# Dersin Amacı ve Hedefi

Bu ders yapısal algoritmanın ve programlamanın temel elemanlarını öğrencilere tanıtacaktır. Programlama dili olarak C programlama dili kullanılacaktır. Bu ders süresince öğretilecek konular içinde tip kavramı, temel tipler, deyimler, değişkenler, sabitler, operatörler, temel giriş/çıkış fonksiyonları, mantıksal ve karşılaştırma operatörleri, döngüler, diziler, katar dizileri (strings), sıralama algoritmaları, arama algoritmaları, dosyalama ve standart fonksiyonlar bulunmaktadır.

Bu dersi başarıyla tamamlayabilen öğrenciler şunları yapabilecektir;

- Problem analiz etme ve algoritma hazırlayarak problemi çözme sürecini anlayacak,
- Programlama dilinin temel yapı taşlarını öğrenmiş olacak,
- Küçükten orta boya kadar program yazma, test etme ve hata ayıklama yapmış olacaktır.

# Giriş

İçinde yaşadığımız teknoloji çağında bilgisayarları yoğun bir şekilde kullanmaktayız. Bilgisayarlar vasıtasıyla kullandığımız programlar günlük hayatta işlerimizi büyük ölçüde kolaylaştırmaktadır. Alış-veriş merkezlerindeki kasiyerlerin kullandığı paket programlar, ticari ve sanayi alanlardaki makinelerin bilgisayar destekli kullanılan programları, eğitim alanlarında kullanılan otomasyonlar ve bilgisayarların kullanıldığı her alanda sayamadığımız birçok program bilgisayar programcıları tarafından programlama dilleri kullanılarak yazılırlar. Otomasyon yazılımları, bir matematiksel işlemin veya bilimsel bir hesaplamanın bilgisayarla çözülmesi hızlı, daha kolay ve doğru yapılmasını sağlar.

Programlama dilleri zaman içerisinde gelişmiş, yeni sürümleri ile değişmiş, kaybolmuş veya yenileri çıkmıştır. Bu nedenle programlama bilgisini asla bir programlama diline bağlı tutmamak gerekir. Eğer programlama mantığı oldukça iyi gelişirse, algoritmaları kolay kurup algılayarak, çok karmaşık sorunlar üzerinde fikir yürütülüp çözüm üretilebilir. Daha sonra da bilinen uygun bir programlama dilinin formatına uygun bir şekilde kodlanabilir. Programlama temeli ve mantığı kavrandıktan sonra, kısa bir sürede bir programlama dili orta düzeyde öğrenilebilir.

ALGORİTMA VE PROGRAMLAMA

## Programlama Nedir?

Bilgisayarların isteğe uygun özel işlemler yapması için programlanması gerekir. Örneğin, bir şirkette kullanılan stok uygulaması, sipariş uygulaması ya da değişik iş takipleri, hastane otomasyonları ya da eğitim kurumlarının kullandığı öğrenci otomasyonları gibi sıralayabiliriz. Programlamaya çok fazla örnek vermek mümkündür.

## Program

Günlük hayattaki bir sorunu bilgisayarla çözmek, rutin işlemleri kolaylaştırmak için bilgisayarların isteğe uygun olarak özel bir takım işlemleri gerçekleştirmesi için programlanması gerekmektedir. İşte yazılan bu yazılımlar ile ortaya çıkan ürün bir programdır. Bilgisayar oyunu, muhasebe işlem programları ve ticari şirketlerde kullanılan paket programlar birer programdır. Bir programın amaca uygun olarak çalışabilmesi için, belirli aşamaları yerine getirmek gerekmektedir.

