

ALGORİTMA VE AKIŞ ŞEMALARI

ADEM AKKUŞ

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

- Algoritma, 800'lü yıllarda yaşamış olan Acem matematikçi Muhammad ibn Musa al-Khwärizmi'nin yaptığı çalışmalarda ortaya konmuştur. 12. yüzyılda bu çalışmalar Latince'ye çevrilirken, çalışmaların sahibi olan al-Kharizmi'nin adından ötürü yaptığı bu çalışma “algorithm” olarak çevrilmiştir. Bu kelime Türkçe'ye ise algoritma olarak girmiştir.
- Tarihçesinden de görüleceği üzere algoritma, bilgisayar dünyasına girmeden önce, matematik alamındaki problemlerin çözümü için kullanılmaktaydı. Daha sonra bilgisayarların geliştirilmesiyle bu alandaki problemlerin çözümünde de kullanılmaya başladı.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

- Algoritma, en basit ifadeyle, bir problemi çözmek için takip edilecek sonlu sayıda adımdan oluşan bir çözüm yoludur.
- Diğer bir ifadeyle algoritma, bir problemin mantıksal çözümünün adım adım nasıl gerçekleştirileceğinin sözlü ifadesidir.
- Algoritma ile oluşturulan çözümler sözel olarak ifade edildiğinden daha standart herkesin gördüğünde ortak olarak aynı sonucu çıkarabileceği hale getirmek için akış diyagramları kullanılır. Akış diyagramları sembollerden oluşmaktadır. Her sembolün belli bir işlevi vardır.
- Algoritması oluşturulmuş bir problemin bilgisayar ortamına aktarılmış haline **program** denir. **Program, problemin çözümünde yapılması gereken işlemler bütünüdür kod karşılığıdır.**
- Algoritmaların program haline getirilmesi için programlama dilleri kullanılır. Programlama dilleri kullanılarak yazılımlar geliştirilir.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmanın temel özellikleri şunlardır:

- **1. Kesinlik:** Algoritma içindeki adımlar herkes tarafından aynı şekilde anlaşılabilir olmalı, farklı anlamlara gelebilecek bulanık ifadeler içermemelidir.
- **2. Sıralı Olma:** Her algoritma için bir başlangıç durumu söz konusudur. Çözüm, bu başlangıç durumu gözönünde bulundurularak gerçekleştirilir. Adımların hangi sırada gerçekleştirileceği çok önemlidir ve net bir şekilde belirtilmelidir.
- **3. Sonluluk:** Algoritma sonlu sayıda adımdan oluşmalı, sınırlı bir zaman diliminde tamamlanmalıdır. Her algoritmanın bir son noktası, bitişi olmalıdır.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

- Bir problemin çözümüyle ilgili adımlar kesin, sıralı ve sonlu bir şekilde ifade edildiğinde problem çözümünün algoritması çıkarılmış olur.

Problem Çözme

Problem çözmede iki temel yöntem vardır:

- Deneysel, deneyimsel ya da deneme yanılma yöntemi
- Algoritma geliştirmek

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Adımları

- **Problemi Tanımlamak:** Algoritmanın amacı belirli bir problemi çözmektir. Problemi ne kadar iyi anlarsak, algoritmayı geliştirmemiz o kadar kolay olur.
- **Girdi ve Çıktıları Belirlemek:** Problemi iyi tanımlamak için başlangıç ve bitiş noktalarını çok net belirlememiz gerekir. Bulacağımız şey, problemin çözüm yoludur. Ama problem çözüldüğünde ortaya çıkacak şeyi, problem içerisindeki parametreleri bilmeliyiz ki algoritmamızı geliştirelim. Bunun için algoritmanın girdilerini ve çıktılarını iyice kavramalıyız.
- **Çözüm Yolları (Algoritmalar) Geliştirmek:** Bir problemin çözümü için çoğunlukla birden fazla seçenek vardır. Eğer yeterince vakit varsa, en iyi çözüm yolunu bulmaya çalışmalıyız.
- **Çözümün Sınanması ve İyileştirilmesi:** Algoritmayı geliştirdikten sonra, kodlamadan kağıt üzerinde nasıl çalışacağını sınamalıyız. Bunu yaptığımızda eğer algorithma bir eksiklik ya da hata çıkarsa, bunu düzeltmeli ve tekrar sınamalıyız.

Bilgisayar programı algoritmalarında iki maddeye daha ihtiyacımız vardır

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Adımları

- **Algoritmanın Kodlanması:** Geliştirilen algoritma belirli bir programlama dilinde kodlanır. Böylece kağıt üzerindeki çözüm bilgisayar üzerinde çalışabilecek hale gelmiş olur.
- **Kodun Sınanması ve İyileştirilmesi:** Yazılan kod da algoritmada olduğu gibi sınanır. Bu sefer sınama bilgisayar üzerinden kod çalıştırılarak gerçekleştirilir. Sınama sırasında ortaya çıkan hatalar ve performans sorunları giderilerek program iyileştirilir.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

□ Yazılım geliştirme sürecinde ihtiyacımız olan bilgi alanlarını şöyle tanımlayabiliriz:

Programlama dili: Yazılım geliştireceksek, en azından bir programlama diline hakim olmamız gerekir.

Yazılım geliştirme arabirimi: IDE (Integrated Development Environment) olarak da bilinen yazılım geliştirme arabirimi, kod düzenleyici, arayüz tasarlayıcı, kod derleyici ve yorumlayıcıyı bir arada barındıran ve yazılı geliştiricilere hayatı kolaylaştıran bir araçtır. Eğer yazılım geliştireceksek en hızlı şekilde en iyi kodu yazmamızı sağlayacak bir yazılım geliştirme arabirimi bilgisine sahip olmamız gerekmektedir.

Platform: Yukarıda bahsi geçen iki konuda bilgi sahibiysek, yazılım geliştireceğimiz platformu iyice kavramalı, bu platforma özel programlama bilgilerini edinmeliyiz. Yazılım geliştirme platformları temel olarak masaüstü işletim sistemleri (Windows), internet ve mobil olarak düşünülebilir. Yani VB.NET ile kodlamayı biliyor olabiliriz ancak Web uygulaması geliştiriyorsanız response, request gibi nesneleri de biliyor olmalıyız.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

□ Yazılım geliştirme sürecinde ihtiyacımız olan bilgi alanlarını şöyle tanımlayabiliriz:

Teknoloji: Geliştirdiğimiz yazılımın üzerinde çalıştığı teknolojilere hakim olmamız gerekir. Örneğin; XML dosyaları işleyerek sipariş alacak bir yazılım geliştireceksek, XML teknolojisini bilmeliyiz. Eğer bir mail alma/yollama programı yazacaksak, SMTP protokolünü bilmeliyiz.

