Sıfırdan başlayarak algoritma ve programlama öğrenme

Emre YAZICI

Yapay Zeka Mühendisi
Programlama Uzmanı - MCP

2008

Lütfen, kitabın herhangi bir sayfasını açıp okumaya başlamayınız. Bunun yerine, açıklamalar bölümünü okuyunuz.

Önsöz

Programlama artık dünyanın vazgeçilmezi haline geldi. Nereye gidersek gidelim, hangi işi yapacak olursak olalım bilgisayarsız ve bilgisayarları yöneten programlarsız bir sistemle karşılaşamıyoruz. Tabi, ülkemiz için bu durum söz konusu değildir. Ülkemiz daha tam olarak, bilişim sektörüne gir**e**memiştir. Ülkemiz, bilişim sektörünü, bilişim sektöründeki değişiklikleri, yaklaşık olarak 5-6 yıl geriden izlemektetir. Bilime önem ver**e**mediğimiz için ***bilimi kullanan değil bilimin kullandığı bir toplum*** haline geliyoruz ve bilime sonradan sahip olmak için çok büyük bedeller ödüyoruz. Sonuç olarak bilim, bize, olduğundan çok pahalıya satılıyor, üstelik bu ürünleri geliştirebilecek kapasiteye sahip olmamıza rağmen...

Yurtdışında geliştirilen sistemlerin (sadece) bazıları Türkler tarafından geliştiriliyor olsa da, hemen hemen hepsi Türkler tarafından geliştirilebilir. İşte beyin göçünü önleyip, teknolojileri Türkiye'de geliştirebilmek için, mümkün olabilecek en yüksek bilgi ve donanımlara sahip olmalıyız.

Türkiye'de hemen hemen hiçbir (yabancı) büyük şirketin, büyük bir Araştırma - Geliştirme ekibi yoktur. Yabancı büyük şirketlerin bazıları, Türkiye'yi sadece ürünleri satabilecekleri bir pazar olarak görmektedirler. Aynı Osmanlı İmparatorluğu'na uygulanan kapitilasyonlarda olduğu gibi...

Dünyada ki en zeki milletler arasında; Türkler, Hindular, Ruslar bulunur. Ancak malesef en çalışkanlar arasında çok gerideyiz. İşte burada, düşmanımızı, "**çalışmamazlık - tembellik**" olarak belirlemeli ve ona karşı savaşmalıyız...

Hedef Kitle

Bu kitabın hedef kitlesinde bir limitleme - kısıtlama bulunmamaktadır. Ancak, kitabı okuyacak kişilerin, bilgisayar kullanmayı (bilgisayarı açıp kapamayı, ofis programlamarını kulllanmayı, bazı ayarları ve dosya işlemlerini yapabilmeyi) bildiklerini varsayıyorum. Eğer, üniversitede bu bölümü (programlama - yazılım mühendisliği) okuyacaksanız, tavsiyem bu kitabı okuyarak, üniversiteye gitmenizdir. Eğer ikinci bir meslek sahibi olmak istiyorsanız veya daha sonra programlamayı okumayı planlıyorsanız, bu kitap sizin için en uygun olanıdır.

Ayrıca, bu kitabı okuyacakların, matematik konusunda bilgili olması – sorun yaşamıyor olması gerekmektedir. Aslında, matematikte bilgili olunması bu kitabın değil programlamanın şartıdır. Matematikte bilinmesi gereken konular; Sayılar, Kümeler , Dört Işlem, Üssü ifadeler , Mantık, Fonksiyonlar , Sayı tabanları, Toplam sembolü, Çarpım sembolü, Matris, Diziler - Seriler .... Eğer bu konularda çok ciddi probleminiz var ise, size bu konular üzerinde göz gezdirmenizi tavsiye ederim. Önemli olan o konu üzerinde soru çözmeniz değil, konuyu anlamış olmanızdır.

Yazılım Mühendisi: Normalde,

- bir projeyi tasarlayan,
- algoritmik (mantık – uğraş gerektiren) kodları yazan,
- veritabanını tasarlayan,
- sistemin akış şemasını hazırlayan,
- test işlemlerini gerçekleştiren
- benzeri planlama işlemlerini gerçekleştiren

kişiye verilen ünvandır.

Kodcu:

- Tasarlanmış olan programın kodunu yazan
- Hazırlanmış şemaya ve tasarım yapan
- Veri girişi – kontrölü yapan

kişiye verilen isimdir.

Programlamacı: Yazılım mühendisi ile kodcu arasında olan ve her ikisininde yapmış olduğu işlemleri gerçekleştiren kişiye verilen isimdir. Bu kitap, yukarıda belirtilen üç kategorideki kişi için tasarlanmıştır.

Yazar Hakkında

Uzun süredir programlama yapmaktayım. Benim, lisede başlayan programlama aşkım şimdiye kadar hiç sönmedi. Liseden sonra, İngiltere'de Yapay Zeka üzerine lisans öğrenimi aldım. Bu sırada beynin çalışma yapısı ile ilgili öğrendiklerimi geliştirdim. Daha sonra, bu kitabı bir okuyucunun nasıl daha kolay öğreneceğini hesaplayarak, planlayarak tasarladım.

Birçok farklı dilde, yüzlerce, proje geliştirdim. Hep farklı sistemler üzerinde çalıştığım için, çok iyi tecrübeye sahip oldum. Tecrübe, aynı konu, aynı program, aynı iş üstünde yıllarca çalışmak değildir. Bu yıllarca çalışma esnasında, aynı kategoride, farklı konuları işlemek, farklı programlar yazmak ve farklı işler üzerinde çalışmaktır.

Uzun süre, bazı programlama ile alakalı forumlarda moderatörlük, bazılarında yazarlık yaptım. Ancak bu süre zarfında gördüm ki, insanların birçoğu programlamayı bilmiyorlar. MSN'imi her gün biri ekliyor, soru soruyordu. Sorunun cevabını versem bile, "ben bunu yapamam, siz yapıp gönderebilir misiniz" diyenler oluyordu. Programlamayı mantıksal olarak değil yüzeysel olarak öğrendiklerinden dolayı, bir sistem kurabilmek onlar için zor oluyordu. Benim amacım, Çin atasözündeki gibi, balık tutmayı öğretmektir.

Türkiye'ye, hem bilgisayarın, hemde alanım olan yapay zekanın faydalarını, önemlerini anlatmak için çok uğraştım. Hâla da uğraşıyorum. Bunun için yaptıklarımın, yazdığım makalelerin haricinde sonunda bir kitap yazmaya karar verdim.

Ünlü yazar Balzac'ın dediği gibi, "bilginin efendisi olmak için, çalışmanın kölesi olmak gerekir". Ama ben sizi, köle olmaktan, bu kitabı yazarak kurtarıyorum. Sayfa sayısının az olması bu kitabın içinde çok fazla bilgi olmadığını düşündürür. Ancak kitapta, çok fazla örnek bulunmadığı için sayfa sayısı azdır. Birçok kitapta yüzlerce sayfa bulunur ama çoğu örnektir. Size bir konuyu tam anlatamadıkları için bol bol örnek sunarlarki aradaki fark kapansın, ayrıca, kitap kalın olunca, okuyucular da, "bu iyi bir yazar" ifadesi oluşsun.

