آموزش زبان برنامه نویسی پایتون
بخش اول

مرجع اصلی

http://w3schools.com

محمدجواد فدائی‌اسلام

۳۰ شهریور ۱۳۹۹

باسمه تعالی

فهرست مطالب

۱- مقدمه

۲- متغیر

۳- عملگرها در پایتون

۴- آرایه‌ها (مجموعه‌ها) در پایتون

۵- توضیحات

۶- دستور ات شرطی

۷- دستور while

۸- دستور for

۹- تمرین

۱- مقدمه

پایتون چیست؟

پایتون یک زبان برنامه نویسی محبوب است. این زبان توسط آقای گیدو ون روسوم[۱]ایجاد شد و در سال ۱۹۹۱ منتشر شد.

چرا پایتون؟

- پایتون بر روی سیستم‌های مختلف کامپیوتری (ویندوز، مَک، لینوکس، رزپری پای و ...) اجرا می‌شود.
- ساختار[۲]زبانی ساده‌ای مشابه با زبان انگلیسی دارد.
- ساختاری دارد که به برنامه‌نویسان اجازه می‌دهد برنامه خود را با تعداد خط کمتری نسبت به سایر زبان‌های برنامه‌نویسی بنویسند.
- زبان مفسری است.
- می‌توان به صورت رویه‌ای[۳] شی‌گرا یا تابعی[۴]در آن به برنامه نویسی پرداخت.

مقایسه ساختار نحوی پایتون با دیگر زبان‌های برنامه‌نویسی

- پایتون برای خوانابودن طراحی شده است. مشابه زبان انگلیسی است که متاثر از ریاضیات باشد.
- در زبان C هر دستور برنامه نویسی با سمی‌کلن پایان می‌یابد. اما در پایتون دستور بعدی در خط جدید نوشته می‌شود.
- در زبان برنامه نویسی C از {} برای معرفی بلوک‌های برنامه استفاده می‌شود. اما در پایتون، تورفتگی[۵]نشان‌دهنده بلوک‌های برنامه است.

نصب پایتون

در اغلب سیستم‌عامل‌های امروزی پایتون نصب است. برای پی‌بردن به این موضوع در خط فرمان[۶] ویندوز دستور زیر را وارد نمایید.

```
C:\Users\Your Name>python –version
```

در صورت نصب نبودن، نرم‌افزار را از آدرس زیر دانلود و نصب نمایید. در این متن ویرایش ۳ پایتون آموزش داده شده است.

https://www.python.org

اجرای اولین برنامه از خط فرمان

---

۱ - Guido van Rossum
۲ - syntax
۳ - procedural
٤ - functional
٥ - indentation
٦ - command line

برای آزمایش سریع یک دستور می توان در خط فرمان، دستور python را اجرا نمود.

```
D:\>python

Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:43:08) [MSC v.1926 32 bit (Intel)] on win32

Type "help", "copyright", "credits" or "license" for more information.

>>>
```

با نمایان شدن <<< شما می‌توانید دستورات پایتون را اجرا نمایید. به طور مثال دستور زیر را اجرا نمایید.

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

پس از اتمام اجرای دستورات پایتون با دستور exit() به خط فرمان برای اجرای دستورات سیستم عامل برگردید.

اجرای دستورات پایتون نوشته شده در یک فایل

برنامه‌های پایتون را می‌توان در هر ویرایشگر متنی نوشت و با پسوند py ذخیره نمود. سپس در خط فرمان آن را اجرا نمود (توجه: مسیر فایل باید برای سیستم شناخته شده باشد). به طور مثال خط زیر را در یک ویرایشگر بنویسید و نام فایل را test1 قرار دهید.

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

این فایل را در خط فرمان به صورت زیر اجرا نمایید.

```
C:\Users\Your Name>python test1.py
```

در بخش اول (این نوشته) ما از محیط توسعه یکپارچه پایتون برای نوشتن و اجرای برنامه‌ها استفاده خواهیم کرد.

۲- متغیر

در پایتون دستوری برای اعلان[۷]متغیر وجود ندارد. متغیر در زمان تخصیص مقدار ایجاد می‌شود:

```
x = 5
y = "John"
```

نوع متغیر با تخصیص جدید به صورت خودکار می‌تواند تغییر کند. در دستورات زیر ابتدا نوع متغیر x عدد صحیح است که در انتساب بعدی به رشته تغییر می کند.

```
x = 4
x = "Sally"
print(x)
```

۷ - declaration

رشته‌ها با نقل قول یگانه یا دوگانه[۸]تعریف می‌شوند و این دو باهم تفاوتی ندارد.

```
x = "John"
x = 'John'
```

نام متغیرها می‌تواند ترکیبی از عدد، حرف و خط زیر (_) باشد. پایتون به حروف کوچک و بزرگ حساس است و نام متغیر با عدد نباید شروع شود.

