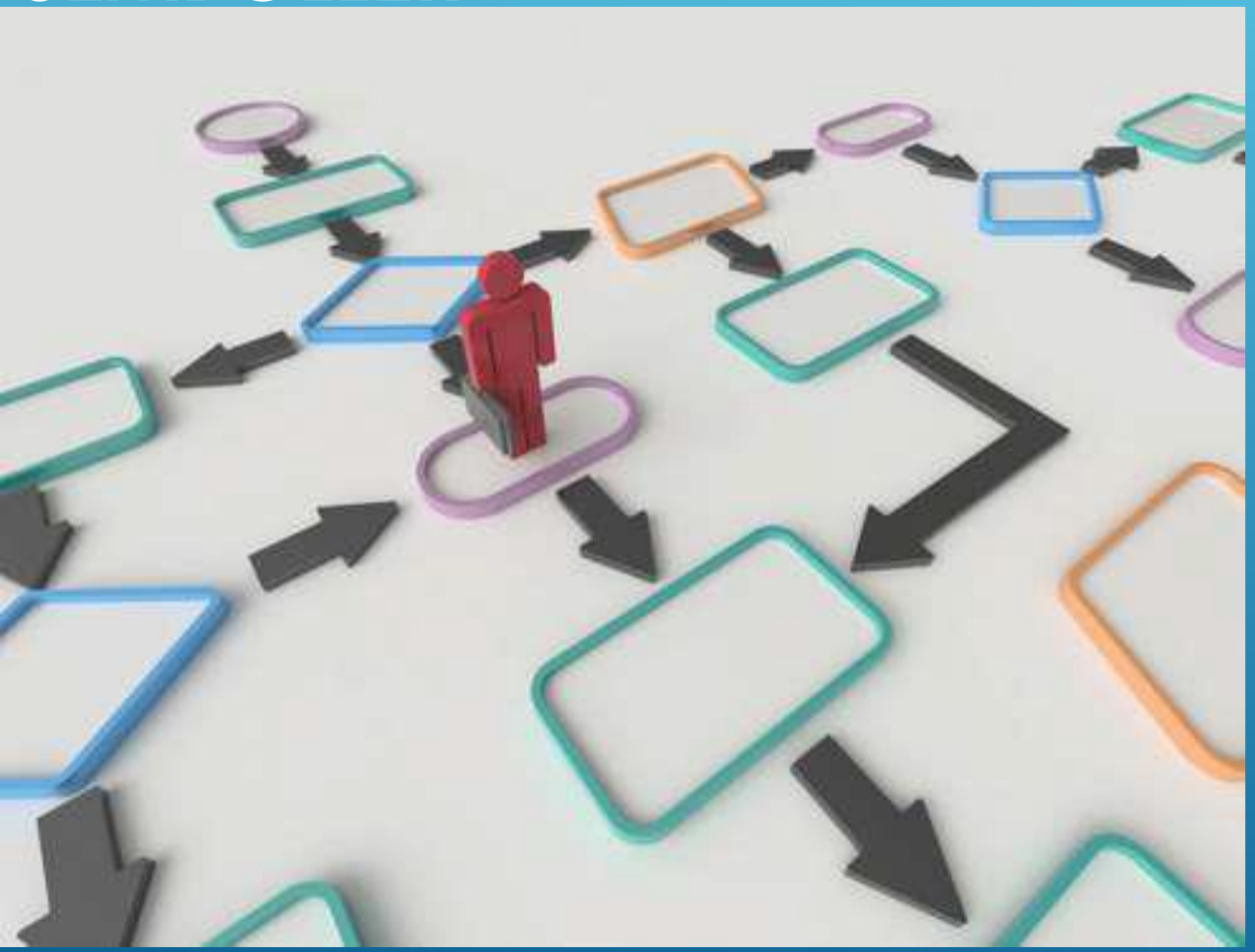
ALGORITMALAR

Bu çizelgeleri geliştirdikten sonraki adım, yapılacak işlemleri bilgisayarın anladığı dilde yazabilmektir. Bu yönergeler "algoritma" olarak adlandırılır. "Sözde kod" algoritmaya çok benzer bir dildir ve bazen algoritma yerine kullanılabilir. Algoritmayı oluşturmak, bilgisayarda problem çözme sürecinin en zor bölümüdür. Modüller etkileşim çizelgesinden ve süreç GSÇ çizelgesinden alınır. Algoritmadaki işlem sayısı, programcının problemi çözme yoluna bağlıdır.