1- ANALİZ: Problem analiz edilir ve çözümlenir. Problem analizi ve çözümlemesi, problemin tanımlanması, problemin giriş bilgileri ve sonuca ulaşmak için bu bilgiler üzerinde yapılması gereken işlemlerin ayrıntılı olarak belirlenmesidir.
2- TASARIM: Yapılan çözümlemeye göre algoritma tasarımı/akış diyagramı oluşturulur. Doğruluğunun mantıksal sınaması yapılır. Akış diyagramı, algoritmaların özel geometrik şekiller ile gösterilmesidir. Algoritmaların ifade edilmesinde akış diyagramlarının yanı sıra, konuşma dili ile programlama dili arasında, sözde kod (pseudo-code) adı verilen bir araç kullanılır. Sözde kod, programlar gibi derlenmez ve işlenmez. Her programcı kendi sözde kodunu geliştirebilir. Fakat kişisel sözde kodlar başkaları tarafından anlaşılabilir bir biçimde açık olmalıdır.
3- KODLAMA: Oluşturulan algoritma/akış diyagramı bir programlama dili ile yazılır (kodlanır).
4- DERLEME: Program derlenir ve çalıştırılır. Yazım hataları varsa düzeltilir. Derleme, bir programlama dili ile yazılan kodların, işletilebilmesi için makine diline çevrilmesidir.
5- TEST: Program istenilen sonuçları üretmiyor ise 1. adıma dönülür ve problem çözümlenmesi ve algoritma/akış diyagramı gözden geçirilir ve revize edilir.
6- DÖKÜMANTASYON: Programın dokümantasyonu oluşturulur.

## Programlama Dili

Bilgisayarda çözülecek bir sorun için çözümün bilgisayara adım adım yazılmasını sağlayan biçimsel kuralları olan ve bu kurallara sıkı sıkıya bağımlılığı gerektiren bir tanımlar kümesidir. Yani, programcı ile bilgisayar arasında bir tercüman durumundadır.

## Derleyici Nedir?

Derleyici (Compiler), bir bilgisayar dilinde yazılmış olan kodu, bilgisayarın (yada elektronik cihazın) donanımına uygun makine diline çeviren bilgisayar programıdır.

Derleyici öncelikle yazılan program kodunun doğru yazılıp yazılmadığını kontrol eder, eğer hatalar varsa bunları programcıya bildirir.

Eğer kod doğru ise derleme yapılan sisteme uygun olan 0 ve 1'lerden oluşan makine kodunu üretir (EXE dosyası).

## Programlamanın Tarihi

Bilgisayarlar, icat edilmeleriyle birlikte belli bir işi yapmak için bir dizi komutlara ihtiyaç duymuşlardır. En başta çok basit işlemler yapan bu komutlar zamanla nesneye yönelme (object orientation) gibi ileri seviyede özellikler kazanmıştır.

İlk programlama dilleri, bilgisayarların üzerinde bazı araçların yerlerini değiştirerek veya yeni bileşenler eklenerek yapılıyordu. Programın işlemesi için bir devinime ihtiyaç vardı. Eskiden programlar fiziksel olarak yazılıyordu. Daha sonra fiziksel programlama yerini elektrik sinyaline bıraktı. Artık, kurulan elektronik devrelere düşük ya da yüksel voltajda akım gönderilerek bilgisayarın davranışı belirlenmeye başlandı. Yüksel voltaj 1, düşük voltaj 0 sayılarını ifade ediyordu. Böylelikle bugün de kullanılan makine dilinin ortaya çıkması için ilk adımlar atılmış oldu.

Ancak bu şekilde programlar yazmak, sistemi oluşturan elektronik devrelerin her program için baştan kurulmasını gerektiriyordu. Böylelikle programlar bazı kavramlar çerçevesinde yazılmaya başlandı. Öncelikle bilgisayar donanımı her program için baştan kurulmamalı, bunun yerine basit

bir donanımın üzerine yazılan komutlar kullanılmalıdır. Daha sonra, programlar tek bir komutlar zinciri yerine, küçük parçalar halinde yazılmalıdır. Bu parçaların programın içinde defalarca kullanılabilmesi yordam (subroutine) kavramını ortaya çıkarmıştır. Bu modelin kullanılması ise mantıksal karşılaştırmaları, döngülerin kullanılmasını ve yazılan kodlar tekrar kullanıldığı için kütüphane (library) mantığını ortaya çıkarmıştır.