می‌توان با دستور زیر چند مقدار را به چند متغیر نسبت داد.

```
x, y, z = "Orange", "Banana", "Cherry"
```

و یا یک مقدار را به چند متغیر نسبت داد.

```
x = y = z = "Orange"
```

تابع print برای نمایش مقدار متغیر است. با استفاده از + می توان دو متغیر را باهم جمع نمود.

```
x = "Python is "
y = "awesome"
z =  x + y
print(z)
```

```
x = 5
y = 10
print(x + y)
```

دستور زیر غلط است.

```
x = 5
y = "John"
print(x + y)
```

انواع داده

به طور پیش‌فرض پایتون از انواع داده زیر پشتیبانی می‌کند:

Text Type	str
Numeric Types	int, float, complex
Sequence Types	list, tuple, range
Mapping Type	dict
Set Types	set, frozenset

۸ - single or double quotes

Boolean Type	bool
Binary Types	bytes, bytearray, memoryview

برای اطلاع از نوع داده، از تابع type می‌توان به صورت زیر استفاده نمود:

```
x = 5
print(type(x))
```

در زمان تخصیص مقدار به متغیر نوع آن تعیین می‌شود. اما از توابع سازنده‌ای[۹] که هم‌نام با نام تابع هستند می‌توان برای تعریف نوع خاصی از داده استفاده کرد. مثال:

```
x = float(20)
```

در دستور بالا float نام تابع سازنده‌ای است که باعث می‌شود نوع متغیر x عدد اعشاری شود. اگر از این تابع سازنده استفاده نشود نوع متغیر x، عدد صحیح می‌شود.

عدد در پایتون

به صورت پیش‌فرض سه نوع عدد داریم:

- عدد صحیح

```
x = 1
y = 35656222554887711
z = -3255522
```

محدودیتی در تعداد ارقام وجود ندارد.

- عدد اعشاری

```
x = 1.10
y = 1.0
z = -35.59
```

محدودیتی در تعداد ارقام وجود ندارد. گاهی برای نمایش عدد اعشاری از e استفاده می‌شود که نشان‌دهنده توان ده است.

```
x = 35e3
y = 12E4
z = -87.7e100
```

$y = 12E4 = 12 \times 10^4$

۹ - constructor function

- عدد مختلط

قسمت اعشاری با استفاده از حرف j نمایش داده می‌شود.

```
x = 3+5j
y = 5j
z = -5j
```

با توابع int()، float() و complex() می‌توان نوع متغیر را عوض نمود. عدد صحیح و اعشاری به هم و به مختلط تبدیل می‌شود. اما عدد مختلط به نوع دیگری تبدیل نمی‌شود.

تغییر نوع داده

با توابع int()، float() و str() می‌توان نوع آرگومان را عوض نمود. به مثال توجه شود:

```
x = int(1)   # x will be 1
y = int(2.8) # y will be 2
z = int("3") # z will be 3
```

```
x = float(1)     # x will be 1.0
y = float(2.8)   # y will be 2.8
z = float("3")   # z will be 3.0
w = float("4.2") # w will be 4.2
```

```
x = str("s1") # x will be 's1'
y = str(2)    # y will be '2'
z = str(3.0)  # z will be '3.0'
```

متغیر رشته‌ای

همان‌طور که گفته شد، رشته‌ها با نقل قول یگانه یا دوگانه تعریف می‌شوند و این دو باهم تفاوتی ندارد. می‌توان با یک نقل قول سه‌تایی یک متغیررشته‌ای چندخطی ایجاد نمود.

```
a = """Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed do eiusmod tempor incididunt
ut labore et dolore magna aliqua."""
print(a)
```

```
a = '''Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed do eiusmod tempor incididunt
ut labore et dolore magna aliqua.'''
print(a)
```