Emre YAZICI

Kitap Hakkında

Size garanti edebilirim ki, eğer bu işi yapabilecekseniz, en iyisi, bu kitabı okuyarak başlamaktır. Daha önceden öğrenenlerin de, öğrendiklerini, unutmasını ve baştan sağlam bir şekilde tekrar anlamalarını öneririm, tabi eğer yeterli olmadıklarını düşünüyor iseler.

Teknolojinin ayaklarımızın altında, bize hizmet için beklediği bir çağda, onu en iyi şekilde kullanabilmek, en büyük gereksinimlerden biridir. Unutmayalım ki, herşeyi - her bilgiyi bilemeyiz ama her gördüğümüzü (konumuzla alakalı) anlayabiliyorsak, o zaman her konuya adapte olabiliriz ve her işi yapabiliriz demektir.

Kitabın, teorik ağırlıklı olduğunu söyleyebilirim ve "hadi hemen kod yazmaya başlayalım" şeklinde düşüncelere sahip olan okurların olacağını tahmin edebiliyorum. Ancak, eğer pratik tabanlı öğrenmeye başlarsanız, sistem kuramazsınız. Bu nedenle, teori + pratik şeklinde yapılacak olan ilerleme ile birlikte, en iyi sonuca ulaşırsınız.

İçindekiler

Bölüm 1: Algoritma

Birinci bölüm, programlama ve birçok mühendislik sisteminin tabanı olan, algoritmanın ne olduğunu, yapısını, nasıl ve ne amaçla kullanıldığını, nasıl algoritma kurulabildiğini, programlamanın ne olduğunu nasıl yapıabileceğini anlatmaktadır.

Anahtar sözcükler: algoritma, programlama, program, programlayabilme, programlanabilme, programlama şeması, analiz, lowest limitation, operatör, analiz, planlama, WBS, AND, flow chart, pseudo kod

Bölüm 2: Programlama ilkeleri

İkinci bölüm, temel programlama ilkelerini, (öyle ki bu temel programlama ilkeleri; değişkenler, matematiksel ifadeler, koşul ifadeleri, parantezler, döngü, toplam sembolü, çarpım sembolü, fonksiyonlar, önermeleri içermektedir) anlatmaktadır.

Anahtar sözcükler: matematik, değişkenü parantez, toplam sembolü, çarpım sembolü, döngü, mantık, önermeler, koşul ifadeleri, fonksyionlar

Bölüm 3: Beyin ve programlama

Üçüncü bölüm, beynin çalışma yapısını, bu çalışma yapısına göre programlamanın nasıl mümkün olabileceğini, sınıf ve nesne kavramını, inheritance – sub – super class ifadelerini anlatmaktadır.

Anahtar sözcükler: beyin, class, object, sınıf, nesne, inheritance, super class, sub class

Bölüm 4: Bilgisayar sistemleri

Dördüncü bölüm, bilgisayar sistemlerini; bit, byte, 1-0, boolean mantığını, işlemcinin çalışma yapısını, programlama dillerinin kategorilerini – özelliklerini, programlama dili ağaçlarını, programlama dili yapısını anlatmaktadır.

Anahtar sözcükler: işlemci, byte, bit, boolean, dil, hafıza, ram, rom, hex

Bölüm 5: Programlama için gerekli diğer bilgiler

Beşinci bölüm, programlama için gerekli diğer bilgiler olan; veri türlerini, programsal blokları, syntaxı, operatörleri, prosedürleri, argümanları, fonksyionları, işaretleri, değişken tanımlamayı anlatmaktadır.

Anahtar sözcükler: syntax, veri türleri, blok, operatör, prosedür, argüman, fonksiyon, işaret, değişken, enum, structure, interface

Bölüm 6: Veritabanı

Altıncı bölüm, veritabanı yapısını, veritabanı yönetim sistemlerini, tablo – alan ilişkisini anlatmaktadır.

Anahtar sözcükler: sql, tablo, alan, kolon, e-r, entity, relation, dbms

Bölüm 7: Test ve Hata ayıklama

Yedinci bölüm, programların test edilmesini, test yöntemlerini, test işleminde dikkat edilmesi gereken hususları, hata ayıklamayı anlatmaktadır.

Anahtar sözcükler: blackbox, whitebox, debug

Bölüm 8: Mimari

Sekizinci bölüm, birçok programlama dili tarafından ortak kullanılan mimarileri ve yapıları anlatmaktadır.

Anahtar sözcükler: threading, çoklu işlem, sockets

1 Algoritma

İçerik

1 Algoritma
1.1 Programlam Öğretimi
1.2 Algoritma
1.2.1 Algoritmaya giriş
1.2.2 Kimler algoritma kurabilir?
1.3 Programlama
1.3.1 Programlama – Program
1.3.3 Programlar ne işe yarar?
1.3.4 Programlanabilme & Programlayabilme
1.3.5 Programlama için gerekenler
1.3.6 Programlama nasıl mümkündür?
1.4 Programlama şeması
1.4.1 İşlem şeması
1.4.2 Analiz
1.4.3 Operatör
1.4.4 Lowest Limitation
1.4.5 Anlama
1.4.6 Kod yazımı
1.5 Planlama
1.5.1 Geniş ve kapsamlı sistemler
1.5.2 WBS
1.5.3 AND
1.5.4 Flow chart
1.5.5 Pseudo Kod
1.6 Programlama kısa tarihi ve yapısı
1.7 Efektif Programlama
1.8 Örnek Programlama

Amaçlar

- Birinci bölüm, programlama ve birçok mühendislik sisteminin tabanı olan, algoritmanın ne olduğunu, yapısını, nasıl ve ne amaçla kullanıldığını, nasıl algoritma kurulabilindiğini,
- programlamanın ne olduğunu, nasıl yapıabileceğini
- programlamada kullanılan tasarım elemanlarını anlatmaktadır.

Anahtar sözcükler

algoritma, programlama, program, programlayabilme, programlanabilme, programlama şeması, analiz, lowest

limitation, operatör, analiz, planlama, WBS, AND, flow chart, pseudo kod

1.1 Programlama öğretimi

Ön bilgi

Lütfen, herhangi bir bölümü, sizi ilgilendirmediğini veya yeterince önemli olmadığını düşünerek, okumadan geçmeyiniz. Bütün bölümler eksiksiz ve sırayla okunulmalıdır. Bununla birlikte, bir konuyu tamamen anlasanızda, o bölümü yinede sonuna kadar okuyunuz ve belirtilen örneklerin hepsini aklınızda canlandırınız.

Kitabın amacı, daha sonra öğreneceğiniz bir programlama dilinde, istediğiniz her türlü işlemi gerçekleştirecek alt yapıyı öğretmektir.

Öğretmen

Bir konuyu biliyor olmak, bir konu üzerinde çalışıyor olmak veya bir konu üzerinde tecrübe sahibi olmak, o konunun iyi anlatılabileceği – öğretilebileceği anlamına gelmemektedir. Çünkü öğretebilmek, farklı bir yetenektir, farklı bir bilgidir. Uzun süreden beri yazılım – programlama üzerinde çalışmakta olan bir yazılım mühendisi olabilirsiniz, ancak bu, bildiklerinizi iyi bir şekilde öğretebileceğiniz anlamına gelmemektedir. Yani mühendis olmanız, öğretmen olduğunuz anlamına gelmez. Her mühendis öğretmendir diye, dersanelerin çoğunda, 2-3 senelik tecrübeye sahip yazılım uzmanları, öğretmen olarak çalışabilmektedir.