1957 yılında IBM, düşük seviye (makine diline yakın) bir programlama dili olan FORTRAN dilini ortaya çıkardı. FORTRAN ile beraber basit mantıksal karşılaştırmalar, döngüler, (true-false) lojik ve (integer, double) sayısal değişkenler kullanılmaya başlandı.

1959 yılında, bu programlama dilinin özelliklerini alıp, giriş çıkış (Input – Output IO) gibi yeni işlevler sağlayan COBOL dili ortaya çıktı. Daha sonra 1968 yılında, COBOL ve FORTRAN dillerinin en iyi özelliklerini alarak Pascal ortaya çıktı. Ayrıca Pascal dili, hafızadaki adresler üzerinde işlem yapmaya olanak veren işaretçi (pointer) kavramını beraberinde getirdi.

1972 yılında C, Pascal dilindeki birçok hatayı gidererek ortaya çıktı. C dili ilk defa Unix işletim sistemini yazmak için kullanılmaya başlanmıştır. C, düşük seviye bir dil olması, kuvvetli giriş çıkış işlemleri sağlaması gibi birçok özelliği ile işletim sistemleri yazılmasında tercih edilmiştir.

Bütün programlama dilleri birçok özelliğe sahip olmasına rağmen, modüler programlamanın birçok eksiğini gidermek amacıyla, yeni bir programlama modeli olan nesneye yönelik programlama - OOP (object oriented programming) ortaya çıkarıldı. C dilinin ve OOP modelinin tüm özellikleriyle C++ dili oluşturuldu.

C++ dilini, Sun Microsystems tarafından çıkartılan Java takip etti. Java dilinin kullanım alanları, nesneye yönelik bir programlama dili olması ve beraberinde getirdiği çöp toplama GC (garbage collection) gibi performans arttırıcı özellikleri ile büyük ölçüde genişledi.

Microsoft, 2000 yılında .NET platformunu sunarak, otuzdan fazla programlama dilini aynı çatı altına topladı. VisualBasic.NET ve Visual C# .NET platformunu kullanan günümüzdeki en güçlü yüksek seviyeli programlama dilleri arasında yer almışlardır.

Her programla dilinin kendine göre bir söz dizimi ve yapısı vardır. Bu söz dizimi ve yapı dilin köküne ve seviyesine göre değişiklik göstermektedir.

Genel olarak programlama dillerini 3 kategoriye ayırabiliriz, bunlar;

- Temel Programlama Dilleri (Fortran, C, Cobol, Basic, Pascal)
- Veriye Yönelik Programlama Dilleri (Lisp, AplSnabol, Icon)
- Nesneye Yönelik Programlama(Simula, C++, Ada95, Java, Visual Basic, C#)

Günümüzde en yaygın olarak kulalnılan yapı nesne tabanlı olan yapıdır. Bu yapı günlük hayatımızla tamamen uyuşarak benzerlik göstermektedir. Bu yapıya göre sistemler nesnelerden, nesneler daha küçük alt nesnelerden oluşur. Bu nesneler kendi özelliklerini kalıtım yolu ile diğer nesnelere aktarabilir, özelliklerini başka nesnelere aktarabilirler.

Programlama dilleri kullanım amaçlarına, geliştirme süreçlerine ve kullanım kolaylıklarına göre de kategorize edilebilirler bunlardan en başlıca olanları;

- Makina Dili
- Düşük Seviye
- Orta Seviye
- Yüksek Seviye
- Çok Yüksek Seviye

dillerdir.

## Makine Dili

Bu programlama dili ile yazılan kodlar sadece 1 ve 0 lardan oluşurlar ve doğrudan işlemci ile iletişim halindedirler bu 1 ve 0 lar işlemcinin mantıksal ve artimetik ünitesinde gerekli değişikliklere sebep olup işlemciyi istediğimiz gibi yönetmemizi sağlarlar. Program hangi dille yazılırsa yazılsın en son olarak yazıldığı dil yardımcı araçlar ile assembly'e çevrilip daha sonra makine dili karşılığı oluşturulup, makine karşılığı işletilirler.

