```
i = 1
while i <= 5 :
    print ("*" , ends = " ")
    i = i+1
```

خروجی → * * * * *

⇐ دریافت سه متغیر از یک سطر (نوشتن)

```
x, y, z = 2, "reza", 76,6
```

⇐ نوشتن سه متغیر در لیست

```
x, y, z = ["a", "b", "c"]
```

↓ مثال

```
x1, x2, x3, x4 = input().split(" ")
print (x1)
```

خروجی که شماره ها را تحلیل داده ایم → : 556, 1 34 45

خروجی آخر → 556

⇐ استفاده از for

```
a = " welcome"
for x in a :
    print (x)
```

خروجی →

```
w
e
l
c
o
m
e
```

```
for x in range(1, 6):
    print (x)
```

خروجی →

```
0
1
2
3
4
5
```

```
my_list = ["a", 2, 3,14 , ["a", "b"]]
print (my_list [4][0])
```

خروجی → a

⇐ جدا کردن کلمات از هم

```
a = "a b c d"
print ( a.split(" "))
```

خروجی → ['a' , 'b', 'c', 'd']

⇐ نوشتن ماتریس

```
i = 5
while i <= 5 :
    j = 1
    while j <= 3:
        print ( i , j )
        j = j+1
    i = i +1
```

⇐ استفاده از append

```
names = ["reza", "ali", "parisa"]
names. append ("parsa")
print(names)
```

خروجی →

['reza' , 'ali' , 'parisa' , 'parsa']

تابع .split(): فقط برای str ها استفاده می‌کنیم. مثال ⇐

```
n = input()
n = n.split()
```

⇐ اگر بخواهیم به عنوان عدد استفاده کنیم →

x_1 = int(n[0]) , x_2 = int(n[1])

جایگذاری چیزی یا چند چیز در لیست ⇒

```
names = ["reza", "ali", "parisa"]
names[0] = "ahmad"
print(names)
```

خروجی ['ahmad', 'ali', 'parisa']

نوشتن کپی لیست ⇒

```
names = ["reza", "ali", "parisa"]
names_copy = names.copy()
```

حال اگر تغییری در names ایجاد کنیم names_copy به همان شکل قبلی خودش باقی میماند.

حذف اولین عضو لیست ⇒

```
names = ["reza", "ali", "parisa"]
names.clear()
print(names)
```

خروجی []

شمارش چیزی در لیست ⇒

```
names = ["reza", "ali", "parisa"]
print(names.count("reza"))
```

خروجی 1

ok اگر شمارش چیزی که در لیست نباشد بخواهیم بدهد صفر را نشان میدهد.

نوشتن لیست در لیست ⇒

```
name1 = ["reza", "ali"]
name2 = ["parisa", "parsa"]
name1.append(name2)
print(name1)
```

خروجی ['reza', 'ali', ['parisa', 'parsa']]

لیست با هم نوشته شوند →

```
print(name1 + name2)
```

خروجی ["reza", "ali", "parisa", "parsa"]

پیدا کردن بزرگترین عدد وارد شده ⇒

```
number = int(input())
max_num = -1
if number < 0 :
    number = number * -1
while number != 0 :
    value = number % 10
    if value >= max :
        max = value
    number = number // 10
print(max_number)
```

index گیر دادن ⇒

```
name1 = ["reza", "ali", "parisa"]
print(name1.index("reza"))
```

خروجی ⇒ 0 (صفر)

index گیر دادن ⇒

```
name1 = ["reza", "ali", "parisa", "ali"]
print(name1.index("ali"))
```

خروجی ⇒ 1

چند تا صدگان، دهگان، یکان ⇒

```
number = int(input())
Sadgan = number // 100
dahgan = (number % 100) // 10
```

پرینت از ایندکس تا ایندکس مخصوص ⇒ a = [1, 2, 3, 4, 5, 6, 7, 8]

```
print(a[2:5])
```

* پرینت لیست از ایندکس دو تا ایندکس پنجی در ایندکس پنجم خودش پرینت نمیشود.