Halbuki, yazılım mühendisleri, bir konunun nasıl öğretileceğini, anlatım tekniklerini bilmek zorunda değildirler. Zaten, yazılım mühendisliği müfredatı içinde öğretim, öğrenci psikolojisi, anlatma teknikleri, beynin çalışma yapısı ve benzeri konuları anlatan dersler bulunmamaktadır.

Bir konuyu çok iyi anlatabilmek için, beynin çalışma yapısını bilmeniz gerekmektedir. İşte bunun için ya kendinizi bu konuda bilgilendirmiş olmalısınız yada beynin çalışma yapısıyla ilgilenen bir bilim dalı olan yapay zeka uzmanı olmalısınız.

Programlama öğretimi – Dil ilişkisi

Bu kitabı diğerlerinden ayıran en önemli özellik; her bir programlama dili için bir veya iki kitap alarak o dilleri öğrenmektense, sadece bu kitabı okuyarak, birçok farklı dili çok kısa sürede öğrenebiliyor hâle gelecek olmanızdır. Piyasada, “C ile programlama”, “Delphi ile programlama”, “Java ile programlama”, “Basic ile programlama”.... gibi, birçok dilde, programlama yapmak üzere hazırlanmış kitaplar bulunur. Programlama dillerinin isimleri farklı olsada, hepsi aynı işlevi gerçekleştirir: **Programlama**. Eğer hepsi programlama kitabı ise, içeriklerinde ortak olan büyük bir bölümün olması gerekmiyor mu? Hatta bu ortak bölümün, kitapların en az %70’ini oluşturması gerekmiyor mu? Çünkü hepsi programlama kitabı ve kullandıkları platform aynı. Peki, varsayalım ki, on tane programlama kitabı aldık (Java ile programlama, C ile programlama ....). Hesabımıza göre bu kitapların %70’i aynı olacaktır. %70 x 10 kitap, %700 = 7 kitap. Yani alacağınız on kitabın yedi tanesi aynıdır. On kitabın yedi tanesi programlamayı anlatmaktadır. Ancak hepsi programlamayı farklı yöntemlerle anlattıkları için, birinin %70’lik bölümünü okuduğunuzda, diğerlerinin %70’lik bölümünü okumuş olmazsınız.

Sonuç olarak, bu şekilde, para ve vakit kaybetmektense, bu kitabı okuyarak programlamayı öğrenip, daha sonra diğer birkaç dili, başka bir kitap alarak, o dili, programlamayı bilerek öğrenebilirsiniz.

Çok karşılaştığım bir soru olan: "Hangi dili seçerek programlamaya başlamalıyım?" sorusunun cevabına gelince: Bu sorunun herhangi bir cevabı yoktur. Birçok dersanede, size programlamayı öğretirken bir dili kullanarak öğretirler. Örneğin: C ile programlama, Java ile programlama.... Programlama ayrı bir ifadedir, ve programlama bir dili kullanarak mümkün olmaz! Size bu durumu anlattıktan sonra birde örnekle pekiştireyim. Bir sürücü kursuna

yazıldığınızı varsayalım. Alacağınız dersler içinde, "Nissan ile araç kullanma", "Honda ile araç kullanma", "Toyota ile araç kullanma" ... olmayacaktır. Önemli olan otomobil kullanabilme yeteneğidir. Bu yeteneği öğrendikten sonra, her otomobili kullanabilirsiniz. Otomobiller arasında sadece birkaç fark bulunur (kullanım açısından: pedalların sertliği, kalkış hızı, boyutları, vites...).

Başka bir örnek olarak; (varsayalım ki) bir iş yerine, şoför olarak iş başvurusu yapıyorsunuz. İnsan kaynakları uzmanı, CV'nize bakıyor, sonra size şöyle bir soru soruyor: "Sizin ehliyetinizin sınıfı B, güzel.. Ancak bizim firmamızın araçları Honda markadır. Siz bu araçları kullanabilir misiniz?". Bu soruya hepimiz gülerek cevap verir. İşte aynı komik durum, programlamanın bir dilde ifade edilmesi içinde geçerlidir. Nasıl, Mercedes ile araç sürme şeklinde bir ifade olamayacağı gibi, bir dille (örneğin C, Delphi..) programlama ifadesi de olmayacaktır. Programlama tektir, programlamayı öğrendikten sonra hemen hemen her dili öğrenmeniz ve o dillerle programlama yapabilmeniz mümkün olacaktır.

Programlamayı öğrendikten sonra, sadece o dili konuşmayı öğrenmek gerekir ki, bu işlem çok kısa zaman alır. O yüzden, dile göre iş yapmayın, işinize göre dili seçin. **Alacağınız ayakkabıya, ayağınızı sığdırmaya çalışmazsınız, ayağınızın sığabileceği ayakkabı alırsınız**.

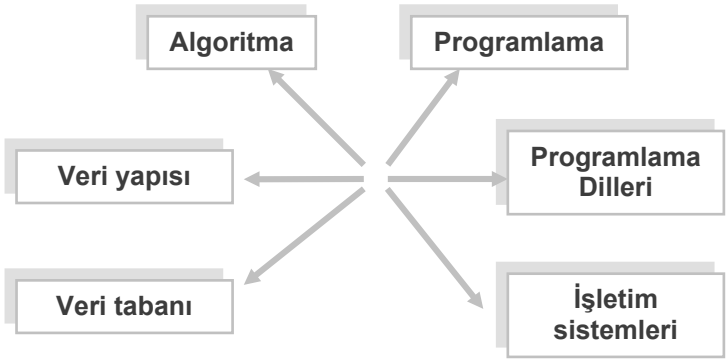
Programlamayı beyin, bir dil bilmeyi de el (veya diğer duyu organları – göz, kulak, ...) olarak düşünelim. El olmadan beyin yine düşünebilir, diğer işlemleri gerçekleştirebilir. Önemli olan beyindir. Ancak el, beyin olmadan hiçbir şey yapamayacaktır. Her ne kadar iki organda çok önemli gibi görünsede, beynin önemi ve görevi çok büyüktür.

> Stephen Hawkings, şu an bir tekerlekli sandalyede yaşayan ve dış dünya ile iletişimini sadece özel bir bilgisayar ile yapabilen ve çağımızın en büyük yaşayan fizikçilerinden biri olarak kabul edilir (bir hastalık nedeniyle).
>
> University College ve Cambbridge'deki Trinity College'da öğrenim gören Hawking, daha sonra Caius College'da araştırma görevlisi seçildi. 21 yaşında iken tedavisi olmayan bir hastalık olan amyotrofik lateral skleroz hastalığına yakalandı ve tekerlekli sandalyeye mahkum oldu. Hastalık Hawking'in konuşma yeteneğini de alıp götürdü. Ancak, dahi bilim adamı için özel olarak dizayn edilen bir bilgisayar, Hawking'in bir klavye aracılığıyla kurduğu cümleleri seslendirerek konuşmasına yardımcı oluyor. Hastalığın giderek ağırlaşan etkisine ve 60 yaşına basmasına rağmen Hawking halen çalışmalarına devam etmektedir. **[e1]**
>
> Hawkings, ellerini kullanamasada, beynini kullanarak, kuramlar oluşturmaktadır.