## Düşük Seviyeli Diller

Makine koduna oldukça yakın programlama dilleridir. Makina hâkimiyeti oldukça gelişmiştir. Bu dillerde program geliştirmek uzun ve zahmetli bir iştir. Bu diller ile oluşturulan programlar oldukça performanslı çalışmaktadırlar. Bu dillere örnek verecek olursak Assembly gibi dilleri verebiliriz.

## Orta Seviye Diller

Bu diller düşük seviye dillere göre geliştirilmesi ve anlaşılması bakımından daha kolay olup, yüksek seviyeli dillere göre de daha performanslı çalışan programlar yazılmasını sağlayan dillerdir. Bu dillere örnek verecek olursak C, C++ gibi dilleri verebiliriz.

## Yüksek Seviyeli Diller

Bu diller genellikle son kullanıcı için program geliştirmekte kullanılırlar. Söz dizimi ve yapısı orta ve düşük seviyeli dillere göre daha anlamlı, anlaşılır ve kolaydır. Programlamaya yeni başlayacakların bu seviye dillerle başlamaları tavsiye edilir. Bu diller her ne kadar kolaylık sağlasalar da program üstündeki hâkimiyetimizi kısıtlayabilirler. Eğer çok performans gerektiren veya doğrudan işlemci ile muhatap olacağınız bir yazılım geliştirmiyorsanız bu seviyedeki diller tercih edilmelidir. Bu seviyedeki dillere örnek verecek olursak C#, java gibi dilleri örnek verebiliriz.

## Çok Yüksek Seviyeli Diller

Bu diller genellikle olay tabanlı diller olup kullanım bakımından en kolay dillerdirler. Bu dillere de örnek verecek olursak Visual Basic, Oracle Forms gibi dilleri verebiliriz.

## Kullanım Amacına Göre Programlama Dilleri

**Bilim ve Mühendislikte:** Pascal, C, C++, Java, Fortran…

**Veritabanı Programcılığında:** Dbase, Acces, Foxpro, Sql…

**Yapay Zeka Kullanımında:** Prolog, Lisp…

**Sistem Programcılığında:** C, C++, Java ve sembolik makina dilleri…

**Web Programcılığında:** Html, Php, Asp…

## Programlama Dillerinin Bazı Özellikleri

**İfade Gücü:** Dili kullanırken gerçek ifadelerin kullanılması ile ilgilidir. Örneğin bir matematikçi ve kimyacı kodlama yaparken kullandığı işaretleri ve terimleri kullanmak isteyecektir.

**Veri Türleri ve Yapıları:** Ön tanımlı değişken türlerinin fazla ve ihtiyaçları karşılaması, bir dilden beklenen bir özelliktir.

**Giriş - Çıkış Kolaylığı:** Dosyalara erişme, karmaşık işlemler yapma imkanlarını kasteden bu özellik, C'de pek gelişmemiştir. Özel kütüphaneler gerektirir. Veri tabanı programlama dilleri bu konuda oldukça gelişmiştir.

**Taşınabilirlik:** Bir sistemde yazılmış kaynak kodun, başka sistemlerde de sorunsuz derlenebilmesidir. Genellikle dilin seviyesi azaldıkça taşınabilirlik azalır. C dili, orta seviyelidir ancak taşınabilirlik bakımından üstündür.

**Alt Programlanabilirlik:** Programın daha ufak programlardan oluşturulmasıdır. Böylece kaynak kod kısalır, algılanması güçlenir, test olanakları artar, kodun güncelleştirmesi ve yeniden kullanılması kolaylaşır.

