**Errors and Exceptions (Hatalar ve İstisnalar) :**

Günlük yaşamımızda olduğu gibi kod yazımında da yapıan hatanın cinsi onun affedilebilirliğini veya düzeltilebilirliğini belirler. Bir programla dili ile kod yazımında karşılaşılan olası hatalar; mantık hataları,kod yazım hataları ve çalışma zamanı hataları olmak üzere üçe ayrılır.

**Logical Errors (Mantık Hataları) :**

Mantık hatalarını tespit etmek oldukça zordur. Çünkü programı çalıştırdığımızda herhangi bir hata ile karşılaşmayız ama program beklenen sonuçları vermez. Bu gibi durumlarda genellkle en başa dönmek algoritmayı yeniden incelemek hatta problemi yeniden gözden geçirmek gerekir. Programı yazarken doğru kabul ettiğiniz komutlar, aslında başka şekilde çalışıyor olabilir.Bu tip hataları gidermek için program (deburg yapılarak) "adım-adım" çalıştırılarak hatalı kod satırı yakalanmaya çalışılır.

**Syntax Errors (Kod Yazım Hataları) :**

Modern programlama dillerinde sytntax hatalarının tespiti kolaydır. Zaten Python yorumlayıcısı satır satır kod yazım denetimi yaptığı için olası hatayı düzeltmeden bir alt satıra geçişe izin vermez.

**Run Time Errors (Çalıma Zamanı Hataları) :**

Run time errors exceptions (istisna) ile afade edilmektedir. Bu nedenle hata yakalama kavramı bazı kaynaklarda exceptions handling (istisna işleme) olarak geçer. Run time errors, programın çalışamsı sırasında beklenmeyen bir durum sonucunda oluşan hatalara denir. Örneğin dosya okuma işleminde, okunucak dosyanın olmaması, istenilen veriileri bir çıkış birimine gönderememesi, bölme işleminde paydanın 0 olması gibi. Bu gibi durumlarda program hata verir ve bu tip hatalara karşı alınan önlemler de exception handling(hata yakalama) olarak adlandırılır.

**Exception Handling (Hata Yakalama) :**

Hata sınıflandırmasını yaptıktan sonra programcının üzerinde duracağı hata türü run time errors olmalıdır zira yazım hataları zaten kullanılan editör programı tarafından tespit edilmektedir. Python'da bütün excepitons tipleri, yerleşik BaseExceptions sınıfının altsınıfıdır. BaseExceptions sınıfı, SystemExit, KeyboardInterrupt, GeneratorExit ve Exceptions olmak üzere 4 alt sınıfa ayrılır. Tüm bildiğimiz hata mesajları ise Exception sınıfı altında yer almaktadır.

BaseException

+-- SystemExit

+-- KeyboardInterrupt

+-- GeneratorExit

+-- Excepiton

Python dilinde hata yakalama işlemleri dört anahtar sözcük ile yönetilir; try, except, else ve finally. Hata yakalam işlemlerinin özünü try/except kod blokları oluşturur. Bu iki kelime birlikte aynı yapıyı oluşturur. try kelimesi tek başına kullanılmaz, except ile birlikte kullanılır. Aynı yapıda seçimlik olarak istenirse else ve finally ifadeleri de kullanılabilir. Bir işlemin her koşulda-hata olsun ya da olmasın-kesin oalrak yapılması istendiğinde try/except bloğuna finally ifadesi eklenir. Örneğin açık kalmış dosyaların kapatılaması 'dosya.close()' gibi işlemlerde genelde finally bloğu içerisine yazılarak olası problemler engellenmiş olur. Seçimli yapılardaki gibi else'nin if-else yapısındaki işlevi neyse try-except-else yapısındaki işlevide aynıdır. Genelde hataları gruplandırmada kullanılır.

try bloğunda atılan hatayı except bloğu yaklar ve onun hangi tip olduğunu belirlemeye çalışır. Önce, hatanın Exception-1 tipinde olup olmadığına bakar. O tipten ise,istenen önlemleri alacak kodları çalıştırır. Değilse, hatayı kendisinden sonra gelen except bloğuna yollar. Bu süreç hata tipi belirlene kadar arka arkaya devam eder. Eğer try bloğunda bir hata atılmadı (ortaya çıkmadı) ise else kısmındaki kod çalışır, istenirse else bloğu içerisine de başka try-except kod blokları tanımlanabilir. Hiçbir except bloğu hata ile karşılaşmazsa finally bloğundaki kod çalışır. try deyimi ile en az 1 tane except tanımlamak gerekir, except sayısı istendiği kadar olabilir.

**raise ile Bilerek Hata Oluşturmak :**

Normalde programın hata oluşturma ihtimali yok ama siz yine de kullanıcıya bazı özel durumalr için uyarı mesajı göndermek istiyorsanız raise deyimi kullanılır. raise yanına kafamıza göre hata kodu yazamayız. Ancak Python'da tanımlı hata mesajlarını yazabiliriz.

**User-Defiend Exceptions (Kullanıcı Tanımlı İstisnalar) :**

Tüm exception tiplerini yerleşik BaseException sınıfının birer alt sınıfıdır. Fakat Python programcıya; kendi exceptions sınıflarını oluşturma izni verir. Bunun için yapılamsı gereken; bir istina sınıfı tanımlamak ve bu istina sınıfının örneğini oluşturmak.