Konu seçimi

Kitap içerisinde birçok farklı konu bulunmaktadır. Bu kitapla, programlamanın geçtiği ve kullanacak olduğunuz birçok konuya giriş yaptım. Ancak kitap içinde bahsi geçen, İşletim sistemleri, Veritabanı sistemleri, Programlama dilleri, Yazılım mühendisliği .. v.b. konular başlı başına onlarca kitap ile ifade edilmektedir. Kitapta, bu konuların temelini anlattım. Aksi taktirde bu konuların hepsi zaten onlarca kitap kadar yer tutardı.

Birçok algoritma veya programlamaya giriş kitabı, içerik bakımından bu kitaptan farklıdır. Yada bu kitap birçok algoritma ve programlamaya giriş kitabından içerik olarak farklıdır. Bu kitaplarda, temel alınan programlama ise, bu konu üzerine yoğunlaşılmış ve ilerlenmiştir. Temel alınan konu algoritma ise, algoritma üzerine yoğunlaşılmış ve bu konuda ilerlenmiştir. Bu tip kitaplarda, size algoritma veya programlama öğretilemez! Çünkü programlama; işletim sistemi ile de veritabanları ile de, programlama dilleri ile de, algoritma ile de ilgilidir. Dolayısıyla, kitabı bir yönde ilerlettikleri için diğer konulardan bilgisiz kalacağınız için, konuları düzgün bir şekilde öğrenemezsiniz.

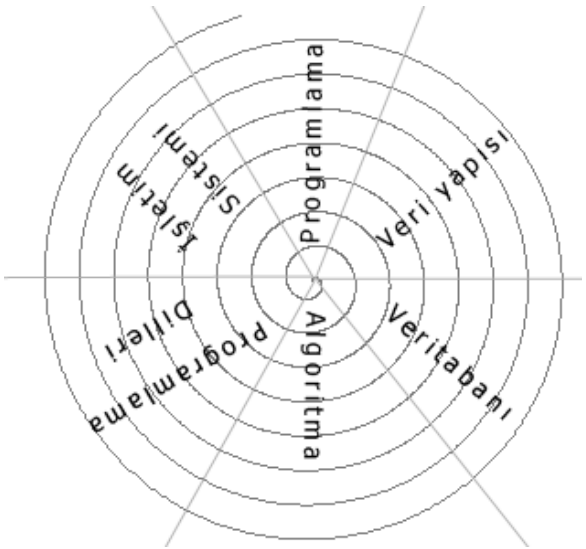


Figür 1.1 A : Genelde kitapların öğretim şekli

Yukardaki diagram, piyasadaki birçok kitabın öğretim şeklini anlatmaktadır. Her kitap kendine, bir yön seçer ve o yönde ilerler. Bir yönde ilerlerken, diğer konulardan neredeyse hiç bahsetmez ve sizin bu konudaki bilginiz az veya hiç olmadığından anlatılan kavramları anlamakta zorluk çekersiniz veya yüzeysel öğrenirsiniz. Bir yönde çok bilgi diğer yönde az veya hiç bilgi vermedikleri için tek bir yönde ilerlersiniz ve öğrendiğiniz konunun diğer konularla arasındaki bağlantısını anlayamazsınız.

Yukarıda bahsi geçen 6 konuyu giriş – orta ve ileri seviye olarak üçe bölelim. Algoritma kitabı alırsanız, bu kitapta, algoritmanın, giriş – orta – ileri seviyesi anlatılır ve programlamanın giriş seviyesi anlatılabilir.

Bu kitapta ise, birbiriyle ilgili olan bütün bu konuların öncelikle "giriş" seviyesi, daha sonra, algoritmanının orta seviyesi, programlamanın orta seviyesi anlatılmıştır.



Figür 1.1 B : Kitabın öğretim şekli

Yukarıdaki şekilde görüldüğü gibi, sırayla, her bölümden belirli miktarda bilgiler öğrenip, sonra, yine ilk bölümden daha ayrıntılı bilgiler öğrenecek ve daha sonra diğer bölümlerden daha ayrıntılı bilgiler..... öğreneceksiniz.

Kitap içinde bazı kısımlardan sonra **[Ara verme bölümü]** ifadesi kullanılmıştır. Kitabı okurken, sadece bu belirtilen kısımlarda ara vermenizi (günlük ara verme veya bir gün içinde

ara verme olarak) tavsiye ederim. Çünkü, bir konuyu ortadan bölerseniz, birbiriyle bağlantılı konuların bütünlüğünü bozduğunuzdan bazı konuları anlamakta güçlük çekebilirsiniz. Bunun haricinde, bir günde, 1'den fazla bölüm okumamanızı tavsiye ederim.

Dersaneler - Süre

Programlamayı öğrenmek için, bir dersaneye gidip, (orta veya ileri düzey) 4-5 bin ytl den başlayan fiyatlarla kayıt olup, dersaneyi bitirdiğinizde; bu kitabı okuduğunuzdaki kadar bilgi öğrenmeden mezun olursunuz. Amacım dersaneleri kötülemek değil. Ancak, birçok dersane, size, normalde 100-200 saat sürecek dersi, 500 saatte verir, 1-1,5 yıl boyunca kursa gitmek zorunda kalırsınız. Hem paranız boşa harcanmış olur, ama en önemlisi, zamanınızın boşa harcanmasıdır. Çünkü zamanın parasal bir karşılığı yoktur. Eğer var diyorsanız ben size o kadar ödeyeyim, bana zamanımızı verin...

Anlatmaya çalıştığım, dersanelerin birçoğunun "*öğrenci bankası*" haline geldiğidir. Normalde, bir öğretimci uzmana, bir dersin ne kadar sürede öğrencilere öğretilebileceğini sorulur. Uzman hesaplar, ve örneğin cevap olarak "200 saati" verir. Ancak finansal işlerle ilgilenen kişi, bu süreyi yetersiz bulur. Çok iyi kâr edilebilinmesi için sürenin 500 saat olması gerektiğini söyler ve bu şekilde karar alınır. Peki uzman (öğretimci) nasıl olacakta, 200 saatlik konuyu 500 saate yayacak?

Eğer normal bir şekilde anlatılsa, 200 saat sonunda öğrenciler, dersaneyi (konuyu öğrendikleri için) bırakıp gidebilir. O halde, öğrencilere, kodu kullanmayı, programlamayı, algoritmayı, mantığı değil de, sadece örnekleri öğretmekten başka çözüm kalmayacaktır. Yani, konuyu öğretip öğrencilerin, ileride kendi kendilerine daha iyi öğrenebilmelerini, karşılaştıkları sorunları çözebilmelerini ve sistemlere adapte olmalarını değilde, en sık karşılaşabileceği programlarla alakalı örnekler göstermeyi amaçlarlar. Öğrenciler, öğrenemediği için, örneklere bağımlı kalacaklardır. Böylece, uzman, bol bol örnek gösterecektir ve öğrenciler programlama mantığını kavrayamadığı için örneklere mahkum kalacaklardır.