پرینت ایندکس ها ⇒

```
a = [1, 2, 3, 4, 5, 6, 7, 8]
print(a[2:])
```

* پرینت لیست از ایندکس سه تا آخر لیست.

اضافه کردن ایندکس در جای مشخص ⇒

```
student_grades = [18, 20, 12, 14]
student_grades.insert(3, 17)
print(student_grades)
```

(3: ایندکس مورد نظر ، 17: چیزی که میخواهیم اضافه کنیم)

خروجی ⇒ [18, 20, 12, 17, 14]

کد جلب ⇒

```
list: [18, 20, 12, 14]
for i in range(len(list)):
    print(list[i], end=' ')
```

خروجی ⇒ 18, 20, 12, 14.

با استفاده از Remove می‌توانیم با دادن عنصری که می‌خواهیم حذف شود، عنصر را از لیست حذف کنیم. با استفاده از Del می‌توانیم با ایندکس، هر عنصری را حذف کنیم.

```
student_grades = [18, 20, 12, 14]
student_grades.remove(20)
print(student_grades)
```

خروجی → [18, 12, 14]

```
student_grades = [18, 20, 12, 14]
del student_grades[1]
print(student_grades)
```

خروجی → [18, 22, 14]

اضافه کردن در لیست ⇐

```
country_capitals = {}
country_capitals['Iran'] = "Tehran"
print(country_capitals)
```

خروجی → {"Iran" : "Tehran"}

شماره ایندکس ها ⇐

```
 -6    -5    -4    -3    -2    -1
["a", "b", "c", "D", "E", "F"]
  0     1     2     3     4     5
```

برعکس کردن ⇐

```
my_list = [5, 6, 1, 2, 3, 9, 81]
print(my_list[5:2:-1]
```

خروجی → [9, 3, 2]

یعنی از ایندکس پنجم تا قبل از دوم را به صورت برعکس می‌کنیم و منفی اینجا نقش برعکس کردن را دارد.

دیکشنری ⇐

```
country_capitals = {"U.S": "D.C", "Italy": "Rome"}
print(country_capitals["Italy"])
```

خروجی → Rome

برای دیکشنری از Del ⇐ country_capitals {"Iran": "Tehran"}

```
del country_capital["Iran"]
print(country_capital)
```

خروجی → {}

از لیست‌ها داریم ⇐

```
country_capitals = {"U.S": "D.C", "Italy": "Rome", "England": "London"}
key_list = country_capitals.key()
print(key_list)
```

خروجی → ["U.S", "Italy", "England"]

```
value_list = country_capitals.values()
print(value_list)
```

خروجی → ["D.C", "Rome", "London"]

سوال ⇒

1 : a=1

2 : -a=2

3 : -a2=3

4 : 2a=4

جواب ⇒ line 4 throws a syntax error, because variables cannot start with a number.

Slice ⇒

The complete syntax of slicing a list is ⇒ [start:end:step]. So, [::2] means get everything from start to end at a step of two.

Python comprasion operators ⇒

Equal => ==

Not Equal => !=

Greater than => >

Less than => <

Addition ⇒ +

Substraction => -

Division => /

modulus => % ⇒ باقی‌مانده

floordivision => //

Shortcut ⇒

[:3] is just a shortcut [0:3]. Both would do the same job, so you can work from index 0 (the first index) to the third one.

---

slice رشته ⇒

```
letters = ["a","b","c","d","e","f","g","h","i"]
Print(letters[::2])
```

خروجی ⇒ ['a','c','e','g','i']

سوال: create a script that generates and prints a list of numbers from 1 to 20?

