

Algoritma ve Programlamaya Giriş Dersi

3. Hafta

Algoritma Hazırlama

- Programlama Adımları
- Algoritma Terimleri
- Algoritma Adımları
- Algoritma Örnekleri

Bu Haftanın Konu Başlıkları

- İnsanın yaşamı boyunca yaptığı **plan** kavramının eşdeğeri bilgisayar dünyasında **algoritma** olarak karşımıza çıkar.
- Farklı programlama dilleri, dünya üzerinde konuşulan farklı dillere benzetilir. Algoritma ise **evrensel bir dildir**. Tanım olarak:
- **Algoritma:** Bir problemin çözümünde yer alan işlemleri gerçekleştirirken izlenecek **adımlar dizisine** algoritma denir.
- Algoritmalar **açıkça belirtilmiş, bir başlangıcı ve sonu olan işlemler kümesidir**.
- Algoritma bilgisayar yazılımı için düşünüldüğünde; yazılımın hangi veriyi nasıl alacağı, nasıl bir işlem dizisinden geçireceği ve ne yöntemle çıktı vereceği gibi konular anlatılır.
- İlk algoritma örnekleri El-Harezmi'nin (9.YY) 'Hisab-el Cebir ve El Mukabala' kitabında sunulmuştur ve algoritma kelimesi de El-Harezmi'nin isminden gelmiştir. (Latince **Alkhorismi** → **Algorizme**)
- **Akış Diyagramı:** Algoritmanın özel geometrik şekillerle çizilmiş görsel haline akış diyagramı denir. Algoritma gibi evrensel bir dildir.


Algoritma ve Akış Diyagramı

- 1 • Problemin irdelenip analiz edilmesi
- 2 • En uygun çözüm yolunun belirlenmesi
- 3 • Algoritmanın hazırlanması (Sözde kod da yazılabilir)
- 4 • Akış şemasının hazırlanması
- 5 • Uygun dilin seçilmesi
- 6 • Seçilen dil ile algoritmanın kodlanması
- 7 • IDE vasıtasıyla hataların giderilmesi
- 8 • Bilinen verilerle test edilip doğrulanması

Program Yazma Adımları

- **1. Tanımlayıcı:** Tamamen programcı tarafından oluşturulan adlandırmalardır.
- Değişken, sabit, fonksiyon, sınıf, nesne gibi yapılara verilen adlardır.
- İlgili terimi çağrıştırmaya tavsiye edilir.
- İngiliz alfabesindeki **A-Z** veya a-z arası 26 harf kullanılabilir.
- 0-9 arası rakamların tümü ve simgelerden sadece alt çizgi (_) kullanılabilir. **Boşluk karakteri kullanılamaz.**
- Tanımlayıcı ismi harf veya alt çizgi ile başlayabilir. Rakamla başlayamaz (veya sadece rakamlardan oluşamaz).
- İlgili programlama dilinin özel terimleri veya komutları tanımlayıcı adı olarak kullanılamaz.
- Kolaylık ve düzen açısından geliştirilmiş çeşitli notasyonlar kullanılabilir:
 - *Macar Notasyonu (txtYazi, intAlan),*
 - *Pascal Notasyonu (OgrenciNo, VeriTabaniAdi)*
 - *Camel Notasyonu (alanAdi, ogrenciSoyadi)*
 - *Snake (Altçizgi) Notasyonu (urun_sayisi, arac_plaka_no)*
 - *Büyük Harf Notasyonu (HARFSAYISI, PERSONELDOGUMTARIHI)*

Algoritmada/Programlamada Kullanılan Terimler

- **2. Değişken:** Programın her çalışmasında farklı değerler alabilen veya çalışma anında değeri değiştirilebilen, ayrılmış bellek/bilgi alanlarına değişken denir.
- Değişken adı tamamen programcıya bırakılmış olsa da anlaşılır ve çağrışım yapacak şekilde yazılması en mantıklı yoldur.
- **3. Sabit:** Kodlama esnasında tanımlanıp değeri belirlenen ve program çalışırken değeri değişmeyen ifadelerden denir.
- Değişken adlandırma ve değer atama kuralları ile oluşturulur.
- **4. Aktarma:** Herhangi bir değeri bir değişkene, bir nesne özelliğine veya bir değişken içeriğini başka bir değişkene aktarmayı ifade eder.
- Aktarma operatörü (=) kullanılarak yapılan bu işlem, değişken türleri ve tür farkları dikkate alınarak yapılmalıdır.
- Aktarma işleminde değer sağdan  sola doğru geçer.
- Soldaki değişkenin eski değeri aktarma sonrasında silinir ve yeni değerini alır.

Algoritmada/Programlamada Kullanılan Terimler

- **5. Sayaç:** Programlarda bazen işlemlerin belli bir sayıda gerçekleşmesi, bazen de yapılan işlemlerin tamamının veya belli şarta uyanlarının sayılması gerekir. Bu durumlarda sayaç kullanılır.
- Sayaç satırı çalıştığında, belirtilen kadar artırma/azaltma sonucu sayaç yeni değerini alır.
- **6. Karar Verme:** Belirlenen bir şarta/şartlara göre programın gidişatının dallanacağı yapılara denir.
- Karşılaştırma operatörleri kullanılır. Gerekirse karşılaştırmayı genişletmek için mantıksal operatörler de devreye girer.
- if-else, ? operatörü ve switch-case gibi deyimlerle koda eklenir.