En İyi Pratikler (Best Practices): Belirli bir teknolojide bir çözüm üretmek istiyorsak, öncelikle bu iş birileri yapmış mı diye bakmakta fayda vardır. Eğer yaptığımız işi daha önceden yapanlar olduysa, onların bunu nasıl çözdüğünü öğrenerek bu şekilde yapmak en doğrusu olacaktır. İşte bu daha önceki pratiklerin en iyilerini bulup bunlardan faydalanmalıyız.

İş Bilmek (Know-How): Yazılım geliştirileceği alana özel bilgilere know-how denir. Yazılımı geliştireceğimiz işi bilmeden o iş için doğru ve kaliteli program yazmamız mümkün değildir.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

- Algoritma ile oluşturulacak çözümler sözel olarak ifade edilir.
- Örneğin sabah kalktığımızda kahvaltı yapılacağı zaman kahvaltı hazırlama algoritması oluşturulursa:
 - Yataktan kalk
 - Mutfağa git
 - Ekmek al
 - Çayı hazırla
 - Dolaptan kahvaltılıkları çıkar
 - Bardağın bitince çayını doldur
 - Karnın doyunca sofradan kalk
 - Kahvaltılıkları dolaba koy
 - Sofrayı temizle

Adım 1:Yataktan kalk

Adım 2:Mutfağa git

Adım 3:Ekmek al

Adım 4:Çayı hazırla

Adım 5:Dolaptan kahvaltılıkları çıkar

Adım 6:Bardağın bitince çayını doldur

Adım 7:Karnın doyunca sofradan kalk

Adım 8:Kahvaltılıkları dolaba koy

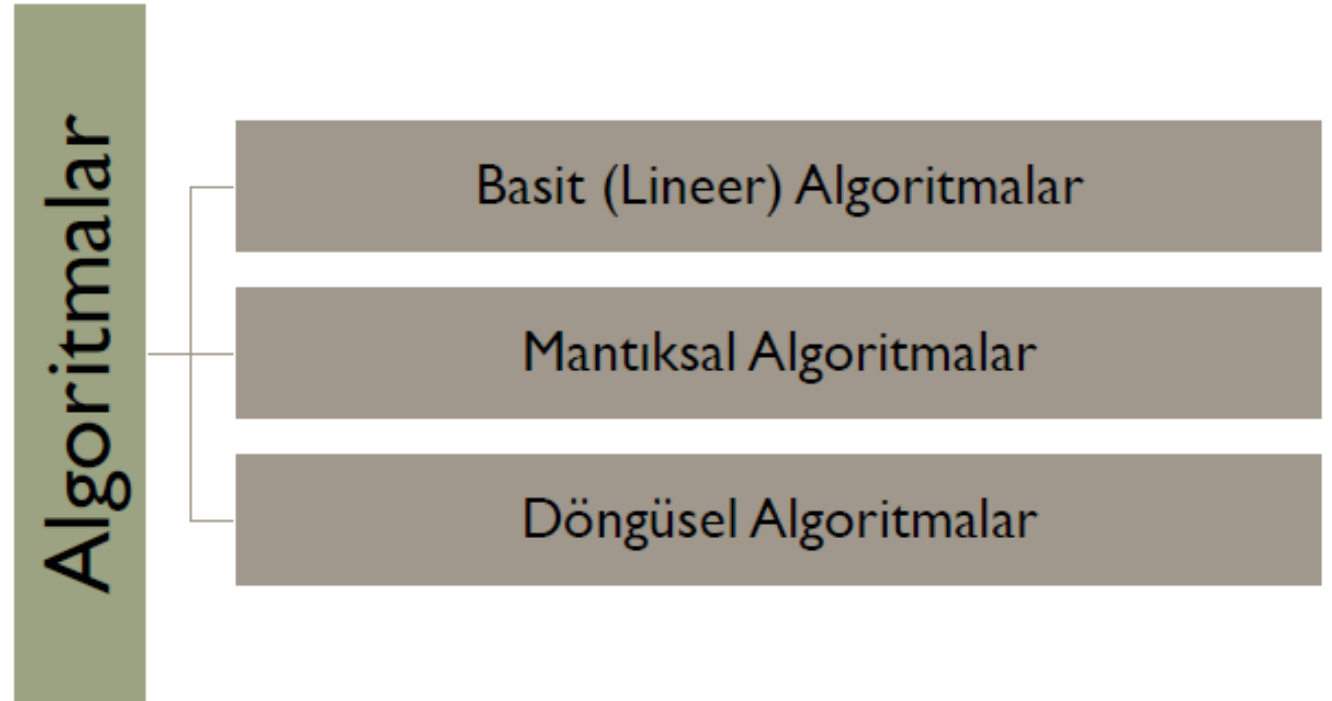
Adım 9:Sofrayı temizle

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmaların Sınıflandırılması

- Algoritmalar karmaşıklık yapılarına göre 3 grupta incelenirler.



ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmaların Sınıflandırılması

- Basit (Lineer Algoritmalar): İçerisinde mantıksal ifadelerin yer almadığı, program akış dallanmalarının olmadığı algoritmalar. Bu algoritmalarda akış düz bir halde baştan sona doğru olacaktır. Çoğunlukla küçük hesaplamaları gerçekleştirmek için kullanılırlar. Bir önceki algoritmaya bakılırsa bir karar yapısının olmadığı görülmektedir.

Örnek:

- Adım 1: Hesaplanacak kilometre uzunluğunuz giriniz; **km**
- Adım 2: Girilen değeri 1000 ile çarpınız; **$m = km * 1000$**
- Adım 3: Hesaplanan değeri ekrana yazdırınız; **m**

Örnek: Dışarıdan girilen üç adet sayısının toplamını, çarpımını ve ortalamasını hesaplayan algoritma;

- Adım 1: Üç adet sayı giriniz; **a, b, c**
- Adım 2: Sayıların toplamını hesaplayınız;
 $toplam = a + b + c$
- Adım 3: Sayıların çarpımlarını hesaplayınız;
 $çarpım = a * b * c$
- Adım 4: Sayıların ortalamasını hesaplayınız;
 $ort = toplam / 3$
- Adım 5: Sayıların toplamını, çarpımını ve ortalamasını ekrana yazdırınız; **toplam, çarpım, ort.**

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmaların Sınıflandırılması

- **Mantıksal Algoritmalar**: Algoritma içerisinde mantıksal karşılaştırmaların bulunduğu yapılardır. Mantıksal karşılaştırmalara göre algoritmanın akışı farklı adımlara geçecektir. Bu şekilde oluşturulan algoritmalara Mantıksal Algoritmalar denir.
- İlk oluşturulan algoritma biraz daha ayrıntılanırsa karar yapılarının ortaya çıktığı görülür.