جواب:

```
my_range = range(1,21)
Print(list(my_range))
```

اگر بخواهیم اعداد اعشاری رو با رقم x ام اعشار بزنیم اینطوری عمل می کنیم ⇐

```python
n= round(float(input("")), 2)
```

(round: تا رقم گفته شده گرد کردن — float: اعداد اعشاری — 2: تا دو رقم اعشار)

ساختار function ⇒

```python
def Salambye(name):
        Print ("Salam", name)
        Print ( "bye bye", name)
Salambye (" Jadi")
```

خروجی →

```
Salam Jadi
bye bye Jadi
```

```python
def hoghoogh ( hour , Per_hour):
        Total_daramad = hour * Per_hour
        return Total_daramad
Print ( ~~Tot~~ hoghoogh (8,2))
```

خروجی → 16

```python
def hoghoogh( hour, Per_hour):
        if hour >8:
                return " error!"
        else...
```

درست کردن board :

```python
def creat_board (size):
        board = [["" for _ in range (size)] for _
                in range(size)]
        return board
        Print(creat_board (5))
```

خروجی → [['', '', '', '', ''], ...] (۵ بار دیگر هست)

درست کردن board :

```python
def Show_board(board):
        for i in range (len(board)):
                for j in range(len(board)):
                        Print(board[i][j] ,end="|")
```

اگر کاربر z و یا y را وارد نکرد دو عدد دیگر را به عنوان پیش فرض برای آن ها در نظر گرفته شود :

```python
def x ( f, y=1 , z=2):
        pass
```

تابع بازگشتی فاکتوریل :

```python
def fac(n):
        if n== 1:
                return 1
        return n* fac(n-1)
Print ()
```

تابع بازگشتی توان :

```python
def power(x,n):
    if n==1:
        return x
    return x * power(x,n-1)
```

کد عضو :

```python
def check(password):
    if len(password)<3:
        return False
    return True.
```

```python
passwords =["123456","123456789"]
def is-valid-password(password):
    if password in passwords:
        return False
    return True.
```

Print(f" reza is {20}") خروجی Reza is 20

* f string is a way to format strings in python.

کد حساب :

```python
for id ba, grade in grades.items():   # str
    temp =f" {id} : {grade}"
    f.write(temp)
```

خروجی →
13 : 18
14 : 16
⋮
→ طبق داده هایی که به آن داده ایم

تابع بازگشتی فیبوناچی :

```python
def fibo(n):
    if n==1:
        return 1
    elif n==2:
        return 1
    return fibo(n-1) + fibo(n-2)
```

(برابر است)

باز کردن یک txt file :

```python
f=open ("file.txt", "w")
s= "ali \n reza"
f.write(s)
```

* اگر بعداز ali (\n) بنویسیم به خط بعدی می رود. (slash n)

By defult the read() method returns the whole text but you can also specify how many ~~charecters~~ characters you want to return:

```python
f=open("demofile.txt" ,"r")
print(f.read(5))
```

By looping through the lines of the file, you can read the whole file, line by line:

```python
f= open("demo file", "r")
for x in f:
    print(x)
```

خواندن txt :

```
f = open("file.txt", "r")
print(f.read())
```

تعیین کردن type :

```
f = open("file.txt", "r")
x = (f.read())
print(type(x))
```

خروجی → str.

```
f = open("file.txt", "r")
x = (f.readline())
print(type(x))
```

خروجی → list.

استفاده strip :

```
f = open("file.txt", "r")
x = (f.readline())
for element in x :
    element = element.strip()
    print(element)
```

```
f = open("file.txt", "r")
x = (f.readline())
for element in x :
    element = element.strip()
    id, grade = element.split(" : ")
    print(f"id is {id} ---grade is {grade}")
```

خروجی →

```
id is 12 --- grade is 19
id is 13 --- grade is 18
⋮
```

در فایل ها "a" به معنی "append" است و "w" به معنای "write" که هرچه داخل فایل گذاشته‌ایم را حذف می‌کند و چیزی که ما می‌خواهیم را جایگزین آن می‌کند.

To create a new file:

"x" → create → will create a file, returns an error if the file exist.

"a" → append → will create a file if the specified file does not exist.

"w" → write → will create a file if the specified file does not exist.

To delete a file, you must import the OS module, and run its os.remove() functio

مثال →

```
import os
os.remove("demofile.txt")
```