مانند دیگر زبان‌های برنامه نویسی متداول، متغیر رشته‌ای در پایتون همان ارایه‌ای از بایت‌ها است که کاراکترهایی با کدگذاری یونیکد در آن قرار دارند. در پایتون نوع داده کاراکتر وجود ندارد و کاراکتر همان آرایه یک‌عنصره است. برای دسترسی به عناصر آرایه می‌توان از [] استفاده کرد.

```
a = "Hello, World!"
print(a[1])
```

کد بالا عنصر خانه با اندیس یک را نمایش می‌دهد. اندیس خانه اول، صفر است.

```
b = "Hello, World!"
print(b[2:5])
```

در تکه کد بالا خانه شماره ۲ تا ۵ (بدون ۵) نمایش داده می‌شود (سه کاراکتر). اندیس منفی هم می‌توان داشت که از انتهای رشته محاسبه می‌کند. خانه آخر دارای اندیس ۱- است. کد زیر با شمارش از انتها خانه شماره ۵ تا ۲ (بدون ۲) را مشخص می‌کند.

```
b = "Hello, World!"
print(b[-5:-2])
```

تابع len()، طول رشته را برمی‌گرداند.

```
a = "Hello, World!"
print(len(a))
```

چند تابع مفید برای متغیرهای رشته‌ای

- حذف فاصله از دو سر رشته با تابع strip()

```
a = " Hello, World! "
print(a.strip()) # returns "Hello, World!"
```

- تبدیل به حروف کوچک lower()
- تبدیل به حروف بزرگ upper()
- جایگزینی یک زیر رشته با یک زیر رشته دیگر replace()

```
a = "Hello, World!"
print(a.replace("H", "J"))
```

- تقسیم رشته split()

```
a = "Hello, World!"
print(a.split(",")) # returns ['Hello', ' World!']
```

- بررسی وجود یک زیر رشته در زیر رشته دیگر in و not in

```
txt = "The rain in Spain stays mainly in the plain"
x = "ain" in txt
print(x)
```

- اتصال دو رشته با +
- ترکیب عدد با رشته با تابع format()

```
age = 36
txt = "My name is John, I am " + age
age = 36
txt = "My name is John, and I am {}"
print(txt.format(age))

# ----------------

quantity = 3
itemno = 567
price = 49.95
myorder = "I want {} pieces of item {} for {} dollars."
print(myorder.format(quantity, itemno, price))

# ----------------

quantity = 3
itemno = 567
price = 49.95
myorder = "I want to pay {2} dollars for {0} pieces of item {1}."
print(myorder.format(quantity, itemno, price))
```

کاراکترهای خاص

برای نمایش کاراکترهای خاص باید از کاراکتر '\' قبل از آنها استفاده نمود.

```
txt = "We are the so-called "Vikings" from the north."

txt = "We are the so-called \"Vikings\" from the north."
```

کاراکتر خاص	کد
نقل قول یگانه	\'
\	\\
خط جدید	\n
tab	\t

متغیر بولی

متغیرهایی که مقدار True یا False را در خود ذخیره می‌کنند. توابع زیادی در پایتون وجود دارد که مقدار بولی باز می‌گردانند. مثل تابع ()isinstance که به کار می‌رود تا بررسی کند که یک متغیر از نوع خاصی هست یا نه.

```
x = 200
print(isinstance(x, int))
```

۳- عملگرها در پایتون

- **عملگر حسابی**

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x – y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

- **عملگر انتساب**

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
//=	x //= 3	x = x // 3
**=	x **= 3	x = x ** 3
&=	x &= 3	x = x & 3
|=	x |= 3	x = x | 3
^=	x ^= 3	x = x ^ 3
>>=	x >>= 3	x = x >> 3
<<=	x <<= 3	x = x << 3

- **عملگر مقایسه**

Operator	Name	Example
==	Equal	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

- **عملگر منطقی**

Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4

not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

- **عملگر این‌همانی (identity operator)**

Operator	Description	Example
is	Returns True if both variables are the same object	x is y
is not	Returns True if both variables are not the same object	x is not y

- **عملگر عضویت**

Operator	Description	Example
in	Returns True if a sequence with the specified value is present in the object	x in y
not in	Returns True if a sequence with the specified value is not present in the object	x not in y

- **عملگر بیتی**

Operator	Name	Description
&	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 leftmost bit in from the left, and let the rightmost bits fall off

۴- آرایه‌ها (مجموعه‌ها) در پایتون

- **List یا لیست**

یک مجموعه از اشیا که ترتیب دارد و قابل تغییر است. عنصر تکراری هم می‌تواند در آن وجود داشته باشد. با براکت، []، تعریف می‌شود. با اندیس می‌توان به عناصر آن دسترسی داست. اولین عنصر دارای اندیس صفر است. اندیس منفی هم می‌تواند دریافت کند که از انتها می‌شمارد.