**Verimlilik:** Derlenen kodun hızlı ve sorunsuz çalışabilmesidir.

**Okunabilirlik:** Kaynak kodun hızlı biçimde anlaşılabilmesidir. İyi bir programcının yazdığı kaynak kod, çok iyi işlev gören ama karışık bir koddan ziyade açık ve anlaşılabilir biçimdedir. Ancak bu ölçüt dile de bağlıdır.

**Esneklik:** Dilin, programcıyı kısıtlamamasıdır. Ancak esnek bir dil, daha az hata vermesine karşın hata oluşma riski daha fazladır.

**Öğrenme Kolaylığı:** Dilin konuşma diline yakınlığı, komutlarının sade ve anlaşılır olması gibi ölçütler o dilin öğrenilmesini etkiler.

**Genellik:** Bir dilin herhangi bir alanda kullanılabilmesidir. Bazı diller sadece mühendislik alanlarında kullanılmasına karşın, C genel amaçlı bir dildir.

**Yapısal Programlanabilirlik:** Programın bloklar halinde yazılması, düz akışı ve altprogramların kullanılması anlamlarına gelen bir programlama tekniğidir. Kodun okunabilirliğini ve verimini artırır.

**Nesne Yönelimlilik:** Yeni diller ve eski dillerin yeni uyarlamaları artık nesne yönelimli olmaya başladılar. Verilerin birbirinden daha kesin çizgilerle ayrılmasını öngören bir programlama tekniğidir.

## En Çok Kullanılan Programlama Dilleri


**C:** Yordamsal programlama dilleri arasındadır. Öğrenilmesi zaman almasına rağmen oldukça kullanışlı ve esnek yapısı ile adından yıllarca bahsettirmiş, bilgisayar programcılığının temel dillerinden biridir. Kullanım bakımından çok geniş bir olanağa sahip olan bu dil en çok kullanılan yordamsal dildir. Genel olarak işletim sistemleri de bu dille yazılırlar.

**C++:** 1980'lerin başlarında Bjarne Stroustrup tarafından geliştirilen bu dil nesneye dayalı programlama dilleri arasındadır. C dilinin hemen hemen bütün özelliklerini kapsayan bu dil C diline ek olarak güçlendirilmiş nesne yönetim özelliği ile şu anda bilgisayar dünyasının en çok kullanılan ve en esnek dillerinden biridir. Bu dil her ne kadar nesneye dayalı bir dil olsa da yordamsal bir dilin özelliklerini de taşımaktadır, aslında bu dil hem Nesnel hem de Yordamsal bir dildir diyebiliriz. Bu dil birçok işletim sistemi tarafından desteklenmektedir.

Bu dil Microsoft tarafından java'dan esinlenilerek geliştirilmiş tamamen nesneye dayalı(OOP) bir dildir. Programcıya internet uygulamaları ve yerel uygulamalar yazmakta bazı kolaylıklar getirmiştir. Bu dilin öğrenilmesi ve kullanımı oldukça kolaydır. Bu dille geliştirilen programlar .NET framework üstünde çalışmaktadırlar. Yani bu diller sadece Windows işletim sistemi ve .NET framework ile çalışmaktadır.

İlk temelleri 1993'de atılan 1995'de de piyasaya sürülen java programlama dili nesneye dayalı dillerdendir. Son yıllarda internet ve yerel uygulamalar geliştirmek için yoğun bir şekilde kullanılan ve sürekli gelişen bir dildir. Oracle tarafından geliştirilen bu dilde bir framework üstünde çalışmaktadır. Fakat bu dil C#'ın aksine birçok platformda çalışabilmektedir.

Bu dil Yordamsal bir dildir, bu özelliiği ile C diline benzerlik gösterir. Öğrenilmesinin zor olmayışı ve bilgisyar eğitimi veren okullarda okutulan bir ders olması sebebiyle kullanım alanı genelde üniversiteler ve bilimsel hesaplamalar yapan kurumlardır.

Bu dil Pascal tabanlı bir dil olup nesnesel bir yapısı vardır. Öğreniminin çok zor olmayıp geçmişte ülkemizde üniversitelerde nesnesel diller arasında en yaygın olarak eğitimi verilen dil olması sebebi ile geçmişten günümüze gelen ve ülkemizde geliştirilmiş programların birçoğu bu dille yazılmıştırlar. Visual programlama özelliği taşır.