Örnek:

- Adım 1: Yataktan kalk
- Adım 2: Mutfığa git
- Adım 3: Eğer ekmek yoksa ekmek al
- Adım 4: Çayı hazırla
- Adım 5: Dolaptan kahvaltılıkaları çıkar
- Adım 6: Bardağın bitince çayını doldur
- Adım 7: Karnın doyunca sofradan kalk
- Adım 8: Eğer kahvaltılıkalar bitmişse bulaşık makinesine koy
- Adım 9: Eğer kahvaltılıkalar bitmemişse kahvaltılıkaları dolaba koy
- Adım 10: Sofrayı temizle

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmaların Sınıflandırılması

□ Mantıksal Algoritmalar:

- Girilen üç adet sayı içinden en büyük sayıyı bulan algoritma yazalım.
- Adım 1: Üç adet sayı giriniz; a, b, c
- Adım 2: En büyük sayı a olsun; $eb = a$
- Adım 3: Eğer b en büyükten büyük ($b > eb$) ise en büyük b olsun; $eb = b$
- Adım 4: Eğer c en büyükten büyük ($c > eb$) ise en büyük c olsun; $eb = c$
- Adım 5: En büyük sayıyı ekrana yazdır; eb

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmaların Sınıflandırılması

□ Mantıksal Algoritmalar:

Sayının pozitif, negatif veya sıfır olduğunu bulan algoritmayı yazınız.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmaların Sınıflandırılması

□ Mantıksal Algoritmalar:

Sayının pozitif, negatif veya sıfır olduğunu bulan algoritma yazalım.

- Adım 1: Sayıyı giriniz; **a**
- Adım 2: Eğer a sayısı sıfırdan büyük ise ekrana 'pozitif' yaz ve adım 5'e git
- Adım 3: Eğer a sayısı sıfırdan küçük ise ekrana 'negatif' yaz ve adım 5'e git
- Adım 4: Eğer a sayısı sıfıra eşit ise ekrana 'sıfır' yaz ve adım 5'e git
- Adım 5: Program bitti.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmaların Sınıflandırılması

- **Döngüsel Algoritmalar**: Program için geliştirilen algorithmada bir işlem birden fazla tekrar ediyorsa döngülü algoritma yapısı kullanılır.
 - Döngüsel algoritmalarda mantıksal karşılaştırma yapısı özel olarak kullanılır.
 - Eğer algoritma içerisinde kullanılan mantıksal karşılaştırma işlemi sonucunda programın akışı karşılaştırma yapılan yerden daha ileriki bir adıma değil de daha önceki adıma gidiyorsa bu şekilde oluşturulmuş algoritmalara döngüsel algoritma denir.
 - Yani döngüsel algoritmalarda mantıksal karşılaştırma sonucunda program daha önceki adımlara gider.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmaların

Sınıflandırılması

□ Döngüsel Algoritmalar

Örnek Pasta Yapım Süreci

1. Pastanın yapımı için gerekli malzemeleri hazırla
2. Yağı bir kaba koy
3. Şekerini aynı kaba yağın üzerine koy
4. Yağ ve şekerini çırp
5. Karışımın üzerine yumurtayı kır
6. Tekrar çırp
7. Kıvama geldi mi diye kontrol et
8.
 - a. Kıvamlı ise 9. adıma devam et
 - b. Değilse 6. adıma dön.
9. Karışımına un koy
10. Karışımına vanilya, kabartma tozu vb. koy
11. Karışımı Kıvama gelinceye kadar çırp
12. Pastayı Kek kalıbına koy
13. Yeteri kadar ısınan fırına pastayı koy
14. Pişimi diye kontrol et
15.
 - a. Pişmiş ise 16. adıma devam et
 - b. Değilse 14. adıma dön
16. Kek'i fırından çıkart
17. Fırını kapat
18. Kek'in ı kapat
19. Kek'in soğumasını bekle
20. Kek'i servis edebilirsin.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmaların Sınıflandırılması

□ Döngüsel Algoritmalar: Bir sayının faktöriyelini bulan algoritma yazalım.

Adım 1: faktöriyeli hesaplanacak sayıyı giriniz; n

Adım 2: faktöriyel değerini 1 yap; $f=1$

Adım 3: indeks değerini 1 yap; $i=1$

Adım 4: faktöriyel değeri ile indeksi çarp ve hesaplanan değeri faktöriyele yaz; $f=f \times i$

Adım 5: indeks değerini bir artır; $i=i+1$

Adım 6: eğer indeks değeri hesaplanacak sayıdan küçük veya eşit ise Adım 4' git

Adım 7: faktöriyel değerini ekrana yaz; f

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmaların Sınıflandırılması

- Döngüsel Algoritmalar: Girilen iki sayının en büyük ortak bölenini (EBOB) bulan algoritma yazalım.
- • Adım 1: Sayıları giriniz; a, b
- • Adım 2: Eğer $a > b$ ise b 'yi a 'dan çıkar ve tekrar a 'ya yaz ($a = a - b$) ve Adım 2'ye git
- • Adım 3: Eğer $b > a$ ise a 'yı b 'den çıkar ve tekrar b 'ye yaz ($b = b - a$) ve Adım 2'ye git
- • Adım 4: Eğer $a = b$ ise adım 5'e git
- • Adım 5: En büyük bölen a değerini ekrana yaz; a

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Geliştirmek

Bir problem çözmek üzere geliştirilen algoritma üç şekilde yazılabilir:

- **Satır algoritma**: Problemin çözüm adımları düz metin olarak açık cümlelerle yazılır.
- **Akış diyagramı (flow-chart)**: Problemin çözüm adımları geometrik şekillerle gösterilir.
- **Sözde kod (pseudo-code)**: Problemin çözüm adımları komut benzeri anlaşılır metinlerle veya kısaltmalarla ifade edilir.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Geliştirmek

Örnek: Klavyeden girilen iki sayıyı toplayıp ekranda gösteren programın algoritmasının üç değişik yöntemle gösterilmesi:

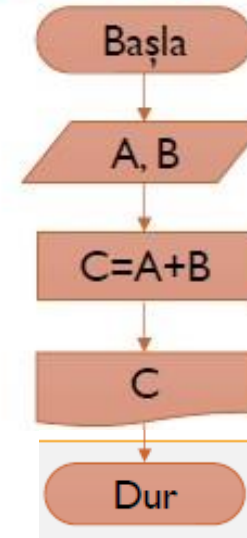
Satır algoritma ile gösterilmesi:

1. Başla
2. Birinci sayıyı (A) klavyeden oku
3. İkinci sayıyı (B) klavyeden oku
4. Girilen sayıları toplayarak sonucu oluştur ($C=A+B$)
5. Sonucu (C) ekrana yazdır
6. Dur

Sözde kod ile gösterilmesi:

1. Başla
2. A oku
3. B oku
4. $C=A+B$
5. C yaz
6. Dur

Akış diyagramı ile gösterilmesi:



ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Geliştirmek

Algoritmaları geliştirirken değişken, sabit, atama, döngü, karar yapısı, alt yordam gibi bir takım öğeler kullanılır.