Tabi birde, ucuz olsun, daha çok öğrenci gelsin diye, programlamaya girişi sadece 10 saatte öğreten dersanelerde vardır. Bu dersaneler sadece zaman kaybetmenize neden olur. Bu anlatılan malesef bir hikaye değildir. Üstüne üstlük, size herhangi bir bilimsel metotla değil, kendilerinin de öğrendiği gibi anlatırlar.

Öğretim Tekniği

Bu konu üzerine yazılmış birçok kitap, benim tabirimle, "sample based" yani "örnek tabanlı" kitaptır. Örnek tabanlı kitaplarda, size bir terimin, tanımını yaptıktan sonra hemen bir örnekle anlatmaya çalışırlar. Ancak bu yöntem öğrenmenizi zorlaştırır. Doğru olan ise, önce tanımı önemsemeyip, gerçek hayattan benzetme yaparak, o terimin ne anlama geldiğini bilimsel ifadelerle anlattıktan sonra pekiştirmek amacılığıyla örnek kullanmaktır. Yani örnek amaç değil araç olmalıdır. Konuyu düzgün anlatamadıkları için, bol bol örnek verip aradaki mesafeyi kapatmaya çalışırlar. Şimdi bunu size daha iyi açıklama için "***bir örnek verelim***".

2, 4, ? . Bu dizide, soru işareti yerine hangi sayı gelmelidir şeklinde bir soru sorulsaydı, bu soruyu çözmek mümkün olamazdı. Çünkü, 2 ile 4 arasındaki olası ilişkiler

- 4, 2'den 2 fazladır
- 4, 2'nin 2 katıdır
- 4, 2'nin karesidir.
- 4, 2'nin, 5 katından 6 eksiktir
- ....

şeklinde sıralanabilir. Bu gibi bir durumda, size daha çok örnek verilmesi gerekmektedir.

1, 2, 4, 8....

Yukardaki seriden anlaşıcağı gibi, ikinci sayı, ilk sayının 2 katı, sonraki sayılar bir önceki sayının iki katı şeklinde devam etmektedirler. İşte karmaşık olan bir terimi, size bol örnek göstererek bu şekilde öğretmeye kalkarlar. Halbuki, terim - örnek göstermektense;

“Elimizde bir dizi sayı bulunmaktadır. Bu sayılar, bir kurala göre dizilmiştir. İlk sayı 1’dir. Sonraki sayılar, bir önceki sayının, iki katıdır”

ifadesi kullanılsa, ve akabinde, örnek gösterilip zaten belirtilen kuraldan anlaşılmış olan örnek pekiştirilse, durum daha kolay kavranır.

İlkokulda, ödevimizi yapmadığımızda, öğretmenimiz konuyu öğrenmemiz için (ayrıca ceza olarak), ödev metnini 5 sayfa - 10 sayfa yazmamızı söyler. Böyle bir durumda yani bir metni 10 sayfa yazdığımızda işin mantığını kavramadığımız (veya çalışmadığımız için) beynimizi değil beyinciğimizi kullanarak bir tür refleks haline getirmeye çalışırız. Yani işin formülünü – anlamını öğrenmemiş oluruz. Sadece sonradan bu kadar çok tekrar sonucunda, işin mantığını kavrayabiliriz (her zaman değil!).

Meslektaşlarımızı (aslında tamda meslektaş sayılmayız) eleştirmek istemem ancak, bazı, "programlama nedir, nasıl yapılır", "algoritma", "programlamaya giriş" v.b. konulu makalelerde - kitaplarda, derslerde (üniversite - lise - dersane), daha programlamayı öğrenmeden bir örnek gösterilir. Yani siz daha programlama nedir bilmezken, C ile programlama gösterilir veya “Merhaba Dünya (merhaba dünya mesajı gösteren bir program)” şeklinde bir giriş yapılır. Peki, yeni doğan bir bebeğin, doğum öncesinde hiç bir gelişim evresi yok mudur? Yada yeni dünyaya gelen bir bebek nasıl olurda, “Merhaba Dünya” diyebilir. Bence bu öğretim şekli yanlıştır, bu yüzden, önce algoritmayı anlatmak adına, bu kitabı yazdım ve sizlere sunuyorum.

```
#include<stdio.h>
int main( void )
{
   printf("Merhaba Dünya");
}
```

Figür 1.1 C : Merhaba dünya örneği

Kitap içinde, bazı yerlerde, gerekli bilgilerin hafızanızda daha uzun süre kalabilmesi için, olağan dışı örnekler kullandım. Beynimiz içinde, yüz binlerce mantıklı bilgi bulunmaktadır. İşte bu yüzden, eğer kurulan örnekler mantıklı olursa hatırlamanız daha uzun, mantıksız olursa, hatırlamanız daha kısa süre alacaktır. Böylece hatırlanması gereken olayları, ilginç ve mantıksız oldukları için daha kolay hatırlayabileceksiniz.

> *Annem, 1977-81 arasında liseyi okurken arkadaşlarıyla kimsayal formülleri hafızalarında tutmak için bir kısaltma bulmuşlar (bu bilgiyi başka birilerinden öğrenmişler). Kimyada sülfürik asit olarak nitelendirilen ve formülü H_2SO_4 olan bu maddenin formülünü hafızalarında tutmak için, “**H**asan **2**, **S**alak **O**sman **4**” gibi bir ifade kullanmışlar. Aradan 26 yıl geçmiş olmasına rağmen, nerede konusu geçerse geçsin, annem bu bilgiyi halen hatırlayabilmektedir. Üstelik, bu bilgiyi 26 yıl içinde hiç kullanmamasına rağmen.*

İşte amacım da, hem kavramanızı kolaylaştıracak hem de hafızanızda uzun süre kalacak örnekler ve açıklamalar kullanarak kitabı ve programlamayı sizin için kolaylaştırmaktır.

Kitabın Farkı

Bu kitabın farkı: benim, bu yanlışlıkların farkında olabilmem, ve size, X saatte öğretilmesi gereken konuyu X saatte öğretmem ve amacımın ticaret değil öğretim olmasıdır. Yani, bu

kitabı aldıktan sonra bana veya yazdığım kitaplara bağımlı kalmayacaksınız (bazı dersanelerin yaptığı gibi), aksine, artık kendi kendinize her türlü programı tasarlayabilecek kapasiteye (biraz da deneyimle) geleceksiniz. Günlük olayları bile programsal olarak ifade etmeye başlayacaksınız. Bazen kendinizi programlamaya öyke kaptıracaksınız ki, çok geveze bir arkadaşınızı susturmak için bir program bile yazmayı düşüneceksiniz.

Bu kitabın okunması için en önemli gereksinim, programlamayı bilmemektir. Çünkü, öğrenmeyi ve inanmayı en zorlaştıran etken bilgidir. Programlama öğrenmiş kişilerde okuyamaz değiller, fakat, şimdiye kadar öğrendiklerini unutmak şartıyla...

Birçoğunuz "kod yazmaya başlayalım artık" düşüncesinde olabilir. Ancak, kod ile öğretilenler sonucunda sadece o kodları kullanmayı öğrenirsiniz ve kendiniz bir "şeyler" keşfedemezsiniz.

