**💡İpuçları:**

* Fark ettiyseniz , ondalık sayı için 👉🏻 **.**değil 👉🏻 kullandık: .**,**3.14
* İfadeyi tırnak işaretleri ile çevrelemek onu stringtür içinde yapar. İlerleyen derslerde değişmez türleri (int, string, tuple vb.) göreceğiz.

What is PEP 8?  
A: PEP stands for Python Enhancement Proposal. PEP 8 is a coding convention, a set of recommendation, about how to write your Python code more readable. In other words, PEP 8 is a document that gives coding conventions for the Python code comprising the standard library in the main Python distribution.

S: PEP 8 nedir?  
C: PEP, Python Geliştirme Önerisi anlamına gelir. PEP 8, Python kodunuzu nasıl daha okunaklı yazacağınızla ilgili bir kodlama kuralı, bir dizi öneridir. Başka bir deyişle, PEP 8, ana Python dağıtımında standart kitaplığı içeren Python kodu için kodlama kuralları veren bir belgedir.

------------------------------------------------------------------

### Some Important PEP 8 Rules

We will show you some important PEP 8 traditional rules that you can follow.

* Limit all lines to a maximum of **79 characters**. For flowing long blocks of text with fewer structural restrictions (docstrings or comments), the line length should be limited to **72 characters**. During this course, we will learn some ways of reducing the length of lines.
* **Spaces**are the preferred indentation method. **Tabs** should be used solely to remain consistent with code that is already indented with tabs. Python 3 disallows mixing the use of tabs and spaces for indentation.
* Avoid extraneous **whitespaces** in the following situations:

Immediately inside parentheses, brackets or braces :

**YES** : spam(meat[1], {milk: 2}) , **NO** : spam( meat[ 1 ], { milk: 2 } )

Between a trailing comma and a following close parenthesis :

**YES** : df[0,] or foo = (2,) , **NO** : df[0, ] or foo = (2, )

Immediately before a comma, semicolon, or colon :

**YES** : if y == 3: print x, y; x, y = y, x , **NO** : if y == 3 : print x , y ; x , y = y , x

Immediately before the open parenthesis that starts the argument list of a function call:

**YES** : print('peace') , **NO** : print ('peace')

More than one space around an assignment (or other) operator to align it with another:

**YES**

1

2

3

4

x = 3

y = 4

long\_vars = 5

**NO**

1

2

3

4

x = 3

y = 4

long\_vars = 5

* Avoid trailing whitespace anywhere. Because it's usually invisible, it can be confusing: e.g. a backslash followed by a space and a newline does not count as a line continuation marker.
* Always surround these binary operators with a single space on either side: assignment (=), augmented assignment (+=, -=, etc.), comparisons (==, <, >, !=, <>, <=, >=, in, not in, is, is not), Booleans (and, or, not).

Failure to follow the basic rules of PEP 8 does not make your program wrong or unable to work. In the near future, you will learn a lot about Python and become a more skilled programmer, but it will always be important to follow the code style.

There's nothing to worry about following PEP 8. You don't need to learn the traditional PEP 8 rules all at once right away. When you need it, you can open and read it now and then. We will also show you some PEP 8 conventions throughout this course.

Bazı Önemli PEP 8 Kuralları

Size izleyebileceğiniz bazı önemli PEP 8 geleneksel kurallarını göstereceğiz.

* Tüm satırları maksimum **79 karakterle** sınırlayın . Daha az yapısal kısıtlamaya sahip uzun metin blokları (belge dizileri veya yorumlar) için satır uzunluğu **72 karakterle** sınırlandırılmalıdır . Bu kurs sırasında, çizgilerin uzunluğunu azaltmanın bazı yollarını öğreneceğiz.
* **Boşluklar** tercih edilen girinti yöntemidir. **Sekmeler** yalnızca, zaten sekmelerle girintili olan kodla tutarlı kalmak için kullanılmalıdır. Python 3, girinti için sekme ve boşluk kullanımının karıştırılmasına izin vermez.
* Aşağıdaki durumlarda gereksiz **boşluklardan kaçının:**

Hemen parantez, parantez veya parantez içinde:

**EVET** : spam(meat[1], {milk: 2}), **HAYIR** : spam( meat[ 1 ], { milk: 2 } )

Sondaki virgül ile aşağıdaki yakın parantez arasında:

**EVET** : df[0,]veya foo = (2,), **HAYIR** : df[0, ]veyafoo = (2, )

Virgül, noktalı virgül veya iki nokta üst üste işaretinden hemen önce:

**EVET** : if y == 3: print x, y; x, y = y, x, **HAYIR** : if y == 3 : print x , y ; x , y = y , x

Bir işlev çağrısının bağımsız değişken listesini başlatan açık parantezden hemen önce:

**EVET** : print('peace'), **HAYIR** : print ('peace')

Bir atama (veya başka) operatörün çevresinde, onu bir başkasıyla hizalamak için birden fazla boşluk:

* Always surround these binary operators with a single space on either side: assignment (=), augmented assignment (+=, -=, etc.), comparisons (==, <, >, !=, <>, <=, >=, in, not in, is, is not), Booleans (and, or, not).
* Bu ikili operatörleri her zaman her iki tarafta tek bir boşlukla çevreleyin: atama (=), artırılmış atama (+=, -=, etc.), karşılaştırmalar (==, <, >, !=, <>, <=, >=, in, not in, is, is not), Boolean (and, or, not).