- **Veri: (data):** Bilgisayarlarda işlenen tüm bilgiler veri olarak adlandırılırlar. Veriler temel olarak sayısal ve alfasayısal olmak üzere ikiye ayrılırlar (Seckin, s.55).
- **Tanımlayıcı: (identifier):** Değişken, sabit, alt yordam, alan gibi programlama birimlerine yazılımcı tarafından verilen isimlerdir.
- **Değişken (variable):** Programın akışı içinde farklı değerleri tutmak üzere ayrılmış bellek bölümüdür. Örneğin $C=A+B$ gibi bir ifadede A, B ve C tanımlayıcıları birer değişkendir.
- **Sabit (constant):** Program her çalıştığında ve programın içinde herhangi bir anda hep aynı değeri döndüren tanımlayıcılara sabit denir. $PI=3.14$ gibi bir ifadeden sonra PI sabiti programın her yerinde 3.14 değerini ifade eder.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Geliştirmek

İsimlendirme Kuralları:

- İngiliz alfabesindeki A-Z veya a-z arası 26 harf kullanılabilir.
- 0-9 arası rakamlar kullanılabilir.
- Simgelerden sadece alt çizgi (_) kullanılabilir.
- Harf veya alt çizgi ile başlayabilir, ancak rakamla başlayamaz veya sadece rakamlardan oluşamaz.
- İlgili programlama dilinin komutu veya saklı/anahtar kelimesi olamaz.

En çok kullanılan standart tanımlayıcı gösterimleri:

- - Pascal case: Kelimelerin ilk harfleri büyük. Örn: AdSoyad, OgrenciNo
- - Camel case: Birinci kelimenin ilk harfi küçük, diğer kelimelerin ilk harfleri büyük. Örn: adSoyad, ogrenciNo

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Geliştirmek

- **Gömülü değer (literal):** Kod içine yazılmış olan metinsel, sayısal ya da diğer veri tiplerindeki sabit değerlere gömülü değer denir. $\pi=3.14$ ifadesindeki 3.14 değeri bir gömülü değerdir. Aynı şekilde mesaj="merhaba" ifadesindeki mesaj bir değişken veya sabit, "merhaba" ise gömülü değerdir.
- **Aritmetik işlemler:** Programlamadaki en temel işlemlerdir. Aritmetik işlemlerde kullanılan operatörler aşağıdaki gibidir.

İşleç	Adı	Örnek
+	Toplama	$C=A+B$
-	Çıkartma	$D=E-F$
*	Çarpma	$X=Y*Z$
/	Bölme	$T=P/S$
=	Eşittir	$A=B+C$

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Geliştirmek

- **Mantıksal İşlemler:** Bir programın akışı içerisinde belirli bir koşula bağlı olarak akışın hangi yönde ilerleyeceğine karar vermede mantıksal işlemler kullanılır. Bu ifadelerin sonucunda doğru (true) ya da yanlış (false) değerleri elde edilir.
- Mantıksal işlemlerde kullanılan işleçler aşağıdaki gibidir.

İşleç	Adı	Örnek
>	Büyük	$A > B$
<	Küçük	$A < B$
==	Eşit	$A == B$
<> (veya !=)	Farklı	$A <> B$ veya $(A != B)$
>=	Büyük Eşit	$A >= B$
<=	Küçük Eşit	$A <= B$

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Geliştirmek

□ **Örnek:** Öğrencinin numarasını, vize ve final notunu aldıktan sonra ortalamasını hesaplayıp numarasını ve not ortalamasını yazdıran program.

1. Başla
2. Öğrencinin numarasını (No) al
3. Öğrencinin adını ve soyadını (AdSoyad) al
4. Öğrencinin vize notunu (VizeNot) al
5. Öğrencinin final notunu (FinalNot) al
6. $\text{Ort} = 0.3 * \text{VizeNot} + 0.7 * \text{FinalNot}$
7. Numara (No) ve ortalamayı (Ort) yaz
8. Dur

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Geliştirmek

- **Sayaç Kullanımı:** Programlarda bazı işlemlerin belirli sayıda yapılması veya işlenen değerlerin sayılması gerekebilir. Örneğin klavyeden girilen bir cümlede kaç tane sesli harf olduğunu bulan programda, cümlenin her harfi sırasıyla çağrılır ve sesli harfler kümesine ait olup olmadığı araştırılır. Eğer sıradaki çağrılan harf bu kümeye ait ise, bunları sayacak olan değişkenin değeri artırılır.

`sayac=sayac+1`

- Burada sağdaki ifadede değişkenin eski değerinin üzerine 1 eklenerek bulunan sonuç yine değişkenin kendisine yeni değer olarak atanmaktadır.

Sayaç yapısı: `sayac=sayac+adım`

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Geliştirmek

- **Döngü Kullanımı:** Programlardaki belirli işlem bloklarını (kod parçalarını); aynı veya farklı değerlerle, verilen sayıda gerçekleştiren çevrim yapılarına döngü denir.
- Örneğin 1 ile 1000 arasındaki tek sayıları ekrana yazdıracak programda, 1 ile 1000 arasında ikişer ikişer artan bir döngü açılır ve döngü değişkeni ardışık olarak yazdırılır.

Döngü Oluşturma Kuralları:

- Döngü değişkenine başlangıç değeri verilir.
- Döngünün artma veya azaltma değeri belirlenir.
- Döngünün bitiş değeri belirlenir.
- Eğer döngü karar ifadeleri ile oluşturuluyorsa, karar işleminden önce döngü değişkenine başlangıç değeri verilmiş olmalı ve döngü içinde

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmanın Hazırlanması

Algoritmadaki adımlar programın sonlu sayıda işlem yapmasını sağlamalıdır.

- Adımlar sıralı ve mantıklı olmalıdır.
- Tekrar eden işlemlerde programın hızı ve etkinliği açısından kaçınılmalıdır.
- Farklı değerlerle gerçekleştirilen aynı işlemleri tekrar tekrar yazmak yerine bunları alt program/fonksiyon olarak oluşturmak daha uygundur.
- Gereksiz işlemlerden kaçınılmalıdır.
- Etkisiz işlemlerden kaçınılmalıdır.
- Bellek verimliliği açısından fazladan veya gereksiz tanımlamalardan kaçınılmalıdır.
- Her algoritmanın bir çıktısı/sonucu olmalıdır.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritma Yazma Kuralları

- Algoritmadaki tüm satırlar 1'den başlayarak numaralandırılmalıdır.
- Bütün algoritmalarda birinci satır "1. Başla" şeklindedir.
- Bütün algoritmalar "Dur" ile biter.
- Algoritmalarda kullanılan "Git" işlem akışı yönlendirme komutu satır numarasıyla birlikte kullanılmalıdır.
- Algoritmalarda kullanılabilecek alt program veya fonksiyonlar, tanımlayıcı isimleri ve parametreleriyle birlikte verilmelidir.
- Algoritmalarındaki adımlar, sınırlı sayıda, açık, net ve kesin olmalıdır.
- Algoritmalarındaki ifadeler anlaşılır ve mümkün olduğunca sade (az ve öz) olmalıdır.
- Algoritmalarındaki ifadeler, herhangi bir programlama diline, donanıma, işletim sistemine vb. bağlı olmamalıdır.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

AKIŞ DİYAGRAMLARI

Bir problemin çözümüne yönelik geliştirilen algoritmanın görsel sembollerle ifade edilmiş şekline akış diyagramı denir. Akış diyagramları program yazarken sürecin izlenmesi ve başkalarına algoritmaları açıklarken yardımcı olur. Akış diyagramları sözde kodların görselleştirilmiş halleri olarak düşünülebilir.