عنصر اخر دارای اندیس ۱- است.

کار با اندیس‌ها

```
thislist = ["a", "b", "c", "d", "e", "f"]
print(thislist[1])   # b
print(thislist[-1])  # f
print(thislist[2:5])   # ['c', 'd', 'e']
```

عنصر پنجم نمایش داده نمی‌شود.

```
print(thislist[:4])
```

از عنصر با اندیس صفر تا با عنصر با اندیس سه

```
print(thislist[2:])
```

از عنصر با اندیس دوم تا آخر

```
print(thislist[-4:-1])   #['c', 'd', 'e']
```

عنصر آخر در نظر گرفته نمی‌شود.

```
thislist[1] = "ss"
```

جایگزینی b با 'ss'

```
for x in thislist:
  print(x)
```

استفاده از حلقه برای دسترسی به عناصر

```
if "c" in thislist:
  print("Yes, 'c' is in the list")
```

```
print(len(thislist))  #6
```

تعداد عناصر آرایه

```
thislist.append("g")
```

اضافه نمودن عنصر به انتهای لیست

```
thislist.insert(1, "bb")
```

درج "bb" در اندیس یکم. طول لیست زیاد می‌شود.

```
thislist.remove("c")
```

حذف عنصر

```
thislist.pop()
```

حذف عنصر با کمک اندیس. در پرانتز می‌تواند اندیس باشد. در صورت خالی بودن پرانتز آخرین عنصر حذف می‌شود.

```
del thislist[0]
```

حذف عنصر

```
del thislist
```

از بین بردن لیست

```
thislist.clear()
```

خالی نمودن لیست؛ لیست تهی است اما وجود دارد.

کپی نمودن لیست

دستور زیر نادرست است

```
list2 = list1
```

روش اول

```
mylist = list(thislist)
```

ایجاد لیست جدید به نام mylist که کپی thislist است.

روش دوم

```
mylist = thislist.copy()
```

```
# ------------------------------------------------
```

اضافه نمودن یک لیست به لیست دیگر

```
list1 = ["a", "b" , "c"]
list2 = [1, 2, 3]
```

روش اول

```
list3 = list1 + list2
```

روش دوم

```
for x in list2:
  list1.append(x)
```

روش سوم

```
list1.extend(list2)
```

```
# -------------------------------
```

ایجاد لیست با استفاده از تابع سازنده list()

```
thislist = list(("apple", "banana", "cherry")) # note the double round-brackets
print(thislist)
```

- **Tuple یا چندتایی**

یک مجموعه از اشیا که ترتیب دارد و قابل تغییر نیست. عنصر تکراری هم می‌تواند در آن وجود داشته باشد. با پرانتز، ()، تعریف می‌شود. با اندیس می‌توان به عناصر آن دسترسی داست. اولین عنصر دارای اندیس صفر است. اندیس منفی هم می‌تواند دریافت کند که از انتها می‌شمارد. عنصر اخر دارای اندیس ۱- است. کارکرد اندیس در ان مشابه لیست است.

```
thistuple = ("apple", "banana", "cherry")
print(thistuple)
```

با تابع len() طول آن را می‌توان یافت. مشابه لیست می‌توان حلقه و دستورات شرطی را برای آن به کار برد. دستور del تاپل را از بین می‌برد.

برای ایجاد تاپل با یک عضو حتما در انتهای آن عضو باید ویرگول گذاشت. در غیر این صورت نوع داده ایجاد شده تاپل نخواهد بود.

```
thistuple = ("apple",)
```

هیچ عنصری نمی‌توان از آن حذف کرد، یا به آن اضافه نمود و یا تغییر داد. برای این کار باید آن را به لیست تبدیل نمود. تغییر لازم را بر روی آن انجام داد و دوباره برگرداند.

```
x = ("apple", "banana", "cherry")
y = list(x)
y[1] = "kiwi"
x = tuple(y)
```

دو تاپل را می‌توان با دستور + تجمع نمود و در یک تاپل جدید ریخت. تابع tuple() یک سازنده است که می‌توان با آن یک تاپل ایجاد نمود.

```
thistuple = tuple(("apple", "banana", "cherry")) # note the double round-brackets
```

