2,8]`
	`list2 = list(filter(lambda x: (x%2 == 0) , list`
	`print(list2)`
	Output:
	`[12, 4, 2, 8]`

lambda with map func.	`list1 = [7,12,4,15]`
	`list2 = list(map(lambda x: (x > 5) , list1))`
	`print(list2)`
	Output:
	`[True, True, False, True]`

try/except function	`try:`
	`  x=5`
	`  print(x)`
	`except:`
	`  print("An Error!")`
	Output:
	`5`

try/except function	`try:`
	`  print(a)`
	`except:`
	`  print("An Error!")`
	Output:
	`An Error!`

	10.0

factorial function	`import math`
	`x=math.factorial(5)`
	`print(x)`
	
	Output:
	
	`120`

ceil/floor functions	`import math`
	`x = math.ceil(1.7)`
	`print(x)`
	
	Output:
	
	`2`

pi function	`import math`
	`x = math.pi`
	`print(x)`
	
	Output:
	
	`3.141592653589793`

try/except/finally	`try:`
	`  print(a)`
	`except:`
	`  print("An Error!")`
	`finally:`
	`  print("I will close the program.")`
	
	Output:
	
	`An Error!`
	`I will close the program.`

try/except/else	`try:`
	`  a=5`
	`  print(a)`
	`except:`
	`  print("An Error!")`
	`else:`
	`  print("All is Good!")`
	
	Output:
	
	`5`
	`All is Good!`

raise function	`for x in range(10):`
	`  if x > 5:`
	`  raise Exception("Higher 5!")`
	
	Output:
	
	`Traceback (most recent call last):`
	`  File "./prog.py", line 3, in <module>`
	`Exception: Higher 5!`

Escape Characters

\b	**Backspace**
\r	**Carriage Return**
\f	**Form Feed**
\xhh	**Hex value**
\n	**New Line**
\ooo	**Octal value**
\'	**Single Quote**
\t	**Tab**
****	**Backslash**

Comparison Characters

==	Equal
!=	Not Equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

Arithmetic Operators

+	Addition
–	Subtraction
*	Multiplication
/	Division
%	Modulus
**	Exponentiation
//	Floor division

Bitwise Operators

&	AND (Sets each bit to 1 if both bits are 1)
|	OR (Sets each bit to 1 if one of two bits is 1)
^	XOR (Sets each bit to 1 if only one of two bits is 1)
~	NOT (Inverts all the bits)
<<	Zero fill left shift (Shift left by pushing zeros in from the right and let the leftmost bits fall off)
>>	Signed right shift (Shift right by pushing copies of the left most bit in from the left, and let the rightmost bits fall off)

Assignment Operators

=	Asign (x = 1)
+=	Add AND (x += 2) (x=x+2)
-=	Subtract AND (x -= 2) (x=x-2)
*=	Multiply AND (x *= 2) (x=x*2)
/=	Divide AND (x /= 2) (x=x/2)
%=	Modulus AND (x %= 3) (x = x % 3)
//=	Divide Floor AND (x //= 3) (x = x // 3)
**=	Exponent AND (x **= 3) (x = x ** 3)
&=	Bitwise AND (x &= 4) (x = x & 4)
|=	Bitwise OR (x |= 4) (x = x | 4)
^=	Bitwise xOR (x ^= 4) (x = x ^ 4)
>>=	Bitwise Right Shift (x >>= 5) (x = x >> 5)
<<=	Bitwise Left Shift (x <<= 5) (x = x << 5)

Other Operators

and	returns True, if both statements are true.
or	returns True, if one of the statements is true.
not	reverse the result, returns False if the result is true.
is	checks that if both objects are the same object. If yes, it returns True.
is not	checks that if both objects are different objects. If yes, it returns True.
in	returns True, if it finds the value as a member in the sequence.
not in	returns True, if it do not find the value as a members in the sequence.