```
1 print("hello world")
```

```
hello world
```

Essential Python 101

```
1 print("I am learning Python 101")
```

```
I am learning Python 101
```

```
1 # comment note
2 1+1
3 print(2*2)
```

```
4
```

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

```
3.5
3
```

```
1 pow(5, 2)
```

```
25
```

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

```
666
```

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

```
1
```

```
1 # assign a variable
2 my_name = "pp"
3 age = 22 # integer
4 gpa = 2.58 # float/ real number
5 movie_lover = True
```

```
1 print(age, gpa, movie_lover, my_name)
```

```
22 2.58 True pp
```

```
# over write a value
age = 23
print(age)
```

```
23
```

```
s23_price = 30000
discount = 0.15 # 15%/ 15/100
new_s23_price = s23_price *(1-discount)
print(new_s23_price)
```

```
25500.0
```

```
# remove variables
del new_s23_price
```

```
# count variables
age = 22
age += 1
age -= 1
age *= 2
age /= 2
print(age)
```

```
22.0
```

```
# data types
# int float str bool
```

```
age = 22
gpa = 2.58
school = "swu"
movie_lover = True
```

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

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

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

```
x = str(x)
print(x, type(x))
```

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

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

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

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

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

```
text = "hello"
text2 = ' "hahaha" '
print(text, text2)

```

```
hello  "hahaha"
```

```
text + text
```

```
'hellohello'
```

```
5+5
```

```
10
```

```
# type hint
age: int = 22
print(age, type(age))
```

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

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

```
hello world
25 5
```

```
# greeting()
def greeting(name="pp", location="cnx"):
    print("hello! " + name)
    print("they are living in " + location)
```

ดับเบิลคลิก (หรือกด Enter) เพื่อแก้ไข

```
greeting(location="japan", name="tanjiro")
```

```
hello! tanjiro
they are living in japan
```

```
def add_two_nums(x, y):
    return x + y
    print
```

```
result = add_two_nums(2, 3)
print(result)
```

```
5
```

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

```
add_two_nums(1,1)
```

```
2
```

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

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

print(text)
print(long_text)
```

```
hello world
 this is
a very long
this is a new line
```

```
# string template : fstrings
my_name = "pp"
location = "cnx"

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

print(text)
```

```
Hi! my name is pp and I live in cnx
```

```
1 text =  "a cat walks into a bar"
2 print(text)
```

```
a cat walks into a bar
```

```
1 # slicing, index starts with 0
2 text[-1]
```

```
'r'
```

```
1 len(text)
```

```
22
```

```
1 text[2:6]
```

```
'cat '
```

```
1 # string is immuntable
2 name = "Python" # -> cython
3 name = "C" + name[1: ]
4 print(name)
```

```
Cython
```

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

+ โค้ด + ข้อความ

```
1 len(text)
```

```
23
```

```
1 # function vs. method
2 # strimg methods
3 text = text.upper()
4 print(text)
```

```
A DUCK WALKS INTO A BAR
```

```
1 text = text.lower()
2 text
```

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

```
1 text.replace("duck", "cat")
2
```

```
'a cat 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'
```

```
# data structure
# 1. list is mutable
shopping_items = ["egg", "milk", "bacon"]
print(shopping_items)
print(shopping_items[0])
print(shopping_items[1: ])
print(len(shopping_items))
```

```
['egg', 'milk', 'bacon']
egg
['milk', 'bacon']
3
```

```
shopping_items[0] = "carrot"
print(shopping_items)
```

```
['carrot', 'milk', 'bacon']
```

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

```
['carrot', 'milk', 'bacon', 'egg']
```

```
# sort items (ascending oder, A-Z)
shopping_items.sort(reverse=True) #descending order
print(shopping_items)
```

```
['milk', 'egg', 'bacon']
```

```
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 in list
shopping_items.pop()
```

```
'bacon'
```

```
shopping_items
```

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

```
shopping_items.append("egg")
```

```
shopping_items.remove("milk")
```

```
# .insert()
shopping_items.insert(1, "milk")
```

```
shopping_items
```

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

```
# list + list
items1 = ['egg', 'milk']
items2 = ['bannana', 'carrot']
print(items1  + items2)
```

```
['egg', 'milk', 'bannana', 'carrot']
```

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

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

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

```
3
```

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

```
6
7 print(user_pw)
```

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

```
1 # tuple unpacking
2 username, password = s1
3
4 print(username, password)
```

```
id001 123456
```

```
1 # tuple unpacking 3 values
2 name, age, _= ("John", 42, 3.78)
3 print(name, age)
```

```
John 42
```

```
1 # set {unique}
2 courses = ["Python", "Python", "R", "SQL"]
```

```
1 set(courses)
```

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

```
1  # dictionary key: value pairs
2  course = {
3      "name": "Data Science Bootcamp",
4      "duration": "4 months",
5      "students": 200,
6      "replay": True,
7      "skills": ["Google Sheets", "SQL", "R", "Python", "Stat",
8                 "ML", "Dashboard", "Data Transformation"]
9
10 }
```

```
1 course["start_time"] = "9am"
2 course["languague"] = "Thai"
3 course
```

```
{'name': 'Data Science Bootcamp',
 'duration': '4 months',
 'students': 200,
 'replay': True,
 'skills': ['Google Sheets',
  'SQL',
  'R',
  'Python',
  'Stat',
  'ML',
  'Dashboard',
  'Data Transformation'],
```