- **Set یا مجموعه**

مجموعه‌ای نامرتب و بدون اندیس که با {} نمایش داده می‌شود.

```
thisset = {"apple", "banana", "cherry"}
print(thisset)
```

اعضای آن اندیس ندارند، اما می‌توان با استفاده از دستور in در حلقه به آنها دسترسی داشت.

```
thisset = {"apple", "banana", "cherry"}
for x in thisset:
  print(x)
```

زمانی که یک متغیر مجموعه ساخته شد. اعضایش قابل تغییر نیست. اما با تابع add() و update() می‌توان به ترتیب یک یا چند عنصر اضافه کرد.

```
thisset = {"apple", "banana", "cherry"}
thisset.add("orange")

thisset.update(["orange", "mango", "grapes"])
```

برای به دست آوردن تعداد عناصر؛ از تابع len() استفاده می‌شود. برای حذف عنصر از مجموعه از تابع remove() و discard() استفاده می‌شود. اگر عنصر در مجموعه نباشد، تابع remove() خطا می‌دهد. اما discard() اینگونه نیست. تابع pop() اخرین عضو را حذف می‌کند (البته آخرین عضو معلوم نیست کدام باشد). تابع clear() مجموعه را خالی می‌کند. دستور del مجموعه را نابود می‌کند.

```
thisset = {"apple", "banana", "cherry"}
thisset.remove("banana")
```

```
thisset = {"apple", "banana", "cherry"}
thisset.discard("banana")
```

```
x = thisset.pop()
```

```
thisset.clear()
```

```
del thisset
```

دو دستور union() و update() برای ادغام دو مجموعه کاربرد دارد. عناصر تکراری با این دو دستور حذف می‌شود.

```
set1 = {"a", "b" , "c"}
set2 = {1, 2, 3}

set3 = set1.union(set2)
```

```
set1 = {"a", "b" , "c"}
set2 = {1, 2, 3}
set1.update(set2)
```

برای ایجاد یک مجموعه تابع سازنده set() نیز وجود دارد.

```
thisset = set(("apple", "banana", "cherry")) # note the double round-brackets
```

- **Dictionary**

مجموعه‌ای نامرتب، قابل تغییر و اندیس‌گذاری شده که هر عنصر آن شامل کلید و مقدار است و با {} ساخته می‌شود.

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

برای رسیدن به مقدار یک عنصر می‌توان کلید عنصر را به نحو زیر فراخوانی کرد و یا از تابع get() بهره برد.

```
x = thisdict["model"]
```

```
x = thisdict.get("model")
```

می‌توان با استفاده از کلید مقدار یک عنصر را هم تغییر داد.

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict["year"] = 2018
```

از حلقه for هم می‌توان استفاده نمود. دستورات زیر کلیدها را یکی‌یکی نمایش می‌دهد.

```
for x in thisdict:
  print(x)
```

دستورات زیر مقادیر را یکی‌یکی نمایش می‌دهد.

```
for x in thisdict:
  print(thisdict[x])
```

یا می‌توان برای نمایش مقادیر به صورت زیر نوشت

```
for x in thisdict.values():
  print(x)
```

با استفاده از تابع item() همزمان می‌توان به کلید و مقدار هر عنصر دست یافت.

```
for x, y in thisdict.items():
  print(x, y)
```

برای بررسی وجود یک کلید در دیکشنری از دستور in می‌توان بهره برد

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
if "model" in thisdict:
  print("Yes, 'model' is one of the keys in the thisdict dictionary")
```

تابع len() برای شمارش تعداد عنصر است.

با استفاده از یک کلید جدید می‌توان یک عنصر جدید به دیکشنری اضافه نمود.

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
```

```
}
thisdict["color"] = "red"
print(thisdict)
```

تابع pop() یک عنصر با یک کلید مشخص را حذف می‌کند.

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict.pop("model")
print(thisdict)
```

از دستور del هم برای حذف می‌توان بهره برد

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
del thisdict["model"]
```

دستور زیر تمام دیکشنری را از بین می‌برد.