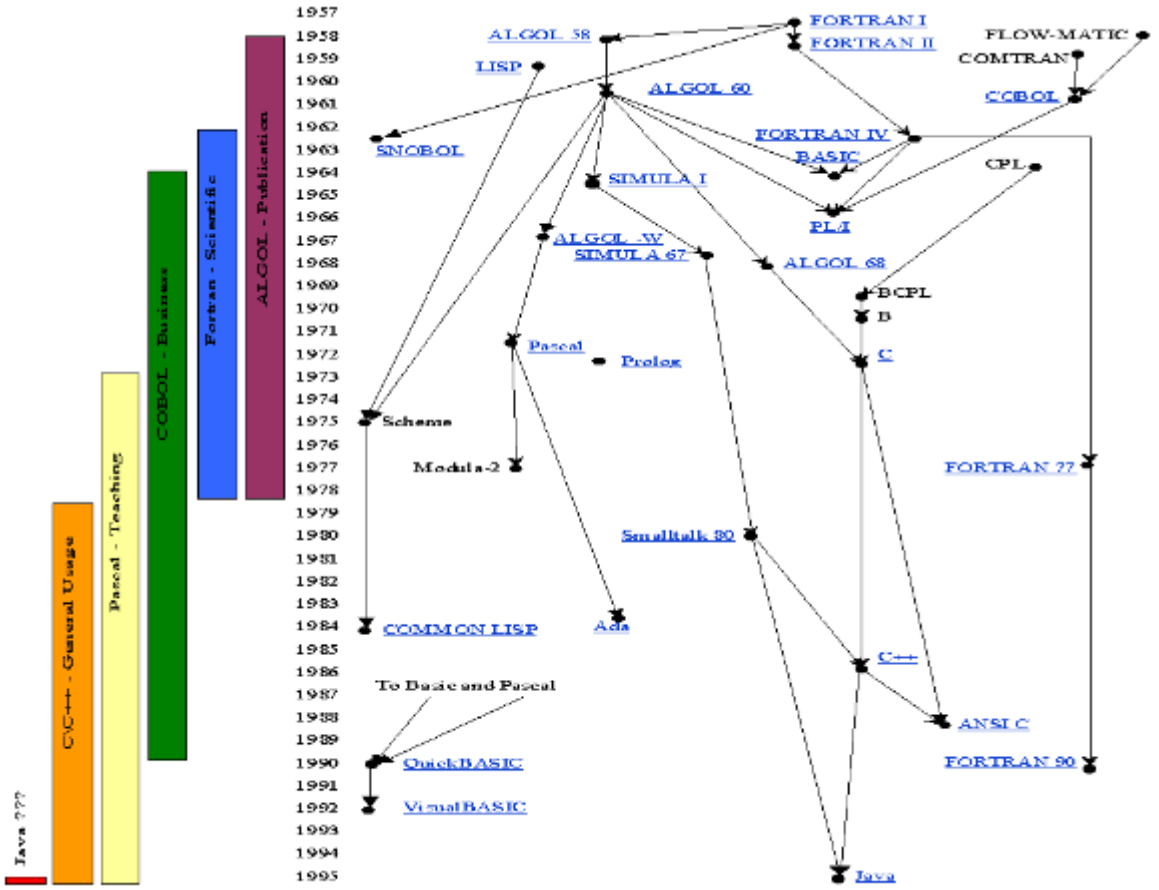

Bu dil Nesnesel diller arasındadır. Basic tabanlı bir dil olup öğrenilmesi kolay, kullanım alanı geniş bir dildir. Özellikle görsel uygulamalarda projenin arabiriminin hızlı yazılmasını sağladığı için genelde kullanıcı arabirimi tasarımlarında kullanılır.