```
 'start_time': '9am',
 'languague': 'Thai'}
```

```
1
```

```
1 # delete
2 del course["languague"]
3 course
```

```
{'name': 'Data Science Bootcamp',
 'duration': '4 months',
 'students': 200,
 'replay': True,
 'skills': ['Google Sheets',
  'SQL',
  'R',
  'Python',
  'Stat',
  'ML',
  'Dashboard',
  'Data Transformation'],
 'start_time': '9am'}
```

```
1 course["replay"] = False
```

```
1 course
```

```
{'name': 'Data Science Bootcamp',
 'duration': '4 months',
 'students': 200,
 'replay': False,
 'skills': ['Google Sheets',
  'SQL',
  'R',
  'Python',
  'Stat',
  'ML',
  'Dashboard',
  'Data Transformation'],
 'start_time': '9am'}
```

```
1 course["skills"][-3: ]
```

```
['ML', 'Dashboard', 'Data Transformation']
```

```
1 list( course.keys())
```

```
['name', 'duration', 'students', 'replay', 'skills', 'start_time']
```

```
1 list( course.values())
```

```
['Data Science Bootcamp',
 '4 months',
```

```
 200,
 False,
 ['Google Sheets',
  'SQL',
  'R',
  'Python',
  'Stat',
  'ML',
  'Dashboard',
  'Data Transformation'],
 '9am']
```

```
1 list(course.items())
```

```
[('name', 'Data Science Bootcamp'),
 ('duration', '4 months'),
 ('students', 200),
 ('replay', False),
 ('skills',
  ['Google Sheets',
   'SQL',
   'R',
   'Python',
   'Stat',
   'ML',
   'Dashboard',
   'Data Transformation']),
 ('start_time', '9am')]
```

```
1 course.get("replay")
```

```
False
```

```
1 # control flow
2 # if for while
```

```
1 # final exam 150 question, pass >= 120
2 score = 125
3 if score >= 120:
4     print("passed")
5 else:
6         print("failed")
7
```

```
passed
```

```
1 def grade(score):
2     if score >= 120:
3      return "Eecellent"
4     elif score >= 100:
5         return "Good"
6     elif score >= 80:
7         return "Okay"
```

```
    else:
      return "Need to read more!"
```

```
result = grade(90)
print(result)
```

```
Okay
```

```
# use and , or in condition
# course == data science, score >= 80 passed
# course == english, score >= 70 passed
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", 85)
```

```
'passed'
```

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

```
grading_all([70, 54, 68, 35])
```

```
[72, 56, 70, 37]
```

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

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

```

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

```
friends = ["pin", "plaii", "eye", "save", "dame"]
[f.upper() for f in friends]
```

```
['PIN', 'PLAII', 'EYE', 'SAVE', 'DAME']
```

```
# while loop
count = 0
```

```
count = 0

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

```
hello
hello
hello
hello
hello
```

```
# chatbot for fruit order
user_name = input("What is yout name?")
```

```
What is yout name?pp
```

```
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? all berry
What fruit do you want to order? lemon
What fruit do you want to order? exit
['banana', 'all berry', 'lemon']
```

```
chatbot("banana")
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-89-d1a3166898fd> in <cell line: 1>()
----> 1 chatbot("banana")

TypeError: chatbot() takes 0 positional arguments but 1 was given

 SEARCH STACK OVERFLOW
```

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

```
how old are u 23
```

```
# OOP - Object oriented programing
# Dog class
class Dog:
    def __init__(self, name, age, breed): # dunner - double under score
```

```
        self.name = name
        self.age = age
        self.breed = breed

dog1 = Dog("milo", 2, "chihuahua")
dog2 = Dog("bacon", 1 ,"shiba")
dog3 = Dog("mootod", 1.2, "golden revertrive")

```

```
print(dog1.name, dog1.breed, dog1.age,
      dog2.name, dog2.breed, dog2.age,
      dog3.name, dog3.breed, dog3.age)
```

```
milo chihuahua 2 bacon shiba 1 mootod golden revertrive 1.2
```

```
class Employee:
    def __init__(self, id, name, dept, pos):
        self.id = id
        self.name = name
        self.dept = dept
        self.pos = pos # position
    def hello(self):
        print(f"Hello! my name is {self.name}")
    def work_hours(self, hours):
        print(f"{self.name} works for {hours} hours.")
    def change_dept(self, new_dept):
        self.dept = new_dept
        print(f"{self.name} is now in {self.dept}.")

```

```
emp1 = Employee(1, "Pin", "Data", "Data analyst")
print(emp1.id, emp1.name, emp1.pos)
```

```
1 Pin Data analyst
```

```
emp1.hello()
```

```
Hello! my name is Pin
```

```
emp1.work_hours(8)
```

```
Pin works for 8 hours.
```

```
emp1.change_dept("Data science")
```

```
Pin is now in Data science.
```

```
emp1.dept
```

```
'Data science'
```

```
class ATM:
    def __init__(self, name, bank, balance):
        self.name = name
        self.bank = bank
        self.balance = balance
    def deposit(self, amt):
        self.balance += amt
```

```
scb = ATM("pp", "scb", 500)
print(scb.balance)
```

```
500
```

```
scb.deposit(100)
print(scb.balance)
```

```
600
```

```
1
```

✓ 0 วินาที เสร็จสมบูรณ์เมื่อ 18:18