AKIŞ ŞEMALARI

Problem çözme surecimiz, bilgisayarın iletişim kurma yöntemi ile şekillenir. Algoritma, bilgisayara hangi işlemi hangi sırada yapması gerektiğini söyleyen yönergeler bütünüdür. Akış şeması ise algoritmanın görsel gösterimidir. Programcı, oluşturulan algoritmadan grafiksel gösterimler oluşturur. Akış şeması, program geliştirmeye başlamadan önceki son adımdır. Akış şemasında hatalar rahatlıkla görülüp düzeltilebilir. Akış şemalarını oluşturmak için kullanılan evrensel simgeler ve bu her bir simgenin anlamı vardır.

ALGORITMA YÖNERGELERI VE AKIŞ ŞEMASı SEMBOLLERI



Simge	İşlev
	Başla/Bitir
	Giriş
	Atama/İşlem
	Denetim (Karar)
	Çıkış
	Döngü
	Akış Yönü
	Bağlaç
	Önceden Tanımlı İşlem/Fonksiyon

Akış şeması, bir problem çözümünün başlangıcından bitişine kadar olan sureci gösterir. Akış şeması içerisindeki her bir simge, algoritmadaki bir işlemi ifade eder. Genellikle işlemler tek yönlü olmasına rağmen karar kutularından iki farklı ok çıkar. Bir karar simgesinden çıkan ok, bazı işlemlerin tekrarlanmasını sağlayabilir; böylece bir "döngü" oluşur.

AKIŞ ŞEMALARıNı OLUŞTURURKEN DIKKAT EDILMESI GEREKEN NOKTALAR

1. Yönergeler, simgelerin içine yazılmalıdır.

2. Hatırlatıcı bilgiler simgenin yanına yazılabilir. Böylece akış şeması ek açıklamalı bir şemaya dönüşür.

3. Bir akış şeması her zaman sayfanın başından başlar ve sonuna doğru gider. Eğer bir sayfaya sığmazsa bir ya da daha fazla bağlantı simgesi kullanılarak diğer sayfaya geçilebilir.

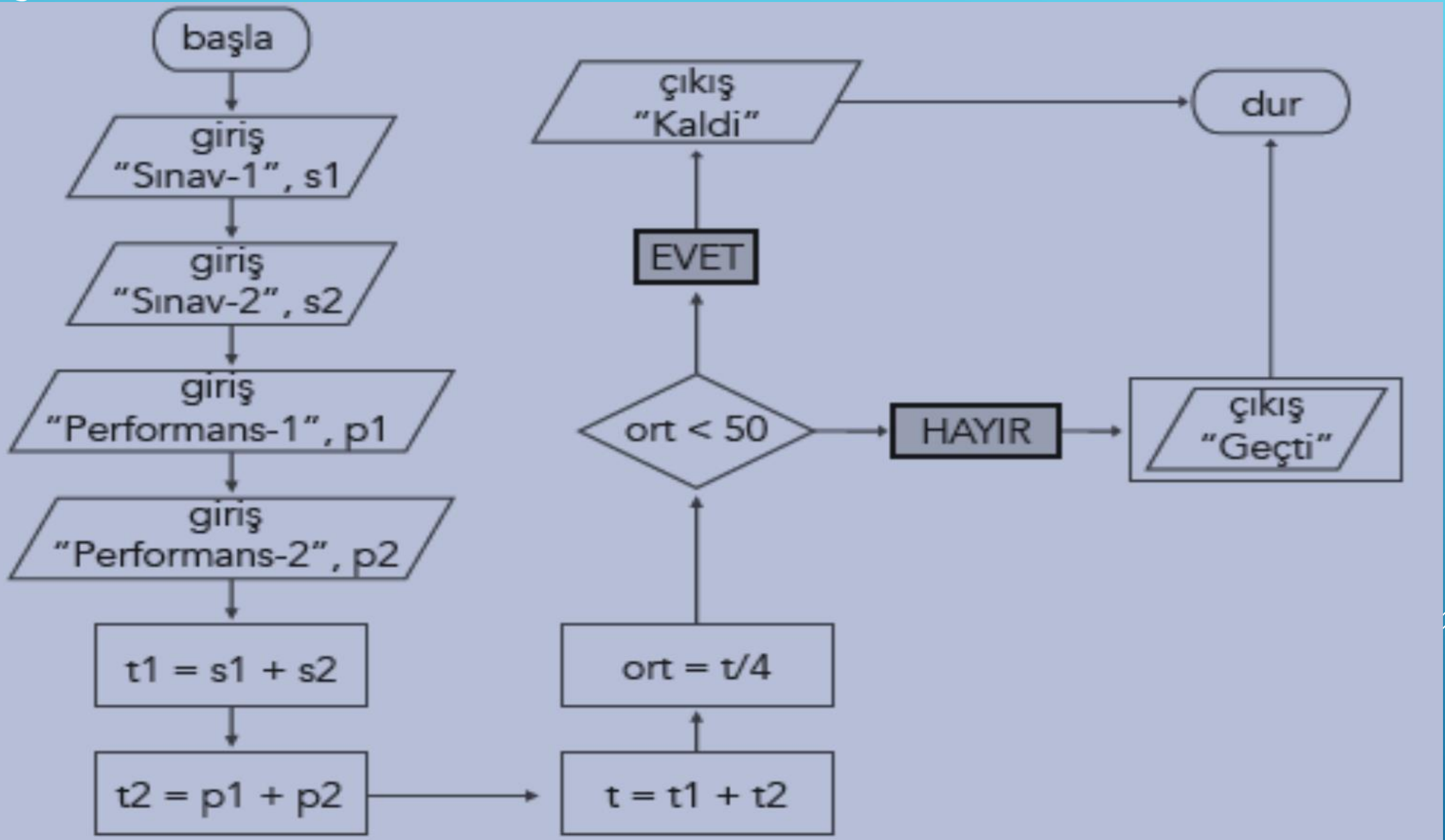
4. Akış şemasını çizmek için uygun yazılımlar kullanılırsa daha standart bir görünüm elde edilir.