Algoritmada/Programlamada Kullanılan Terimler

- **7. Döngü:** Aynı/benzer işlemleri belli bir sayıda gerçekleştirmek, her adımda ardışık işlemler yaptırmak veya dinamik bir şekilde veriler üretmek, almak, ekrana yazmak için döngüler kullanılır.
- Rutin işlemleri daha az satırla yazabilir ve otomatikleştirebiliriz.
- Döngüde iterasyon değişkeni adı verilen değişkene başlangıç değeri, alabileceği son değer ve artış/azalış miktarı tanımlanmalıdır.
- **8. Ardışık Toplama:** Programlarda bir değişkenin değeri üzerine ardışık olarak gelen yeni değerleri ekleme işlemidir. Sayaca benzer yapıdadır.
- **9. Ardışık Çarpım:** Bir değişkenin, ardışık olarak gelen yeni değerlerle sürekli çarpılması işlemidir. Faktöriyel hesabı gibi yapılarda tercih edilir.
- Ardışık toplamada başlangıç değeri 0; ardışık çarpımda ise 1 olarak belirlenmelidir.

Algoritmada/Programlamada Kullanılan Terimler

- Problem etraflıca düşünülür ve tüm olasılıklar gözden geçirilir.
- En az komutla, en kısa sürede ve en doğru sonuca hassasiyetle ulaştıracak yöntem belirlenir.
- Tanımlayıcı isimleri belirlenir.
- Her işlem adımına bir numara verilir. (İlk adım **Başla**, son adım **Bitir**)
- Problemden işlenecek veriler girilir veya ilgili ortamdan alınır.
- Yapılacak işlemler ve kullanılacak yöntemler **açıkça** belirtilir.
- Bulunan sonuçlar görüntülenir ya da belirlenecek bir yerde saklanır.
- Algoritma tüm ayrıntıları anlatmalıdır. Boşluk olmamalıdır.
- Esnek olmalı ve değişikliklere uygun olmalıdır.
- Herhangi bir programlama diline bağımlı olmamalıdır.

Algoritma Hazırlama Kuralları

- Program geliştirme sürecini kolaylaştırır.
- Hatalı kodlama vakalarını azaltır.
- Ekstra gibi görünse de program yazımı için geçen genel süreyi kısaltır.
- İşlem akışını gösterdiğinden hata bulmayı kolaylaştırır.
- Sonradan yapılacak güncellemelerde kolaylık sağlar.

İyi bir program,

- İşlemleri hatasız gerçekleştirip doğru sonuçlar üretmelidir.
- Hızlı olmalıdır.
- Bellekte mümkün olduğunca **az yer** kaplamalıdır.
- Sistem kaynaklarını gereksiz yere kullanmamalıdır.
- Kodları sade olmalı, değişiklik ve güncellemeler kolayca yapılabilir.

Algoritma/Akış diyagramının avantajları

- 1) Başla
- 2) Birinci sayıyı (**A**) ve ikinci sayıyı (**B**) oku (*kullanıcıdan al*)
- 3) Eğer $A = B$ ise ekrana yaz: "A B'ye eşit", Git 6
- 4) Eğer $A > B$ ise ekrana yaz: "A B'den büyük", Git 6
- 5) Ekrana yaz: "B A'dan büyük"
- 6) Bitir

- 1) Başla
- 2) Üçgenin kenarlarını (**a**), (**b**), (**c**) oku
- 3) Eğer $(a = b)$ ve $(b = c)$ ise Yaz "Eşkenar", Git 6
- 4) Eğer $(a = b)$ veya $(a = c)$ veya $(b = c)$ ise Yaz "İkizkenar", Git 6
- 5) Yaz "Çeşitkenar"
- 6) Bitir

Algoritma Örnekleri

SUBÜ - Sakarya MYO

10

Bilgisayar Programcılığı

- 1) Başla
 - 2) Bir sayı (**N**) oku
 - 3) **Faktor** = 1
 - 4) **S** = 1
 - 5) **Faktor** = **Faktor** x **S**
 - 6) Eğer $S < N$ ise $S = S + 1$, git 5
 - 7) Yaz **Faktor**
 - 8) Bitir
-
- 1) Başla
 - 2) **T** = 0
 - 3) **I** = 1
 - 4) Sayı adedini oku (**N**)
 - 5) Eğer $I \leq N$ ise git 6, değilse git 10
 - 6) Sayıyı oku (**Sayı**)
 - 7) $T = T + \text{Sayı}$
 - 8) $I = I + 1$
 - 9) Git 5
 - 10) **Ort** = T / N
 - 11) Yaz **Ort**
 - 12) Bitir

Algoritma Örnekleri

SUBÜ - Sakarya MYO

11

Bilgisayar Programcılığı

- *Algoritma Geliştirme ve Programlamaya Giriş*, Dr. F. Vatansever, 7. Baskı, Seçkin Yay. 2009-Ankara
- *Algoritma ve C# Programlama*, Erhan ARI, 2. Baskı, Seçkin Yay. 2015-Ankara
- http://enformatik.kku.edu.tr/dokumanlar/BOLUM-1_ALGORITMA_AKIS_SEMA.pdf

Kaynaklar