------------------------------------------------------

Q: What are the comments and how do you write it in Python?  
A: Comments are used to explain code when the basic code itself isn't clear. Python ignores comments, and so will not execute code in there, or raise syntax errors for plain English sentences. Comments in Python start with a # character. '#' character converts all subsequent characters to the comment form that Python does nothing.  
  
# this is a single line comment  
  
print("Hello World!") # this is an inline comment

S: Yorumlar nelerdir ve bunu Python'da nasıl yazıyorsunuz?  
C: Temel kodun kendisi net olmadığında, yorumlar kodu açıklamak için kullanılır. Python yorumları yok sayar ve bu nedenle orada kod yürütmez veya düz İngilizce cümleler için sözdizimi hataları oluşturmaz. Python'da yorumlar # karakteriyle başlar. '#' karakteri, sonraki tüm karakterleri Python'un hiçbir şey yapmadığı yorum formuna dönüştürür.  
  
# this is a single line comment  
  
print("Hello World!") # this is an inline comment

\_\_doc\_\_ bak!!!!!

### Conventional (PEP 8) Naming Rules

If variables have poor names, even your own code may seem unclear to you in a couple of months. Now let's learn how to choose good names for our variables in accordance with PEP 8 rules:

* Choose lowercase words and use underscore to split the words:
* **price = [22, 44, 66]**,
* **low\_price = 12.00**
* Do not use the characters 'l' (lowercase letter el), 'O' (uppercase letter oh), or 'I' (uppercase letter eye) as single-character variable names. In some fonts, these characters are indistinguishable from the numerals one and zero. If you want to use 'l', use 'L' instead.
* **l = 'It is not correct use'**,
* **O = "It's also incorrect use"**

**⚠️Avoid ! :**

* Do not use specific Python keywords (name of a function or phrase) as a name, like sum, max, min, in, or, for, etc.
* Use a sensible name. The variable name needs to be legible and meaningful and explain to the reader what types of values will be stored in it.
* **figures = 'this is better'**,
* **f = 'it is not meaningful'**
* Don't choose too common names. Use a name to describe the meaning of the variable. However, try to limit it to no more than 3 words.
* If the word you intend to choose is long, try to find the most common and expected short form to make it easy to predict later.

### Geleneksel (PEP 8) Adlandırma Kuralları

Değişkenlerin adları kötüyse, kendi kodunuz bile birkaç ay içinde size belirsiz görünebilir. Şimdi değişkenlerimiz için nasıl iyi isimler seçeceğimizi PEP 8 kurallarına göre öğrenelim:

* Küçük harfli kelimeler seçin ve kelimeleri bölmek için alt çizgi kullanın:
* **price = [22, 44, 66]**,
* **low\_price = 12.00**
* Karakterleri 'l'(küçük harf el), 'O'(büyük harf oh) veya 'I'(büyük harf göz) tek karakterli değişken adları olarak kullanmayın. Bazı yazı tiplerinde bu karakterler bir ve sıfır rakamlarından ayırt edilemez. Kullanmak istiyorsanız 'l', 'L'bunun yerine kullanın.
* **l = 'It is not correct use'**,
* **O = "It's also incorrect use"**

**⚠️ Kaçının ! :**

* Belirli Python anahtar sözcüklerini (bir işlevin veya tümcenin adı) ad olarak kullanmayın sum, max, min, in, or, for, vb.
* Mantıklı bir ad kullanın. Değişken adı okunaklı ve anlamlı olmalı ve okuyucuya içinde ne tür değerlerin saklanacağını açıklamalıdır.
* **figures = 'this is better'**,
* **f = 'it is not meaningful'**
* Çok yaygın isimler seçmeyin. Değişkenin anlamını açıklamak için bir ad kullanın. Ancak, bunu en fazla 3 kelimeyle sınırlamaya çalışın.
* Seçmeyi düşündüğünüz kelime uzunsa, daha sonra tahmin etmeyi kolaylaştırmak için en yaygın ve beklenen kısa formu bulmaya çalışın.

| **alue1** | **Logic** | **Value2** | **Returns** |
| --- | --- | --- | --- |
| True | and | True | True |
| True | and | False | False |
| False | and | False | False |
| False | and | True | False |
| True | or | True | True |
| True | or | False | True |
| False | or | False | False |
| False | or | True | True |