Akış diyagramlarında;

Başla,

Bitir,

Girdi,

Çıktı,

Karar ve

İşlem





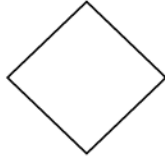


sembolleri kullanılır.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

AKIŞ DİYAGRAMLARI

Akış diyagramı oluşturulurken yaygın olarak kullanılan semboller:

Sembol	Açıklaması
	Akış yönü: Diyagramda şekiller arasındaki akışı gösterirler. Ok yönü akış yönünü ifade eder.
	Başla/Bitir: Programın başladığı ve bittiği konumu gösterir. Her programda başla ve dur diyagramları mutlaka olmalıdır.
	Girdi/Çıktı: Çevre birimleri ile yapılan bilgi alışverişini simgeler. Yapılacak işlem şekil içerisine yazılır.
	İşlem/Atama: Değişkenlere değer atamaları ve matematiksel veya dizgisel işlemlerin yapıldığı aşamalarda kullanılır. İşlemler bu şekil içerisine kısa olarak yazılır.
	Karşılaştırma ve karar: Karşılaştırma işlemi ve sonuçta varılan karar durumuna göre akış yönünü belirleyen işlemlerde kullanılır. Kıyaslama ifadesi şekil içine yazılır, karar E (evet) veya H (hayır) simgesi ile belirtilen bir uçtan çıkan akış ile başka bir diyagrama gider.
	Alt süreç: Bir işin tamamlanması için alt süreçler ve uygulamalar varsa bu süreçleri simgeler. Sürecin kendisi değil ancak tanımı şekil içine yazılır.
	Belge: Basılı belge veya okunabilir ekran çıktısının ismi şekil içerisine yazılır. Çoklu belge çıktısı için birden fazla şekil kullanılır.

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

AKIŞ DİYAGRAMLARI



Başla/Dur: Algoritmayı başlatmak/sonlandırmak için kullanılır.

Başla

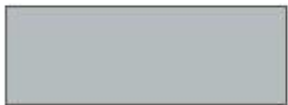
Dur



Veri Girişi: Bilgisayara klavyeden veri girişini temsil eden şekildir.

OgrNo

A, B, C



İşlem: Programın çalışması sırasında yapılacak işlemleri ifade etmek için kullanılan şekildir.

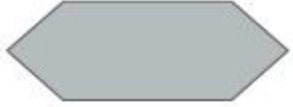
$c=a^2+2ab$

$A=PI*r^2$

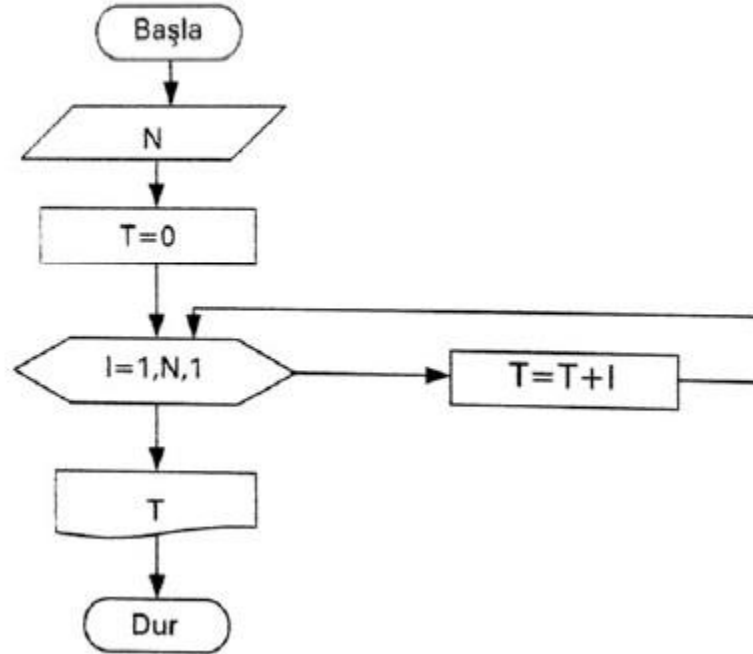
ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

AKIŞ DİYAGRAMLARI



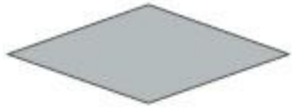
Döngü: Tekrarlamalı işlemler döngü olarak adlandırılır. Şeklin içine döngünün başlangıç, bitiş ve artış (adım) değerleri yazılır.



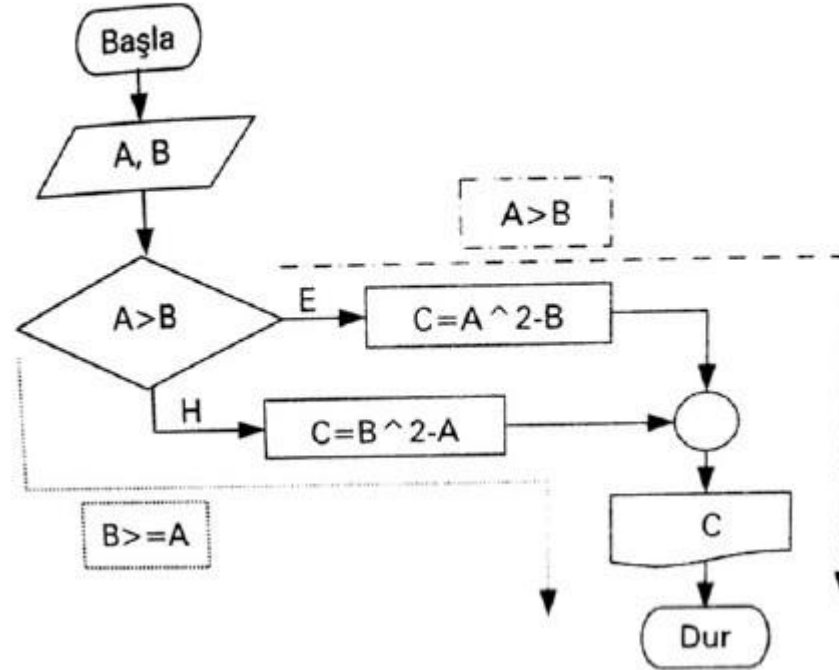
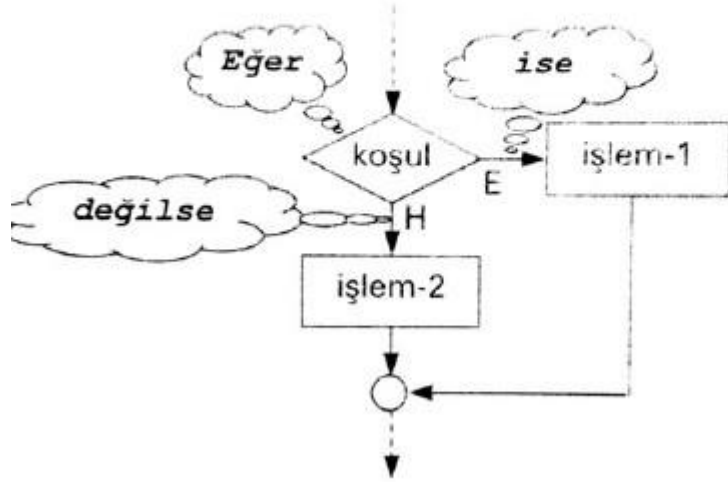
ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