5. Simgeler, içeriğindeki yazının rahatça okunabileceği kadar büyük yapılmalıdır.

HARICÎ VE DÂHILÎ DOKÜMANTASYON

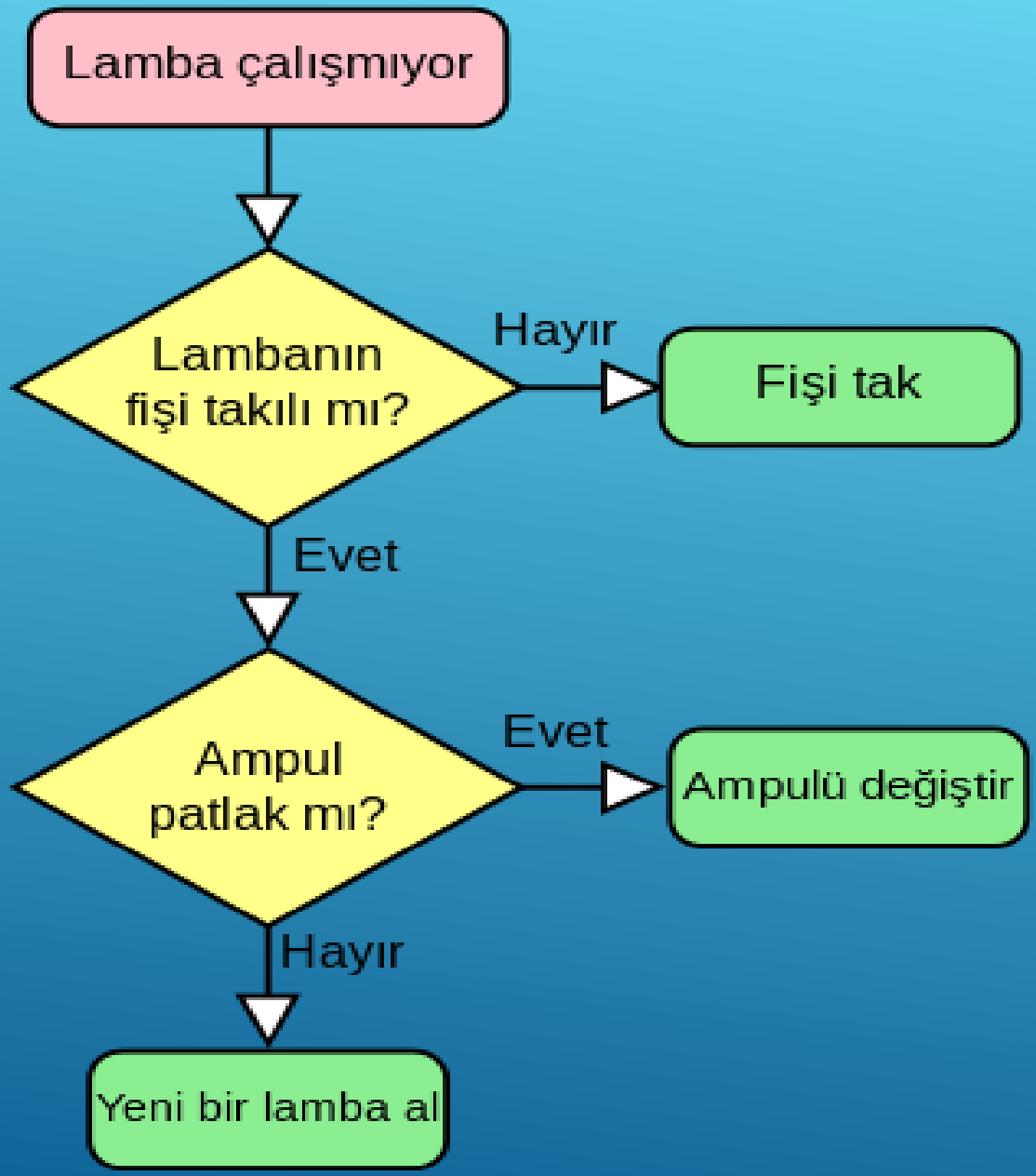
İyi programcılar, kodları başkaları tarafından rahatça anlaşılabilsin diye satırlar arasına açıklamalar yazarlar. Bu açıklamalar, diğer programcılar acısından büyük önem taşır çünkü kod üzerinde değişiklik yapılabilmesi için her bir satırın ya da fonksiyonun işlevinin anlaşılması gerekir. Bu şekilde, yazılıma ait "***dahili dokümantasyon***" oluşturulmuş olunur. Kod satırları haricinde yazılımın kullanımına ve teknik gereksinimlere ait bilgilerden oluşan "***harici dokümantasyon***" hazırlanır. Bu bilgiler, diğer kullanıcılar tarafından ortaya çıkan problemleri çözmek için kullanılır.