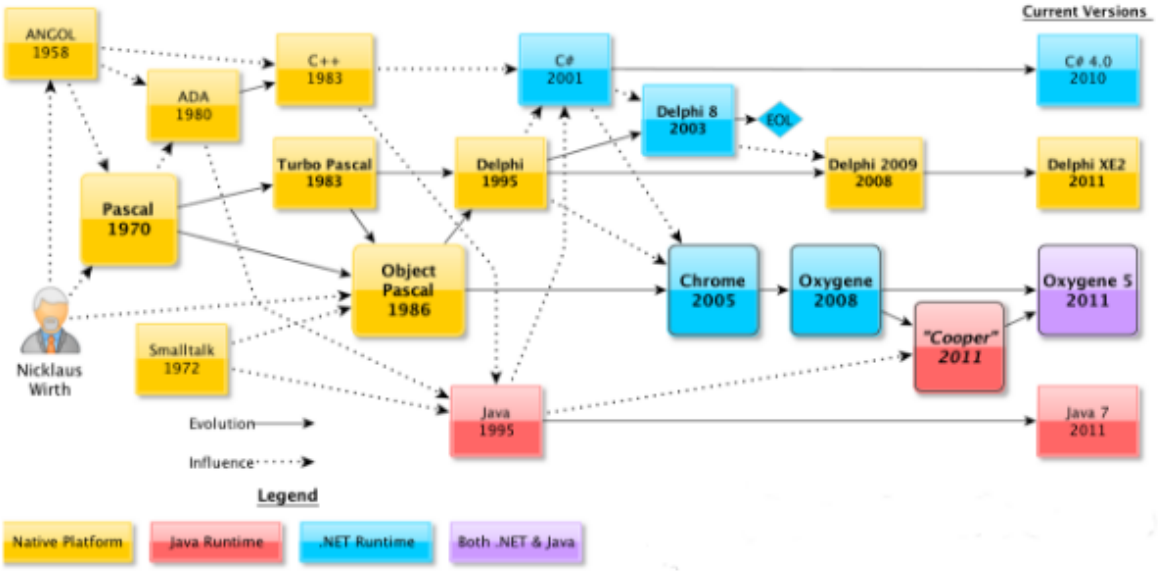

1990 yılında Guido van Rossum tarafından Amsterdam'da geliştirilmeye başlanmıştır. Son derece kolay okunabilir bir söz dizimine sahipt Nesnesel bir dildir. Birçok platformda problemsizce çalışabilecek şekilde tasarlanmış olup kullanım bakımından yükselen bir grafiğe sahiptir.


Adı Türkçesi Zengin Metin İşaret Dili olan Hyper Text Markup Language sözcüklerinin ilk harflerinden oluşur. Bu dil ile kendi kendine çalışan programlar yazılamaz. HTML'in özelliği birden fazla bilgi içeriği türü olan sayfaların bir biri ile bağlanmasında ve şekillendirilmesinde kullanılır.


1995 yılında Rasmus Lerdorf tarafından ilk ortaya atılan PHP'nin geliştirilmesi bugün PHP topluluğu tarafından sürdürülmektedir. Bu web tabanlı uygulamalarda yoğun bir şekilde kulalnılır. C/C++ diline olan benzerliği nedeniyle bu dili önceden bilenlerin PHP öğrenmesi oldukça kolaydır. Kolay öğrenilmesi ve hızlı performansı nedeniyle Facebook, YouTube, Yahoo ve Wikipedia gibi dünyaca ünlü sitelerin kaynak kodudur.

## Programlama Dillerinin İlişkileri



## Algoritma Nedir?

Algoritma, bir problem sınıfının sistemli çözümü için; sıralanmış, belirsizlik taşımayan, işlenebilir, sonlu işlem adımları kümesidir.

Algoritma, sayıları kolay ve doğru tanımlama sanatıdır. Bir işlem dizisinin sonucunu elde etmek için, çok iyi tanımlanmış, sonlu sayıdaki işlem cümlelerinin bütünüdür.

