```
1 print("Hello World")
```

```
Hello World
```

Essential Python 101

To day we are learning Python 101 for beginners.

- variables
- data typyes
- data structures
- function
- control flow
- OOP

```
1 print("I am learing Python 101!")
```

```
I am learing Python 101!
```

```
1 # comment
2 # this is just a note
3 print(1+1)
4 print(2*2)
5 print(5*3)
6
```

```
2
4
15
```

```
1 # casic calculation
2 1 + 1
3 2 * 2
4 5 - 3
5 print(7/2)
6 print(7 // 2)   #fool division
```

```
3.5
3
```

```
1 pow(5, 3)
```

```
125
```

```
1 abs(-666)
```

```
666
```

```
1 # modulo
2 5 % 3
```

```
2
```

```
1 # 5 building blocks
2 # 1. variables
3 # 2. data types
4 # 3. data structures
5 # 4. function
6 # 5. control flow
7 # 6. OOP
```

```
1  # assiign a variable
2  my_name = "Jarb"
3  age = 21
4  gpa = 3.22
5  movie_lover = True # False
```

```
1 # case sensitive
2 print(my_name, age, gpa, movie_lover)
```

```
Jarb 21 3.22 True
```

```
age = 22
print(age)
```

```
22
```

```
# over write value
age = 34
new_age = age - 12
print(age, new_age)
```

```
34 22
```

```
s23_price = 30000
discount = 0.15
new_s23_price = s23_price *(1-discount)

print(new_s23_price)
```

```
25500.0
```

```
# remove variable
del s23_price
```

```
# count variable
age = 21
age += 1
age += 1
age += 1
age -= 2
age *= 2
age /= 2
print(age)
```

```
22.0
```

```
#  data types
# int float ste bool
```

```
age = 21
gpa =  3.22
school = "RMUTT"
movie_lover = True
```

```
# check data types
print(type(age))
print(type(gpa))
print(type(school))
print(type(movie_lover))
```

```
<class 'int'>
<class 'float'>
<class 'str'>
<class 'bool'>
```

```
# convert type
x = 100
x = str(x)

print(x, type(x))

```

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

```
y = False    # T=1, F=0
y = int(y)
print(y, type(y))
```

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

```
z = 1
z = bool(z)
print(z, type(z))
```

```
True <class 'bool'>
```

```
age = 22
print(age + age, age*2, age/2)
```

```
44 44 11.0
```

```
text = "Hello"
text2 =   '   "hahahaha"  '
print(text, text2)
```

```
Hello    "hahahaha"
```

```
# type hint
age: int =  21
my_name: str = "Jarb "
gpa: float = 3.22
seafood: bool = True
```

```
# function
print("hello", "world")
print(pow(5, 2 ), abs(-5))
```

```
hello world
25 5
```

```
# greeting()
def greeting(name) :
    print("Hello! " +  name)
```

```
greeting("Jarb")
```

```
Hello! Jarb
```

```
def add_two_nums(num1, num2):
    print("hello world")
    print("Done !")
    return num1 + num2
```

```
result = add_two_nums(5, 15)
print(result)
```

```
hello world
Done !
20
```

```
def add_two_nums(a: int, b: int)  ->  int :
    return a+b
```

```
add_two_nums(5, 6 )
```

```
11
```

```
 # work with string
 text = "hello world"

 long_text = """
 this is a
 very long text
 this is a new line """

 print(text)
print(long_text)
```

```
hello world

this is a
very long text
this is a new line
```

```
# string template : fstrings
my_name = "Jhon Wick"
location = "London"

text = f"Hi my name is {my_name} and I live in {location}"

print(text)
```

```
Hi my name is Jhon Wick and I live in London
```

```
text = "a duck walks into a bar"
print(text)

```

```
a duck walks into a bar
```

```
# slicing, index starts with 0
print(text[0], text[-1], text[22])
```

```
a r r
```

```
text = "a duck walks into a bar"
```

```
# function vs. method
#  string method
text = text.lower()
```

```
text.replace("duck", "lion")
```

```
'a lion walks into a bar'
```

```
words = text.split()
print(words, type(words))
```

```
['a', 'duck', 'walks', 'into', 'a', 'bar'] <class 'list'>
```

```
" ".join(words)
```

```
'a duck walks into a bar'
```

```
# method = function สร้างขึ้นมาสำหรับ object นั้นๆ
# string method
#string is immutable
```

```
# data structures
# 1. list []
# 2. tupld ()
# 3. dictionary {}
# 4. set {unique}
```

```
# list
shopping_items = ["banana", "egg", "milk"]

shopping_items[0] = "pinepple"

shopping_items[1] = "hm cheese"
print(shopping_items)

```

```
['pinepple', 'hm cheese', 'milk']
```

```
 # list methods
 shopping_items.append("egg")
 print(shopping_items)
```

```
['pinepple', 'hm cheese', 'milk', 'egg', 'egg', 'egg', 'egg']
```

```
# sort items (ascednding order, A-Z)
shopping_items.sort()
print(shopping_items)
```

```
['egg', 'egg', 'egg', 'egg', 'hm cheese', 'milk', 'pinepple']
```

```
scores = [90, 88, 85, 92, 75]
print(len(scores),sum(scores), min(scores), max(scores))
```

```
5 430 75 92
```

```
sum(scores) / len(scores)
```

```
86.0
```

```
def mean(scores):
    return sum(scores) / len(scores)
```

```
scores = [90, 88, 85, 92, 75]
print(len(scores),sum(scores),
      min(scores), max(scores), mean(scores))
```

```
5 430 75 92 86.0
```

```
# remove last item
shopping_items.pop()
shopping_items
```

```
['egg', 'egg', 'egg', 'egg']
```

```
shopping_items.append("milk")
print(shopping_items)
```

```
['Pig', 'egg', 'egg', 'egg', 'egg', 'milk', 'milk', 'milk']
```

```
# insert ()
shopping_items.insert(0, "Pig")
```

```
# list + list
items1 = ['egg', 'milk']
items2 = ['banana', 'bread']

print(items1 + items2)
```

```
['egg', 'milk', 'banana', 'bread']
```

```
 # tuple () is immutable
tup_items = ('egg', 'bread', 'pepsi')
tup_items
```

```
('egg', 'bread', 'pepsi')
```

```
tup_items.count('egg')
```

```
1
```

```
# username passwoed
# student1, student2
s1 = ("id001", "123456")
s2 = ("id002", "654321")
user_pw = (s1,s2)

print(user_pw)
```

```
(('id001', '123456'), ('id002', '654321'))
```

```
# tuple unpacking
username, password =s1

print(username, password)