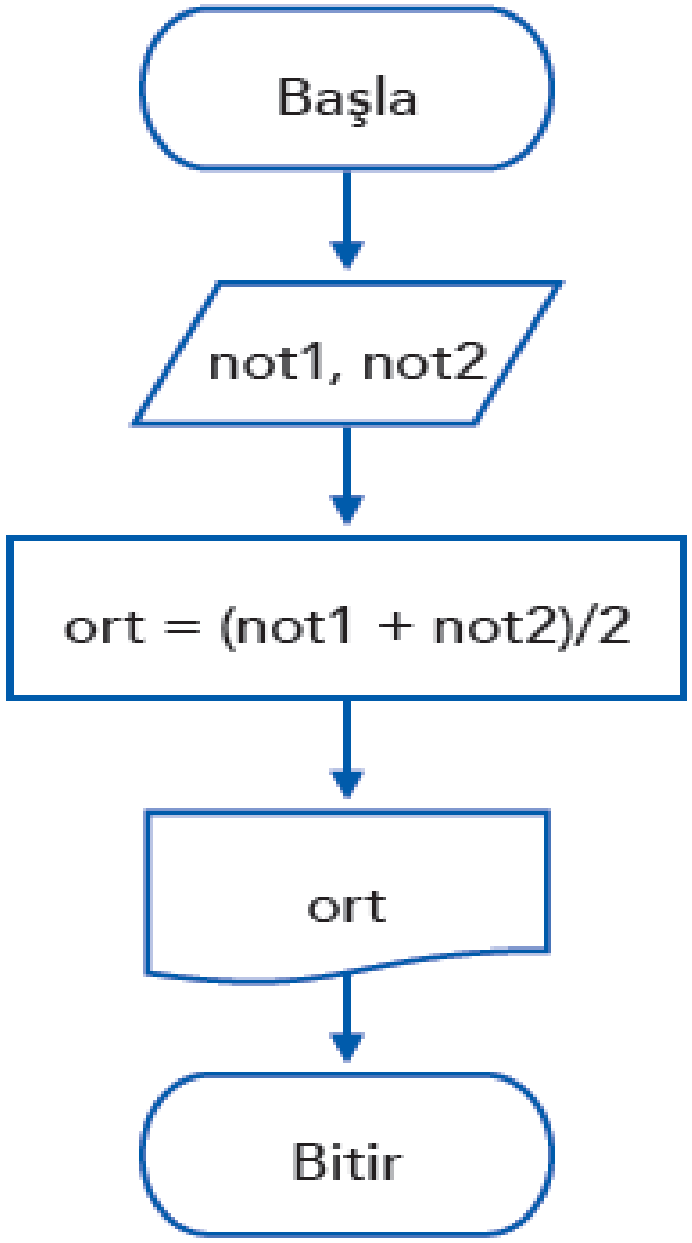
ÇÖZÜMÜN PROGRAMLANMASı/KODLANMASı



ALGORITMA MANTIĞINI AŞAĞIDAKI ÖRNEKLERLE DAHA IYI PEKIŞTIRELIM.