Programlama mantığına göre algoritma; bir işlemi sonuçlandırmak, bir problemi mantıksal bir sıraya göre çözmek için, sembollerle veya kelimelerle anlatım şeklidir.

Algoritma yaklaşımı, 9. yüzyılda yaşamış Türk-İslam matematikçi ve astronomu Harezmi'nin ikinci dereceden denklemlerin kolayca çözümü için geliştirdiği çözüm yönteminin genelleştirilmiş şeklidir; algoritma sözcüğü de onun adından türemiştir.

Algoritma kısaca çözüm adımlarıdır. Farkında olmasak da hayatın her anında algoritma kullanırız. Evden okula ulaşmak için ulaşım algoritmasını, açlığımızı gidermek için yemek algoritmasını tercih ederiz. Amaç okula ulaşmak ya da doymak ise algoritmadaki bir hata oldukça can sıkıcı olabilir. Okula ulaşma algoritmasını ele alalım. Her algoritmada "Başla" ve "Bitir" ifadelerine yer verilir.

1. Başla
2. Otobüs durağına git.
3. Üniversiteye giden otobüsü bekle.
4. Otobüse bin.
5. Biletini okut.
6. Üniversite durağında in.
7. Okula gir.
8. Bitir

Basit gibi görünen bu algoritma üzerinde bir hata yapalım. Örneğin 3 numaralı adımı atlayalım. Eğer yeterince şanslı isek doğru sonuca ulaşabiliriz (Yani üniversiteye giden otobüse

binebiliriz). Ama büyük ihtimalle ilk gelen otobüse bindiğimiz için şehrin üniversiteden uzak bir semtine gideceğimizden sonuncu adımı asla gerçekleştiremeyeceğiz. Hergün yaptığımız ama yaparken düşünmediğimiz birkaç eylem için algoritmayı yukarıdaki gibi kurmak algoritma kurma prensiplerini anlamayı kolaylaştıracaktır.

Başka örneklerle durumu açıklayacak olursak; yemek pişirmek bile, belli bir sıra gerektirmektedir. Öncelikle bu yemeğin malzemesi alınır. Daha sonra yemek pişirilir ve sonunda servis yapılır. Başka bir örnek verecek olursak; size 10 sayısı ile 20 sayısının toplam sonucu sorulabilir. Burada ilk yapılacak işlem iki sayıyı alırız, bu sayıları beynimizle toplama işleminden geçirir ve daha sonra sonucu söyleriz. Dikkat edilecek olursa, bu iki örnek arasında konu bakımından farklılıklar olsa bile, işlem sırası bakımından hiçbir farkın olmadığını görürüz. Bu sıraları değiştirmek mümkün değildir. Bu sıranın ne olduğuna bakacak olursak 3 ana bölümden oluştuğunu görürüz. Bunlar;

1. Giriş Bölümü

2. İşlem Bölümü

3. Sonuç Bölümü

Yukarıdaki iki farklı örneği karşılaştıracak olursak; yemek malzemesi alınması ve iki sayının toplama işlemi için bize verilmesi giriş bölümünü ifade etmektedir. Yemeğin pişirilmesi ve diğer taraftan sayıların beynimizle toplama işleminin gerçekleştirilmesi işlem bölümünü, yemeğin servise sunulması, ve sayıların toplam sonucunun söylenmesi bize sonuç bölümünü gösterir. Bu sıra asla değişmez.

Algoritma kurulurken dikkat edilecek hususlar şunlardır;

1-Mutlaka bir başlangıcı ve sonu olmalıdır. Sonu olmayan bir algoritmanın test edilmesinin mümkün olmadığı açıktır.

2-Bir algoritmadaki ifadeler kesin olmalıdır. Belirsizlikler algoritma ile ulaşılmak istenen sonucun elde edilememesine neden olur.

3-Algoritma sınırlı sayıda belirli kurallarla kurulmalıdır. Ve bu belirli kurallar takip edecekleri bir sıraya sahip olmalıdırlar.