#Pythonda nəyi niyə öyrənim? (part 2)

Öncəki Bölüm: <https://www.linkedin.com/feed/update/urn:li:activity:6996492685167484929/>

#ControlFlow structures(Axın tənzimləməsi strukturları):

1. İf, Elif, Else ifadələrini part 1-də qeyd etdiyim – boolean dəyişənlər, qarşılaşdırma, aidlik və identity operatorları ilə birlikdə istifadə edərik. Tez-tez qarşılaşdığım çalışmalarda problemlərə yol açan əməllər bunlardır:

* İf, Elif, Else ifadələrində elif işlədə biləcəyimiz yerdə nested if istifadəsi çox edilən oxunaqlığı azaldan əməllərdən biridir.
* Oxunaqlığı azaldan eləcə də, #executetime -ni (ki, əməllərdə ləngimələrə yol açır) artıran digər məqsədəuyğun olmayan əməllərdən biri də eyniməqsədli olan if ifadələrinin elif ilə işlənə bilməsi mümkün olan hallarda alt-alta düzülməsidir.
* is identity operatoru ilə == qarşılaşdırma operatorunun yalnış olaraq qarışdırılması və eləcə də müqayisəsi

Əlavələrinizi şərhlərdə gözləyirəm.

İf, Elif, Else haqqında:

<https://www.learnpython.dev/02-introduction-to-python/110-control-statements-looping/10-if-else-elif/>

is identity operatoru haqqında:

<https://python-istihza.yazbel.com/islecler.html>

1. Dövrlər(#loops) – #For və #While dövrləri ilə insanlar yeni başlarkən belə sual verə bilirlər: While dövrünü tam olaraq necə istifadə edək? Buna cavab vermək üçün For dövrü ilə müqayisəsinə baxaq:

* Əgər bir kod yazarsaq və dövrdəki şərt sonsuz olaraq ödənərsə bu zaman for sonsuz olaraq çalışar. While dövrü compile time error xətası verər.
* İterasiya üçün defolt funksiya For dövrü üçün mövcuddur (range() və ya xrange() ). While dövrü üçün defolt iterasiya funksiyası mövcud deyildir.
* İnitalizasiya (Başlama) strukturu : For dövründə şərt ödənən kimi dərhal dayanar. Təkrarlanma mümkün deyildir. While dövrü hər bir dövr başlanğıcı sətrində təkrarlanar. Şərt ödəndikdən sonra onun (şərtin) ödəndiyi təsdiqlənənə qədər təkrarlanar.
* Generatorlar funksiyalar ilə For dövrü birbaşa şərt alaraq çalışar. While dövrü üçün birbaşa olaraq bu mümkün deyildir.
* Sürət baxımından müqayisədə For dövrü While dövründən daha sürətlidir.

Beləliklə, for dövrünü avtomatik sükan mexanizmi olan bir avtomobilə bənzədə bilərik, keçidlər səlis və hərşey nəzarət altındadır. While dövrü isə əllə idarəetmə mexanizminə malik avtomobil kimidir. Hər şey #developer (tərtibatçıdan) -dən asılıdır. Onun daxilində iterasiya və generasiya əməllərini özü qura bilər. Break continue və pass əmrlərini işlədərək xəyalındakı (avtomobilə və sürətə ☺) nəticəyə sahib ola bilərsən.

İterator və Generatorlar haqqında məqalə: <https://medium.com/python-yaz-lar/python-generator-ve-i-teratorler-a53e59f7c5b1>

1. Break, continue və pass əmrlərini müxtəlif yerlərdə işlədirik. Lakin, ən təhlükəli vəziyyət continue ilə yarana bilir. Çünki yerləşməsindən asılı olaraq dövrlərdə iterasiyanı başa alaraq istənilən nəticədə xətalara yol aça bilir. Ona görə də düzgün istifadə edilməsi və diqqətlə işlədilməsi lazımdır. Duz yeməyə dad verir, amma lazımı yer lazımı miqdar məsələsini buna misal çəkə bilərəm. (Yoxsa o yemək zay ola bilir. 😀)

Ardını növbəti partda paylaşacam. Şərhlər əlavələriniz üçün açıqdır. ☺