Eğer amacımız programlama öğrenmekse, doğal olan, programlamayı, programlama dilleriyle yapmamız gerektiği ve programlama dili öğrenmemiz gerektiğidir. Dolayısıyla, aslında bu kitap bir DİL kitabıdır. Yani bu kitabı, bir yabancı dil (ingilizce, ispanyolca...) kitabı okuyormuş gibi okumalıyız.

Dil kitaplarının dayandığı temel,

"1. dilde ifade edilen X terimi, 2. dilde Y şeklinde ifade edilir"
"1. dilde ifade edilen Z terimi, 2. dilde T şeklinde ifade edilir"

şeklindedir.

Bu kitap benzeri işlemleri anlatacaktır ancak, bu ifadelerden ibaret değildir. Yani aslında bu kitapta bir çeşit gremer öğreneceksiniz. Piyasada, yabancı dil öğreten binlerce kitap bulunur. Ancak hiçbir kitap, herhangi bir yabancı dili nasıl öğreneceğinizi anlatmaz!!! Sadece o dili öğretir. Ben size, herhangi bir dili öğrenebileceğiniz bir kitap veriyorum.

<u>İKİ AMACIMIZ BULUNUYOR:</u>

İngilizce öğretilirken, nasıl aşağıdaki gibi herhangi bir metni, ingilizcedeki haline

"Orada birşey var"
"There is something over there"

çeviriyorsak, <u>bir programıda işte bu şekilde metinleri çevirerek, yapacağız</u>! İşte bunun için önce (1. amacımız) hangi metinleri nasıl yazacağımızı öğrenmeliyiz sonrasında ise bu (2. amacımız) metinleri nasıl bilgisayar dillerine çevireceğimizi öğrenmeliyiz.

Kitap içinde, "ÇEVİRME" ifadesi geçen kısımlara dikkat ediniz. Bu ifadelerin geçtiği bölümlerde, bir metnin nasıl programlama diline çevrileceği anlatılmıştır.

1.2 Algoritma

1.2.1 Algoritmaya giriş

Bilgisayar Bilgisayarlar; onlara anlatılabilen (yada ifade edilebilen) belirli emirleri; işleyen, belirli verileri depolayabilen, gerçekleştirdiği bu işlemleri insanlara göre daha hızlı ve hatasız tamamlayabilen (*bilgisayarlar hata yap**a**mamaktadır – sadece onları programlayan programcılar hata yaparsa, bilgisayarlarda bu hatalı emirleri gerçekleştirmiş olduğu için hata yaparlar*) elektronik makinelerdir.

Bilgisayarları, kontrol etmek için bilgisayar programlarını kullanırız ve programlarla bilgisayarları yönetiriz. Yani programlar, bilgisayarların **direksiyonudur** veya **dizginidir** diyebiliriz.

Konuya başlamadan önce, inanmanız gereken, bu işin aslında düşündüğünüzden daha kolay olduğudur. İşin zor kısmı, sizi, programlama ve algoritmanın kolay olduğuna inandırmaktır, programlama öğretmek değil!!!

Programlama öğrenmek için okuduğunuz bir kitabın algoritma konusu ile başlaması olağandır. Çünkü, her tür planlama ve programlamanın temeli, algoritma kurmakla mümkün olacaktır. Bu kitabı okuyan herkes algoritma kurabilecek yeteneğe sahip olduğundan dolayı, okuyucular için; "programlama yapabileceği garantisi" verilebilir (tabi bu hemen her programı yazabileceğiniz anlamına gelmemektedir). Çünkü, "**okuma**-yazma" bile algoritma gerektirmektedir, ve siz okurlar, **okuma** bildiğiniz için algoritma kurabilir, dolayısıyla programlama yapabilirsiniz.

> Algoritmaya giriş yapmadan önce, bir soruya cevap vermeye çalışalım: "**Neden düşünürüz**?". Şuan, "Neden düşünürüz" sorusunun cevabını bulmak için **düşünüyorsunuz, çünkü bu sorunun cevabını bilmiyorsunuz**. O halde, düşünmenin nedenlerinden birinin, bir **soru**nun cevabı bulmak olduğunu söyleyebiliriz. Çünkü eğer bu sorunun cevabını biliyor olsaydınız, düşünmenize gerek kalmazdı ve direkt olarak cevaplardınız.

Algoritma **Algoritma**, "**algorithm** is a procedure or formula for solving problems **[e2]**" yani; "problemleri çözmek için tasarlanmış bir prosedür veya formül" anlamında gelmektedir.

Daha açıkcası, **bir problemin çözümü için izlenecek yol** (yöntem - metod) **yani** (bu yöntem içerisinde bulunan) **bir dizi işleme verilen addır**. Bir dizi işlem: tek bir işlem veya birden fazla işlem olabilir. Eğer yapılacak bir dizi işlem yok ise, ortada problem olan bir durum söz konusu değildir.

Algoritma Kısa Tarihi
*Algoritmanın kurucusu dokuzuncu yüzyıl başlarında yaşayan **Müslüman-Türk** matematik alimi Ebu Abdullah Muhammed bin Musa el-Harezmi'dir. Matematikçiler için temel olan Kitab-ül Muhtasar fi Hesab-il Cebri ve'l-Mukabele adlı eseri meşhurdur. Kitabın aslı, Oxford'daki Bodliana kütüphanesindedir. Matematikteki şöhreti on altıncı yüzyılda Avrupa'yı etkisi altına almıştır. Harezmi'nin ismi Avrupa'da türlü şekillerle söylenmiştir. Latince'de "Alkhorismi" şeklinde söylenerek bulduğu metod Algoritma (algorizme) olarak literatüre geçmiştir.* **[e3]**

Bir problemi çözmek için, düşünme eylemi gerçekleştirdiğimiz her evrede, algoritma kurarız veya daha önceden (bizim veya başkası tarafından) kurulmuş olan algoritmaları kullanırız. Bazı durumlarda da, geçerli olan durumlar için algoritma kur**a**mayabiliriz. Gelişen dünya – bilişim sayesinde artık her tür problem çözülmeye, çözüm yolları da herkes tarafından öğrenilmeye (internet veya kitaplar tarafından) başlandı. Artık bu noktadan sonra, önemli olan, algoritma kurabilme değil,

kuracağınız algoritmaların; **ne kadar hızlı, ne kadar kapsamlı, ne kadar ucuz ve ne kadar yüksek performanslı** olduğudur.

Algoritma kurmanın nedeni, **düşünmektir** ve düşünmenin nedeni ise, **problem çözmekdir (dört paragraf öncesinde, düşünmenin nedenlerinden birinin, bilinmeyen bir sorunun cevabını bulmak olduğunu söylemiştim, bir sorunun cevabının bilinmemesi de bir problemdir)**.

Problem Problem; istenilmeyen bir durumdur**[durum: Bir zaman kesiti içinde bir şeyi belirleyen şartların hepsi, vaziyet, hâl, keyfiyet, mevki, pozisyon]** (bir kullanıcı, kişinin veya topluluğun istemediği bir durum – değiştirilmesi gerekli olan bir durum) ve algoritma kurularak, bir dizi işlem oluşturulur. Bu işlemler, o anki durumu, problem durumundan çıkarır ve istenilen hâle getirir.