AKIŞ DİYAGRAMLARI



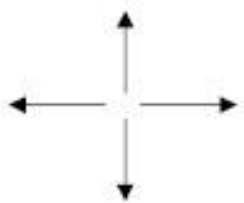
Karar: Karar verme/karşılaştırma işlemlerini temsil eden şekildir. Koşul veya mantıksal operatörlerle bağlı koşullar, şeklin içine yazılır.



ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

AKIŞ DİYAGRAMLARI



İşlem Akış Yönleri: İşlem akışının hangi yönde olduğunu gösteren oklardır.

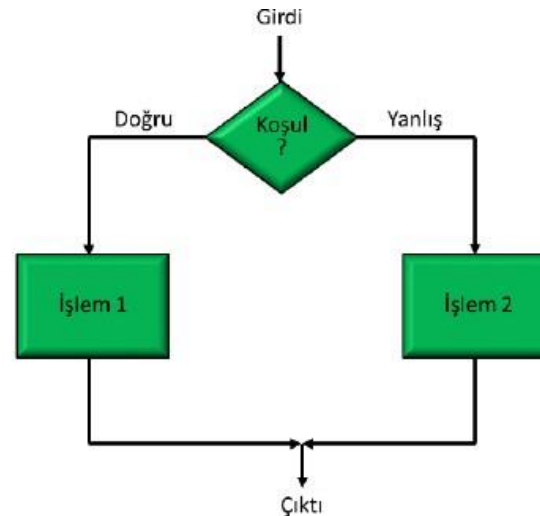
ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

AKIŞ DİYAGRAMLARI

- Algoritma tasarımında işlem adımları kadar işlem sırası da önemli olup, algoritmaların şekilsel gösterimi olan akış diyagramlarında ok ile gösterilen akış yönüne dikkat edilmelidir. Akış yönüne göre kavşakların olduğu yerler karar verme noktalarıdır. Algoritmalarda sıklıkla karşılaşılan karar yapılarının sözde kod ve akış diyagramlarının anlaşılması algoritma tasarımı açısından çok önemlidir.
- Karşılaştırma sonucunun doğru olduğu durumda işlem 1, yanlış olduğu durumda işlem 2'yi yapacak algoritmanın sözde kodu ve akış diyagramı aşağıda verilmiştir.

```
SET Girdi  
IF Koşul THEN  
    COMPUTE işlem 1  
ELSE  
    COMPUTE işlem 2  
ENDIF
```



ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

AKIŞ DİYAGRAMLARI

İki veya çok alternatifli koşul yapıları

Bazı durumlarda basit koşul yapıları yetse de bazen akışın ikiye (veya daha fazla) ayrılıp birinden devam etmesi gerekebilir. Yani, basit koşul yapılarının yetmediği yerde iki veya daha çok alternatifli koşul yapıları kullanılır.

Örnek: Kullanıcıdan bir sayı alarak, girilen sayının pozitif, negative veya sıfır olduğunu ekrana yazdıran programın algoritmasını tasarlayın (Kodlab s.34).

1. Başla
2. Yaz “Bir Sayı Girin”
3. Oku Sayı
4. Eğer Sayı >0 ise Yaz “Girdiğiniz Sayı Pozitiftir”
5. Eğer Sayı <0 ise Yaz “Girdiğiniz Sayı Negatiftir”
6. Eğer Sayı $=0$ ise Yaz “Girdiğiniz Sayı Sıfırdır”
7. Dur

Bu algorithmada 4, 5, ve 6'ncı satırlardaki koşullar sırayla sınanır ve komutlardan sadece birisi çalışmış olur.

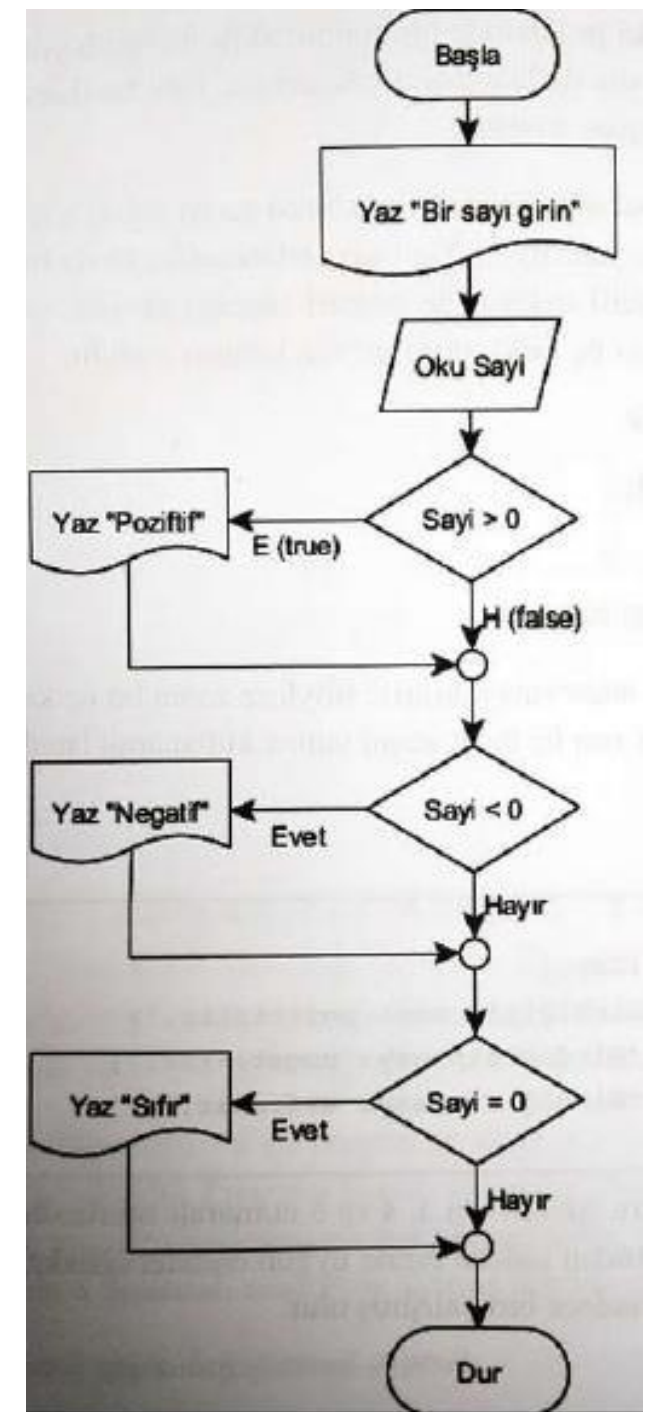
ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