```
del thisdict
```

اما دستور زیر دیکشنری را خالی می‌کند

```
thisdict.clear()
```

دیکشنری لانه‌ای

عناصر دیکشنری خود دیکشنری هستند.

روش اول ساخت دیکشنری لانه‌ای

```
myfamily = {
  "child1" : {
    "name" : "Emil",
    "year" : 2004
  },
  "child2" : {
    "name" : "Tobias",
    "year" : 2007
  },
```

```
  "child3" : {
    "name" : "Linus",
    "year" : 2011
  }
}
```

روش دوم

```
child1 = {
  "name" : "Emil",
  "year" : 2004
}
child2 = {
  "name" : "Tobias",
  "year" : 2007
}
child3 = {
  "name" : "Linus",
  "year" : 2011
}

myfamily = {
  "child1" : child1,
  "child2" : child2,
  "child3" : child3
}
```

تابع سازنده ()dic برای ساخت دیکشنری وجود دارد:

```
thisdict = dict(brand="Ford", model="Mustang", year=1964)
# note that keywords are not string literals
# note the use of equals rather than colon for the assignment
print(thisdict)
```

۵- توضیحات

برای ایجاد توضیح در یک خط از کاراکتر # استفاده می‌شود. تمام نوشته‌های بعد از این کاراکتر تا انتهای جمله توضیح در نظر گرفته می‌شود.

```
#This is a comment
print("Hello, World!")
```

برای ایجاد توضیحات چندخطی از سه علامت نقل قول در ابتدا و انتهای نوشته استفاده می‌شود.
نقل قول یگانه و دوگانه تفاوتی ندارد.

```
"""
This is a comment
written in
more than just one line
"""
```

۶- دستورات شرطی

- دستورات زیر مثالی از کاربرد if است:

```
a = 33
b = 200
if b > a:
  print("b is greater than a")
```

پس از دستورات شرطی و حلقه یک بلوک دستوری وجود دارد. این بلوک با تورفتگی در پایتون مشخص می‌شود. میزان تورفتگی دلخواه است. اما میزان آن برای عناصر یک بلوک یکسان باید باشد.

- دستور elif

```
a = 33
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
```

- دستور else

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

- دستور if کوتاه

```
if a > b: print("a is greater than b")
```

- دستور if ... else کوتاه

```
a = 2
b = 330
print("A") if a > b else print("B")
```

- دستور شرطی تو در تو

```
if x > 10:
  print("Above ten,")
  if x > 20:
    print("and also above 20!")
  else:
    print("but not above 20.")
```

۷- دستور while

مثالی از دستور while در زیر آمده است

```
i = 1
while i < 6:
  print(i)
  i += 1
```

- دستور break و continue

```
i = 1
while i < 6:
  print(i)
  if i == 3:
    break
  i += 1

# ----------------------------

i = 0
while i < 6:
  i += 1
  if i == 3:
    continue
  print(i)
```

- دستور else در حلقه‌ها

```
i = 1
while i < 6:
  print(i)
  i += 1
```

```
else:
  print("i is no longer less than 6")
```

۸- دستور for

- مثال‌هایی از دستور for

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```

دستور for برای یک رشته

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

- استفاده از تابع ()range برای داشتن شمارنده در حلقه

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

شمارش از ۰ تا ۵

```
    for x in range(2, 6):
      print(x)
```

شمارش از ۲ تا ۵

```
    for x in range(2, 30, 3):
      print(x)
```

شمارش از ۲ تا ۲۹ با گام ۳

۹- تمرین

- بدون استفاده از کتابخانه‌های پایتون برنامه‌های زیر را بنویسید

۱- آرایه زیر موجود است بزرگترین عنصر آن را بیابد

```
n=[23, 3, 45, 96, 11, 7, 36, 9, 27, 85]
```

۲- آرایه بالا را مرتب کنید

۳- دهمین عدد رشته فیبوناچی را بیابید

۴- رشته زیر را معکوس کنید

```
Str1='Semnan University'
```

۵- مقدار ب.م.م. دو عدد زیر را بیابید

```
n = 654, m = 744
```

۶- چهار عدد کامل را بیابید

عدد کامل، یک عدد صحیح مثبت است که برابر با مجموع مقسوم‌علیه‌های سرهٔ خود (همهٔ مقسوم‌علیه‌های غیر از خود عدد) باشد.

۷- بررسی نماید آیا عدد n=23543 اول است. در صورت اول بودن 'yes' چاپ کند.

۸- اعداد اول بین ۱۰۰۰ تا ۲۰۰۰ را چاپ نماید

۹- برای n=? خروجی زیر را چاپ کنید (به طور مثال ۵)

```
*****
_****
__***
___**
____*
```

۱۰- از رشته زیر حرف e را حذف کند و جای آن zs قرار دهد.

```
Str1='Semnan University'
```