İlk okul çağlarından hatırladığımız "matematik problemleri" de aynı esasla çözülür. Problemlerde, bilinmeyen bir durum söz konusudur. Bir işçi problemi düşünelim... Evimizi boyatacağız ve boya ustasının günlüğü 40 YTL (veya artık 2009'da isek, 40 Lira) olsun ve boyacı bir günde sadece bir oda boyayabilsin.. Bizde buradan oda sayısı ile günlük masraf olan 40 lirayı çarpıp boyacıya vereceğimiz ücreti hesaplarız...

Yapay zeka (insan gibi düşünebilen – çözüm üretebilen – sistemler geliştiren bir bilim dalı) sistemlerinde de; insan gibi düşünen sistemler geliştirmede ilk temel alınan yöntemlerden biri, başlangıç durumundan, istenilen duruma ulaşan bir formül bulmaktır. Algoritma kurma evresi aşağıdaki diagramdaki gibi gerçekleşmektedir.

Figür 1.2.1 A : Problem ve Çözüm durumu

Şimdiye kadar ki bölüm tamamen teorikti, bu öğrendiklerimizi pratiğe dökelim.

Bu anlatılanların aklınızda daha uzun süre kalmasını ve anlaşılır olmasını sağlamak amacıyla, geçmişe çocukluğunuza dönelim. Hepimiz, Hansel ve Gretel hikayesini hatırlarız. Bu hikaye içinde geçen 1-2 cümleyi yazayım. (*Hansel ile Gretel iki kardeştir ve üvey anneleri onların kaybolmasını ister.*)

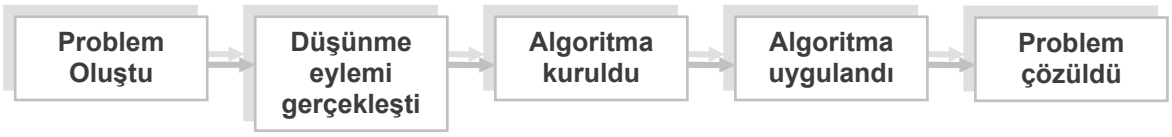
".... Hansel zeki bir çocukmuş. Sabah ormana doğru yürürlerken, akşam yemeğinde cebine sakladığı kuru ekmeğin kırıntılarını (yere iz bırakıp kaybolmamak ve daha sonra bu izi takip ederek evin yolunu bulmak için için) yere saçıp arkasında bir iz bırakmış....."

Çok basit bir örnek olarak; Hansel, evin yolunu kaybetmemek için bir algoritma yani çözüm bularak, ekmek kırıntılarını yere işaret bırakmak için kullanmıştır.

Algoritma üstünde neden bu kadar yoğun olarak durulduğunu ileri ki bölümlerde daha iyi kavrayacaksınız. Birçoğunuzun, "***bu konuları zaten herkes biliyor***" dediğini tahmin ediyorum. Bazılarınızda, "***bu konular aslında zor olduğu için mi üzerinde bu kadar duruluyor?***" şeklinde düşünüyor olabilir. Bu düşüncenin aksine algoritma konusu inanılmaz derecede kolaydır. Ancak, algoritmanın öneminin sürekli olarak aklımızda olmasını sağlamak gerekmektedir. İşte bu yüzden bir kaç örnek daha vererek, algoritmanın hayatımızın bir parçası olduğunu göstermek daha doğru olacaktır.

Öğrenmenizi daha kolaylaştırmak için, ilginç bir olayı aklınızda tutmanızı öneririm. Örneğin, yanınızda, bir uzaylı(geri kalmış bir gezegenden gelen – ilk kez dünyamıza gelen) veya medeniyetten uzak (eğitilmemiş) kalmış 12-15 yaşlarında bir çocuk olduğunu, ben kitapta algoritma konusunu anlatırken, sizde sanki onun yanındaymışsınız ve öğrettiklerimi tasdik edermişcesine ona destek ve yardımcı olduğunuzu düşünün. Siz algoritma kurmasını biliyorsunuz ve bu konu ile ilgili anlatacaklarımı da duyduğunuzda çok kolay bir şekilde anlayabilirsiniz. Yani ben sadece bir kitap yazdım, siz oradan (müfredat gibi) bakıp, aynı bir öğretmen gibi birisine anlatıyormuşsunuz gibi düşünün. Hatta içinizden "bunları da gördük biz bu hayatta... kolay terimler... bunları çok rahat yapabilirim..." ifadelerini geçirebilirsiniz. Bu durum sizi hem rahatlatacak hemde konuları daha kolay şekilde HATIRLAMANIZI sağlayacaktır. Hatırlamanızı diyorum çünkü, siz zaten algoritmanın ne olduğunu biliyorsunuz, tek bilmediğiniz, "**algoritmayı bildiğiniz ve algoritma kurmanın ne kadar kolay olduğunu bildiğiniz**"'dir.

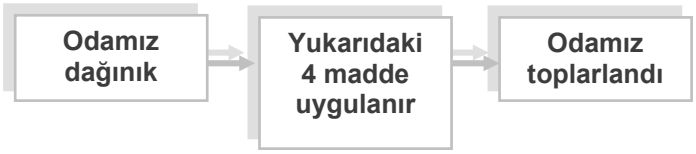
İnsan beyni, yukarıdaki figürde gösterildiği gibi; bir başlangıç durumundan, sonuç durumu elde etmek için işlemler bulmak üzere çalışmaktadır. Tabi ki, algoritma kurma öncesinde, problemin oluşması, ve kurulan algoritmanın sorunu çözmesinden sonra, sorunun çözülmesi durumları da eklediğimizde daha ayrıntılı bir figür elde ederiz.



Figür 1.2.1 B : Bir problemin gerçekleşmesi ve çözümü

Örneğin; (varsayalım ki) çalışma odamız dağınık ve odamızın dağınık olması bizim için bir problem, çünkü biz istenilen durum olarak odamızın düzenli olmasını istiyoruz. İşte, odamızın dağınık durumundan → düzenli durumuna geçebilmesi için bir algoritma kuruyoruz;

- Eşyalarımızı yerine yerleştiriyoruz
- Gereksiz kağıtları çöpe atıyoruz
- Açık kitapları kapatıp rafa kaldırıyoruz
- Odamızı temizliyor ve süpürüyoruz



Figür 1.2.1 C : Örnekteki algoritma işlemleri, ilk ve son durum

Bu adımlar sonucunda (öyle ki bu adımlar, yukarda bahsettiğimiz bir dizi işlemdir) odamız düzenli hale geliyor. Sonuçta diyebilirizki, algoritma istenilmeyen bir durumdan istenilen bir duruma **mantıklı** geçiş için bir yöntemdir.