AKIŞ DİYAGRAMLARI

İki veya çok alternatifli koşul yapıları

1. Başla
2. Yaz "Bir Sayı Girin"
3. Oku Sayı
4. Eğer $Sayı > 0$ ise Yaz "Girdiğiniz Sayı Pozitifdir"
5. Eğer $Sayı < 0$ ise Yaz "Girdiğiniz Sayı Negatiftir"
6. Eğer $Sayı = 0$ ise Yaz "Girdiğiniz Sayı Sıfırdır"
7. Dur



ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmalar Arasında Dönüşüm

Satır Algoritmalarından Sözde Kod Oluşturmak

Bir satır algoritmayı sözde koda dönüştürürken aşağıdaki adımları izleriz:

- - Girdi ve çıktılar (değişkenler) belirlenir
- - Sıralı adımlar, karar yapıları, tekrarlı yapılar ve işlemler belirlenir
- - Yapı, işlem ve adımlar uygun şekilde birleştirilir.

```
1. Başla  
2. Yaz 1  
3. Yaz 2  
4. Yaz 3  
5. Dur
```



```
DISPLAY 1  
DISPLAY 2  
DISPLAY 3
```

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmalar Arasında Dönüşüm

Satır Algoritmalarından Sözde Kod Oluşturmak

Örnek: İki sayıyı alıp, bunları toplayarak toplamı ekrana yazdıran algoritmanın satır kodu ve sözde kodu:

```
1. Başla  
2. Oku (A,B)  
3. C=A+B  
4. Yaz C  
5. Dur
```



```
GET A  
GET B  
C=A+B  
DISPLAY C
```

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmalar Arasında Dönüşüm

Satır Algoritmalarından Sözde Kod Oluşturmak

```
1. Başla
2. Yaz "Yaşınızı Giriniz"
3. Oku (Yas)
4. Eğer
  4.1. Yas >= 18 ise Yaz "Uygulamayı İndirebilirsiniz"
  4.2. Değilse ise Yaz "Uygulamayı İndiremezsiniz"
5. Dur
```



```
DISPLAY "Yaşınızı Giriniz"
GET Yas
IF Yas >= 18 THEN
  DISPLAY "Uygulamayı İndirebilirsiniz"
ELSE
  DISPLAY "Uygulamayı İndiremezsiniz"
ENDIF
```

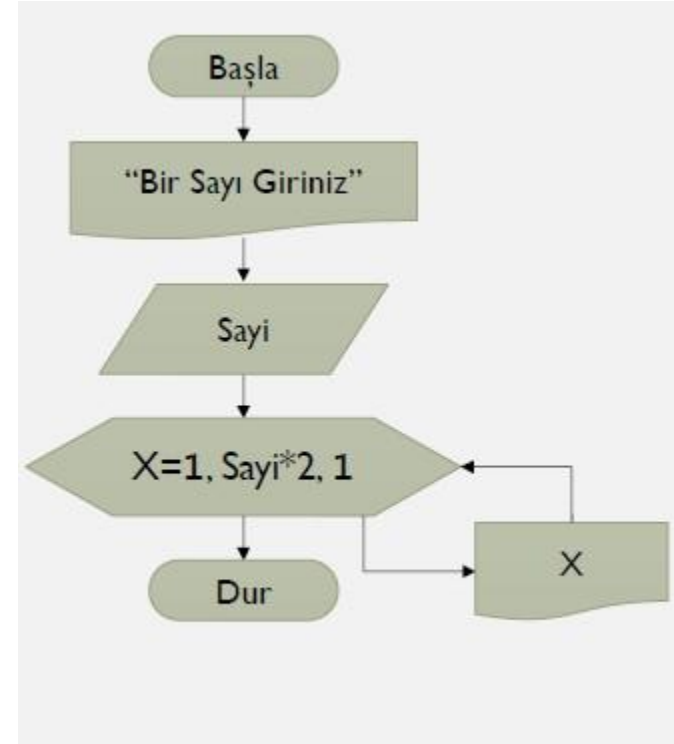
ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmalar Arasında Dönüşüm

Satır Algoritmalarından Akış Diyagramı Oluşturmak

1. Başla
2. Yaz “Bir Sayı Giriniz”
3. Oku Sayı
4. Döngü (X=1 TO Sayı*2 STEP 1)
5. Yaz X
6. DöngüSonu
7. Dur

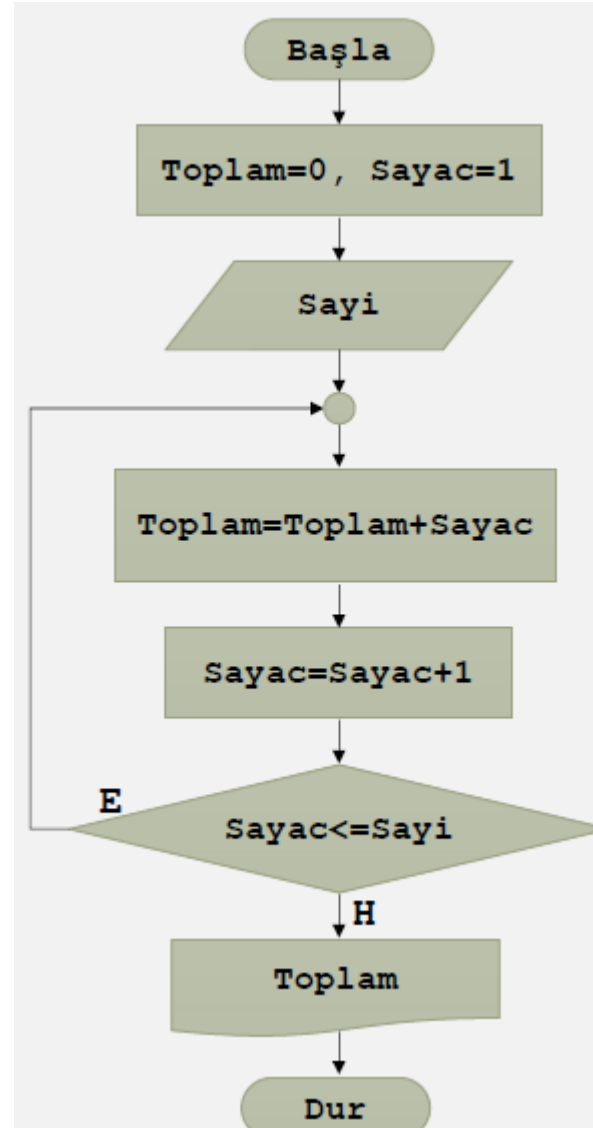


ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Algoritmalar Arasında Dönüşüm