```

```
id001 123456
```

```
# tuple unpacking 3 values
name, age, gpa = ("Jarb",  21,  3.22)
print( name, age)
```

```
Jarb 21
```

```
# set {unique}
coures = ["Python", "Python", "R", "SQL", "SQL", "sql"]
```

```
set(coures)
```

```
{'Python', 'R', 'SQL', 'sql'}
```

```
#  dictionart key: value pairs
course = {
```

```
    "name": "Data Science Bootcamp",
    "duration": "4 months",
    "students": 200,
    "replay": True,
    "skills":  ["google sheets", "SQL"]
}
```

```
course["strat_time"]  = "9am"

course["language"] = "Thai"
del course["language"]
del course["strat_time"]

course["replay"] = False

course
```

```
{'name': 'Data Science Bootcamp',
 'duration': '4 months',
 'students': 200,
 'replay': False,
 'skills': ['google sheets', 'SQL']}
```

```
course["skills"][1:]
```

```
['SQL']
```

```
# Recap
# list, ddictionary = mutable
# tuple, string = immutable
```

```
# contorl flow
# if for while
```

```
 # final exam 150 questions, pass >= 120
def grade(score):
    if score >= 120:
        return "Excellent"
    elif score >= 100:
        return "good"
    elif score >= 80:
        return "Okay"
    else:
        return "Need to read more!"
```

```
result = grade(95)
print(result)
```

```
Okay
```

```
 # use and, or  in condition
 # ccourse == data science, score >= 80 passed
 # course == english, score >= 70 passd
def grade(course, score):
    if course == "english" and score >= 70 :
        return "passed"
    elif course ==  "data science" and score >= 80 :
        return "passed"
    else :
        return " failed "
```

```
grade("data science", 81)
```

```
'passed'
```

```
not True; not False
```

```
True
```

```
# for loop
# if score >= 80, passed
def grading_all(scores):
    new_score = []
    for score in scores :
        new_score.append(score  + 2)
    return(new_score)
```

```
8
9
```

```
grading_all([75, 88, 90, 95, 52])
```

```
[77, 90, 92, 97, 54]
```

```
# list comprehension
scores = [75, 88, 90, 95, 52]
```

```
new_scores = [s*2 for s in scores]

new_scores
```

```
[150, 176, 180, 190, 104]
```

```
# list comprehension
friends = ["toy", "ink", "bee", "zue", "yos"]
[f.upper() for f in friends]

```

```
['TOY', 'INK', 'BEE', 'ZUE', 'YOS']
```

```
# while loop
count = 0

while count <5:
    print("hello")
    count +=1
```

```
hello
hello
hello
hello
hello
```

```
# chatbot for fruit order
username = input("what is your name?")
```

```
what is your name?jarb
```

```
username
```

```
'jarb'
```

```
def chatbot() :
    fruits = []
    while True :
        fruit = input("What fruit do you want to order?")
        if fruit == "exit":
            return fruits
        fruits.append(fruit)
```

```
chatbot()
```

```
What fruit do you want to order?banana
What fruit do you want to order?stacke
What fruit do you want to order?exit
['banana', 'stacke']
```

```
# HW 01 - chatbot to order pizza
# HW 02 - pao ying chup
```

```
age = int( input("how old are you"))
```

```
how old are you21
```

```
type (age)
```

```
int
```

```
# OOP - object Orented Programming

```

```
class Dog :
    def __init__(self, name, age, breed) :
        self.name = name
        self.age =age
        self.breed = breed
```

```
dog1 = Dog("kim", 2 , "chihuahua")
dog2 = Dog("pem", 3, "bulldog")
dog3 = Dog("tim", 3.5, "german shepherd")
```

```
print(dog1.name, dog1.age, dog1.breed)
```

```
kim 2 chihuahua
```

```
class Employee:
    def __intit__(self, id, name, dept, pos) :
        self.id = id
        self.name = name
        self.dept = dept
        self.pos = pos     # position
    def hello(self) :
         print("Hello!")
```

```
emp1 = Employee(1, "Jhon", "Finance", "Financial Analyst")
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-298-cf6dc4680c98> in <cell line: 1>()
----> 1 emp1 = Employee(1, "Jhon", "Finance", "Financial Analyst")

TypeError: Employee() takes no arguments

SEARCH STACK OVERFLOW
```

```
print(emp1.name, emp1.pos)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-289-135fcef07a8f> in <cell line: 1>()
----> 1 print(emp1.name, emp1.pos)

NameError: name 'emp1' is not defined

SEARCH STACK OVERFLOW
```

```
1
```

```
1
```

```
1
```

```
1
```

ดับเบิลคลิก (หรือกด Enter) เพื่อแก้ไข