Birçoğunuzun aklında, "***böyle bir durum için neden algoritma kurduk?***" sorusu oluşmuştur. Belki hergün, belki her hafta yapıyor olduğumuz ve **BASİT** olan bir problem için neden algoritma kuralım? Düşüncemiz kesinlikle **doğrudur**. Çünkü, algoritma;

- **ya daha önceden kurulmamış durumlar için**
- **ya daha önceden kurulmuş ancak unutulmuş durumlar için**
- **ya daha önceden kurulmuş ancak başarıya ulaşamamış durumlar için**

kurulur. Hepimiz, daha önce ya odamızı topladığımız yada toplayanları gördüğümüz, odamızı toplamamız gerektiğinde neler yapacağımızı düşündüğümüz için, böyle basit durumda algoritma kurmaya gerek görmeyiz. Tabi ki haklıyız da. Burada belirtilmek istenen, algoritma kurmanın günlük hayatta önemsenmeyecek bir ayrıntı olduğu ve daha önce onlarca hatta yüzlerce algoritma kurduğumuz veya öğrendiğimizdir.

Şimdi de, daha önce algoritmasını kurmadığınızı düşündüğüm bir konu hakkında, algoritma kurma örneği vereyim: Elimizde, 3 ve 5 litrelik birer su bidonu olduğunu ve bir çeşme başında olduğumuzu düşünelim. Varsayalım ki, görevimiz, 5 litrelik su bidonu içine, 2 litre su doldurmak olsun. (elimizde bir tartı veya benzeri bir araç bulunmamaktadır)

Şuan bu problemi çözmek için gerekli yolun ne olduğunu bilmiyoruz. Ancak, bu soruyu hepimiz rahatlıkla çözebiliriz. Bidonlara ve suya uygulayabileceğimiz işlemlerin listesini çıkaralım:

- bir bidona su ekleyebiliriz (çeşmeden)
- bir bidondaki suyu dökebiliriz
- bir bidondan diğer bidona su dökebiliriz

TRT'de yayınlanan "Bir kelime, bir işlem" yarışmasını hatırlayalım. Bu yarışma programında "işlem" bölümünde, amaç; verilen bir dizi rasgele sayıyı kullanarak, bir dizi işlem sonrasında, istenilen (hedef) sayıya ulaşmaktır. Yarışmacılar, belirli bir algoritma kurarak, hedef sayıya ulaşmaya çalışırlar.

Bu sorunu çözmek için algoritma kurarken;

- 5 litrelik bidonda, 2 litre su olabilmesi için, öncelikle 5 litrelik bidonu doldururuz
- Sonra, 5 litrelik bidondaki suyu yavaş yavaş, 3 litrelik bidona dökeriz. 3 litrelik bidon dolduğu an (5-3=2), 5 litrelik bidonda, 2 litre su kalmış olacaktır.

Figür 1.2.1 D : Örnekteki algoritma işlemleri, ilk ve son durum

Fonksiyonlar konusunda ayrıntılı bilgiler ileriki bölümlerde verilecektir. Ancak, biz matematiksel fonksiyonları hatırlayalım. Bir kaset çaları yani teybi düşünelim. Teybin görevi, içine konulan kasedin içindeki şarkıları çalmaktır. Dikkat edersek teybin görevi hiç değişmemektedir. İçine hangi kaseti koyarsak o kasedi çalabilmektedir. Her bir şarkıyı çalan ayrı ayrı araçlar keşfetmektense, aynı işlevi gerçekleştirdiği için, bir teyp alırız ve bu teybe yerleştireceğimiz kaset ile istediğimiz şarkıyı çalarız. Fonksiyonları bir teybe benzetebiliriz. İçine hangi değeri (kasedi) yerleştirirsek o değerle işlem yapar ve ona göre çıktı (şarkıyı) verir.

Figür 1.2.1 E : Fonksiyon örneği

y = f (x)

Aslında günlük hayatta, bir şeyleri girdi olarak alan ve sonra bir tür çıktı veren herşeyi bir fonksyion gibi düşünebilirsiniz. Bir devlet dairesine gittiniz, bir resmi evrak (girdi) verdiğiniz, bu evrağa numara atıldı, imza atıldı, mühür basıldı ve size geri (çıktı) verildi. İşte orada, bu işlemleri gerçekleştiren memur, bilgisayarda fonksiyon olarak adlandırılabilir.

Burada, kaseti x (girdi), teybi, fonksiyon (f) ve çalan müziği de y (çıktı) olarak ifade edebiliriz. Unutmayalım ki, yerleştirilen kasete göre bir müzik çalınacağı için, giren x değerine göre bir y değeri çıkacaktır, ve fonksiyon görevini gören teyp hep aynı işlemi (kaset çalma) gerçekleştirecektir. Aynı şekildede ; $y=f^{-1}(x)$ şeklinde bir ifadede de, (üssü -1 ifadesi olduğu için) ters işlem yapılacaktır. Yani bu sefer teyp dışarıdaki bir ses kaynağından sesi alıp kasete kayıt edecektir.

y = f (x)

şeklinde bir fonksiyon ve eşitlik düşünelim. Bu eşitlikte, X istenilmeyen (problem durumu) durumunu, y ise, sonuç (istenen) durumunu simgelesin. F olarak ifade edilen terim ise, matematikten hatırladığımız bir fonksiyonu simgelesin. F fonksiyonu, girdi olan X (yani istenilmeyen durum) durumuna belirli işlemler uygular ve sonuç olarak, bize, Y(yani istenilen durum) durumunu çıkarır

Son durum = algoritma işlemleri (İlk durum)

Figür 1.2.1 F : Fonksiyonların, algoritmik gösterimi

Başka bir örnek olarak; (varsayalım ki) otomobilimizin lastiği patladı, ve bu durum (lastiğin patlak olması) bizim istemediğimiz bir durum – yani bir problemdir. Çözüm durumu olarak; lastiğin patlak olmamasını veya, otomobildeki patlak lastiği başka bir yedek lastikle değiştirilmiş olmasını verebiliriz.

```
Lastik sağlam = işlemler ( Lastik patlak )
```

İşlemler adlı fonksiyon içerisindeki yapılacak olan işlemleri, hepimiz tahmin ederiz. Fakat, bu işlemi bir robota öğrettiğimizi düşünelim. Bilim kurgu filmlerinde, robotların olduğu durumlarda, bazı sahneleri, robotun göz kesitinden görürüz. Bu sahnelerde, robota, "düşmanı öldür", "aracı sür" gibi emirlerin verildiğini görmüşüzdür. Varsayalım ki, yıl 2020, otomobilimizi sürüyoruz, yanımızda yardımcı robotumuz bulunuyor. Lastik patlıyor ve bu durumda, robotun gözü önüne yine emirler geliyor. Robotun bu emirleri anlayarak, işlemi gerçekleştirdiğini düşünelim.

Durum: Sahibin kullandığı aracın lastiği patladı

- Kriko kullanarak otomobili kaldır
- Patlak lastiği çıkar
- Yedek lastiği tak
- Krikoyu gevşeterek otomobili yere indir

Gerçek hayatta her gün, onlarca, yüzlerce algoritma kuruyor veya kurulmuş olan algoritmaları görüyor – kullanıyor olabiliriz. [Bu olayı, yani robotun yukarıdaki işlemleri, gözü önünde görmesini, gözlerinizi kapatarak zihninizde canlandırınız]

Algoritma, matematiksel olarak ifade edilebilir. Aslında dış dünyadaki olayların, eylemlerin bile matematiksel tanımı bulunmaktadır. Programlamanın kökü algoritmadır. Algoritmanın