Akış Diyagramlarından Sözde Kod Oluşturmak



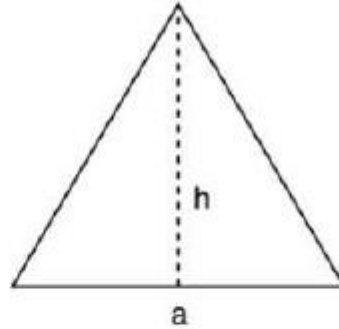
```
Toplam=0
GET Sayi
FOR Sayac=1 TO Sayi STEP 1
    Toplam=Toplam+Sayac
ENDFOR
DISPLAY Toplam
```

ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

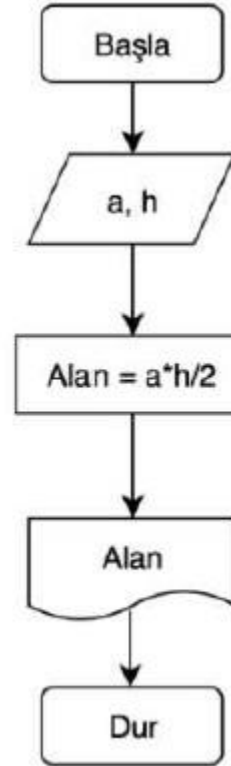
Genel Örnek:

Klavyeden bir kenar uzunluğu ve o kenara ait yüksekliği girilen üçgenin alanını hesaplayan programın satır kodunu ve akış diyagramını geliştiriniz.



$$\text{Alan} = (a \cdot h) / 2$$

1. Başla
2. Kenar uzunluğunu (a) gir
3. Yüksekliği (h) gir
4. $\text{Alan} = a \cdot h / 2$
5. Yaz Alan
6. Dur

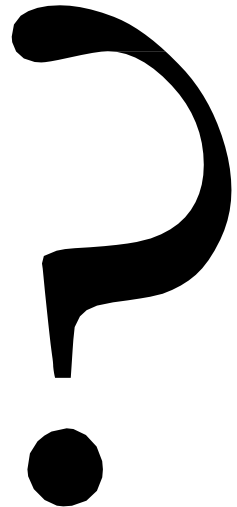


ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Genel Örnek:

- Girilen iki sayının ortalamasını alan programın algoritmasını geliştirerek akış diyagramını çizersiniz.

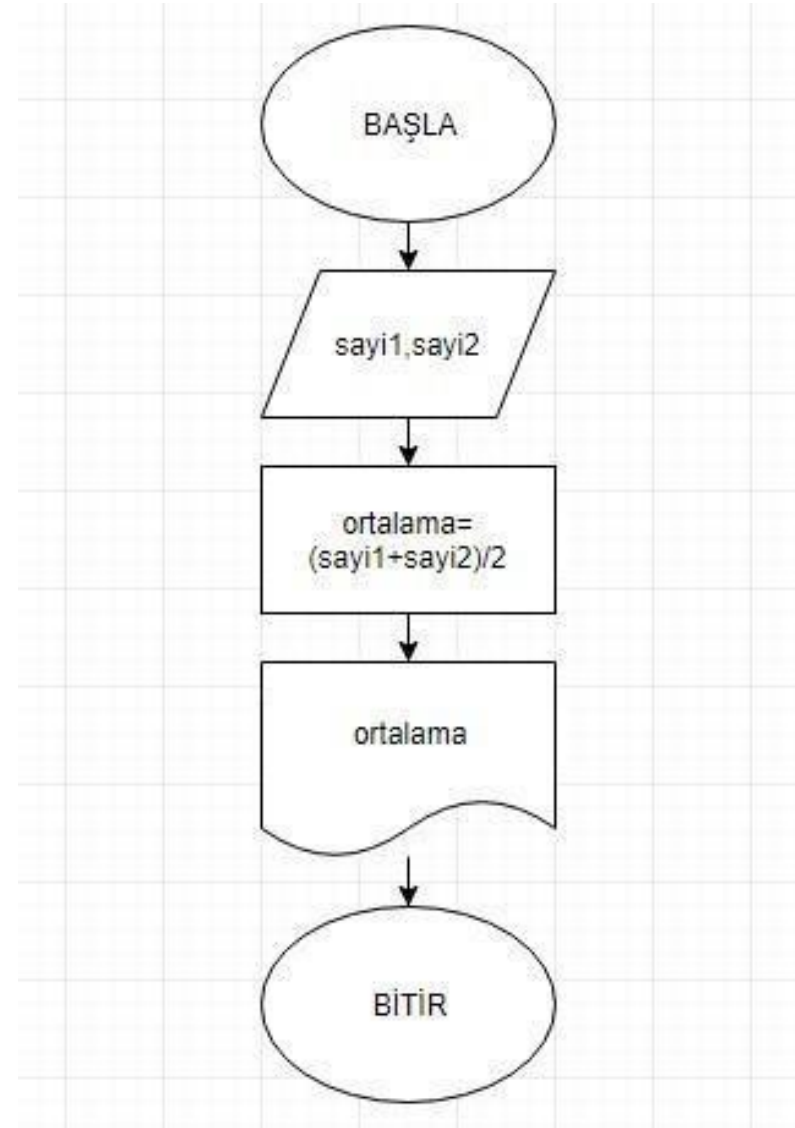


ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Genel Örnek:

- Girilen iki sayının ortalamasını alan programın algoritmasını geliştirerek akış diyagramını çiziniz.

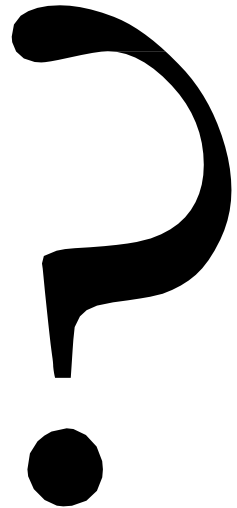


ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Genel Örnek:

Klavyeden dik kenarlarının uzunluğu verilen bir üçgende, hipotenüsün uzunluğunu bulan programı satır kod ve akış diyagramı olarak ifade ediniz.



ALGORİTMA VE AKIŞ ŞEMALARI

ALGORİTMA

Genel Örnek:

Klavyeden dik kenarlarının uzunluğu verilen bir üçgende, hipotenüsün uzunluğunu bulan programı satır kod ve akış diyagramı olarak ifade ediniz.

