**Giriş**

Genel olarak her program Veri girişi, işlem ( hesaplamalar), ve veri çıkışı olmak üzere üç aşamadan oluşmaktadır.

Veri Girişi: Gerekli kütüphanelerin eklendiği, değişkenlerin tanımladığı ve başlangıç değerlerinin atandığı yerdir.

İşlem ve Hesaplamalar : Girdilerden çıktıların üretildiği yerdir.

Veri Çıkışı ( Program Çıktısı ) : Çıkış verisinin içeriği ve formatı önemlidir. İçeriği, elde edilen veya edilecek bilgiyi, formatı, çıkış içeriğinin sunum şeklini gösterir.

>>> a=int ( input ( 'Kenar Uzunluğu : '))

Kenar Uzuluğu : 5

>>> Alan : a\*a

>>> print ("Karenin Alanı= ",Alan)

Karenin Alanı= 25

>>>

**Python Giriş Çıkış İşlem ve Komutları**

Python'da veri girişi için **input** komutu kullanılır. Ekrana bir mesaj ya da değişken içeriği yazdırmak için ise **print** komutu kullanılır.

**Input Komutunun Kullanımı**

ad= input ('Adınız?') Bu kullanımda kalvyeden girilen veri , string ad değişkeninde tutulur.

nu= int (intput( ' Numaranız?' )) Bu kullanımda klavyeden girilen veri, int ( tamsayı) tipindeki nu değişkeninde tutulur.

b= float ( intput (' Maaşiniz?')) Bu kullanımda klaveyden girilen veri, float ( kesirlisayı) tipindeki b değişkeninde tutulur.

**Print (Veri Çıkışı/Ekrana Yazdırma) Komutunun Kullanımı**

**Python Diline Ait Anahtar Kelimeler Keywords**

Python diline ait komutlar değişken ismi olarak kullanılmaz.

* as = Bir modül veya nesneyi farklı bir isimle temsil etmekte kullanılır.
* assert = Test yazılımlarında bir koşulun mutlaka sağlanması gerektiğini vurgular.
* is = Eşit mi anlamında sorgu operatörüdür.
* from = Bir modülün sadece belirtilen fonksiyonlarını içe aktarmak için kullanılır.
* yield = Return gibi işleve sahiptir. Farkı; değeri tek tek geri döndürür.

**Mesajlar**

Kullanıcı ile iletişim kurmak için kullanılır. Kullanıcıyı yönlendirmek amacıyla çift tırnak ""arasına ( " bir sayı giriniz" gibi) yazılan ifadelere mesaj denir. Mesajları ekranda göstermek için print komutu kullanılır.

**Escape ( Çıkış ) Karakterleri**

Ekrana basılıcak mesajın formatını değiştirmek/ayarlamak için escape karakterlerini print komutu ile birikte kullanılır.

* \n = Bir alt satıra geç.
* \t = Bir tab boşluk bırak
* \v = Dikey bir tab boşluk bırak
* \b = Backspace print("12\b3") = 13
* \r = Satır başına dönüş

**Sayı Dönüşümleri**

Her ne kadar gündelik hayatta onlu (desimal) sayı sistemini kullanıyor olsakta bilgisayar diğer sayı sistemleri ile de işlem yapar. Bu sayı sistemlerinin dönüşümlerinde print komutu, dönüşüm fonksiyonları ile kullanılır.

**Değişkenler ve Değer Ataması**

İçerisinde veri sakladığımız, ismini ve şeklini bizim belirlediğimiz bellek alanlarına **variable** ( değişken ) adı verilmektedir.

Bilgilerin geçici olarak tutulduğu yere **bellek** denir.

**Python Veri Tipleri**

Python’da diğer dillerin aksine değişken tanımlanırken veri tipinin belirtilmesine gerek yoktur. Fakat her değişkenin belirli bir veri tipine sahip olduğu unutulmamaldır. Python veri tipinin tanımlamasını otomatik yapar.

Pythonda mantıksal ( iki durumun karşılaştırılması gibi ) sorgulamalar **bool** veri tipinde sonuç üretirken; string ( sözel) ifadeler **str** veri tipinde sonuç üretir. Sayılar ise üç farklı veri tipinde tutulur. Tam sayılar için **int**, kesirli sayılar için **float** ve karmaşık ( kompleks) sayılar için **complex** olmak üzere üç adet sayısal veri tipi kullanılır. Python’da bir değişken veya fonksiyonun veri tipini öğrenmek için **type()** fonksiyonu kullanılır.

**Değişkene İsim Verme Kuralları**

* Değişkenin ilk karakteri mutlaka harf olmalı
* Değişken isimleri içinde alt çizgi hariç, boşluk veya diğer özel karakterler bulunmamalıdır.
* Türkçe karakter bulundurmamaya özen gösterilmeli.
* Değişken isimleri, kullanılın programın komutlarını içeremez. O programla diline ait reserved words veya keywords kullanılmaz.
* Python'da büyük-küçük har ayrımı yapan case sensitive bir dildir. ( Yani AD ile ad birbiriden farklı değişkenledir.)

**İki Değişkenin İçeriğini Birbirine Aktarma ( Yer Değiştirme İşlemi )**

a, b = b, a #a=b ve b=a yapılmış oldu.

**Değişkenlerin Faliyet Alanları**

Bir değişkenin faliyet alanı onun genel ya da yerel olarak tanımlanmasına neden olur. Eğer bir değişken, ana fonksiyonda/programda yer alıyorsa **global**(genel), bir alt fonksiyonda ya da kod bloğu içerisinde tanımlanıyorsa **local**(yerel) değişken olarak adlandırılır. Ana fonksiyonda tanımlanan bir değişken alt fonksiyonda da faliyet gösterir fakat tersi doğru değildir. Yerel bir fonksiyonu global olarak tanımlamak için başına 'global' anahtar kelimesi getirilir.

**Veri Tipi Dönüşümü**

* int(): String veya float veri tipini int (tamsayı) veri tipine dönüştürür.
* float():String veya int veri tipini float (kesirli sayı) veri tipine dönüştürür.
* str():İfadeyi string veri tipine dönüştürür.

**SABİTLER**

Sabit değeri sayısal ve stirng/sözel olmak üzere ikiye ayrılır.Çift tırnak işaretleri arasına yazılan her türlü mesaj aslında string bir sabittir. Değişkenin değeri programın çalışması boyunca hiç değişmeyecekse sabit tanımlaması yapılır. Pythonda sabit tanımlaması için özel bir komut yoktur. Fakat bir gelenek olarak sabit tanımlaması büyük harfle yazılır. Örneğin PI=3.14

**Enum Sabiti ( Enumeration Constant)**

Kullanıcıların tanımladığı, sıralı semboller kümesi olarak tanımlayabileceğimiz enum sabiti, normalde mevcut olmayan, bize özel verileri sıralamak amacıyla kullanılır.Enum tanımı için tamsayı tipindeki sembolik sabitler kümesi diyebiliriz. Eğer tanımlama yapılırken değişkenlere bir değer ataması yapılmazsa ilk sabitin değeri 1 olarak kabul edilir. Sabit\_ismi=Enum ('Sabit\_ismi' , [a,b,c,d])

#sabit\_ismi=Enum("sabit-ismi",[A,B,C,D,E,.....])

from enum import Enum

gunler=Enum("gunler",["Pazartesi","Salı","Çarşamba","Perşembe","Cuma","Cumartesi","Pazar"])

print("Pazartesi :",gunler.Pazartesi.value)

#value ile enum sabitindeki bir eleman sırası öğrenilmektedir.