- Sorun: Lamba Çalışmıyor.
- Soru: Lambanın Fişi Takılı mı?
- Hayır: Fişi tak (Sorun çözüldü. Bu noktadan sonra diğer adımlara bakmaya ve işlem yapmaya gerek kalmaz.)
- Evet: O zaman başka bir sorun var. Çözüm bulmak için yeni bir soru sormak gerek;
- Soru: Ampul patlak mı?
- Evet: Ampulü değiştir(Sorun çözüldü. Bu noktadan sonra diğer adımlara bakmaya ve işlem yapmaya gerek kalmaz.)
- Hayır: O zaman yeni bir lamba al.
- Bu algoritma sayesinde sorunun lambadan kaynaklandığını anladık ve çözüm olarak yeni bir lamba alınması gerektiği sonucuna vardık.

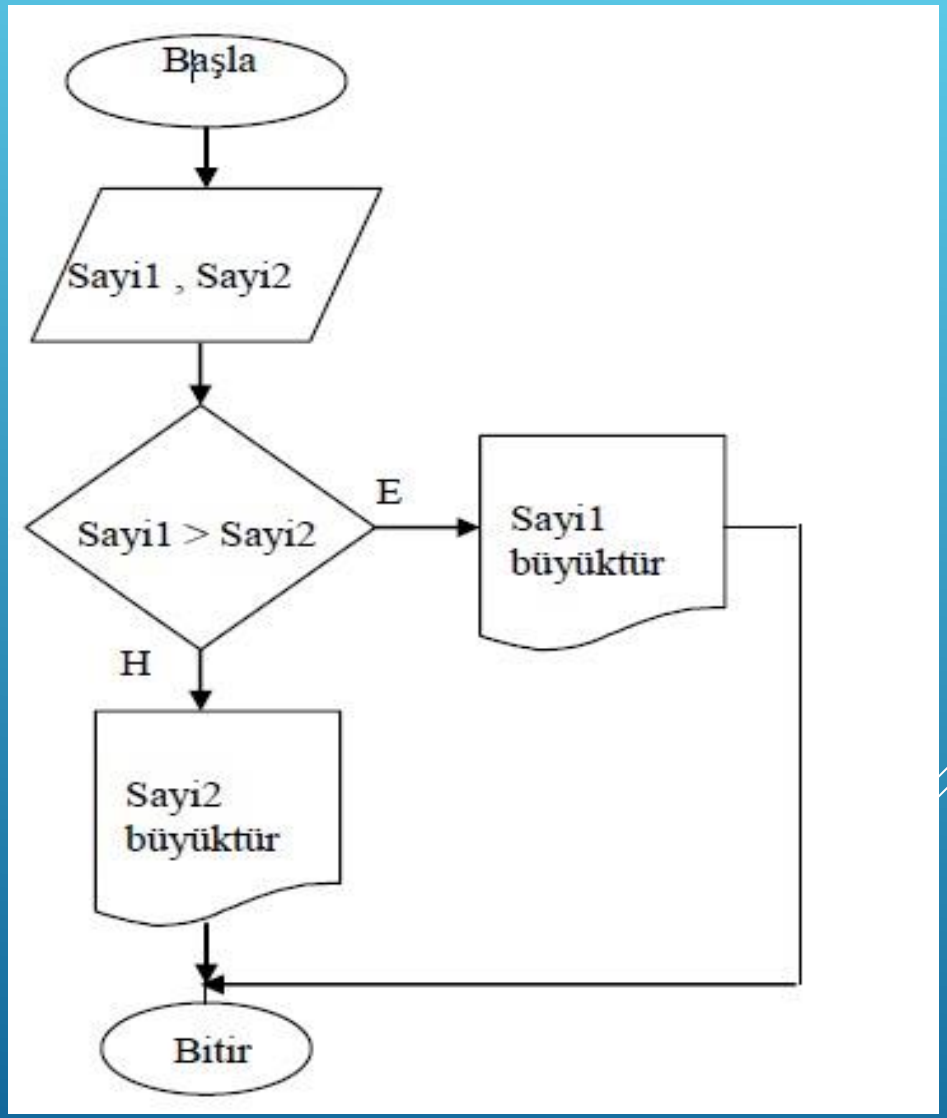

Lamba çalışmıyor
Lambanın fişi takılı mı?
Hayır
Fişi tak
Evet
Ampul patlak mı?
Evet
Ampulü değiştir
Hayır
Yeni bir lamba al

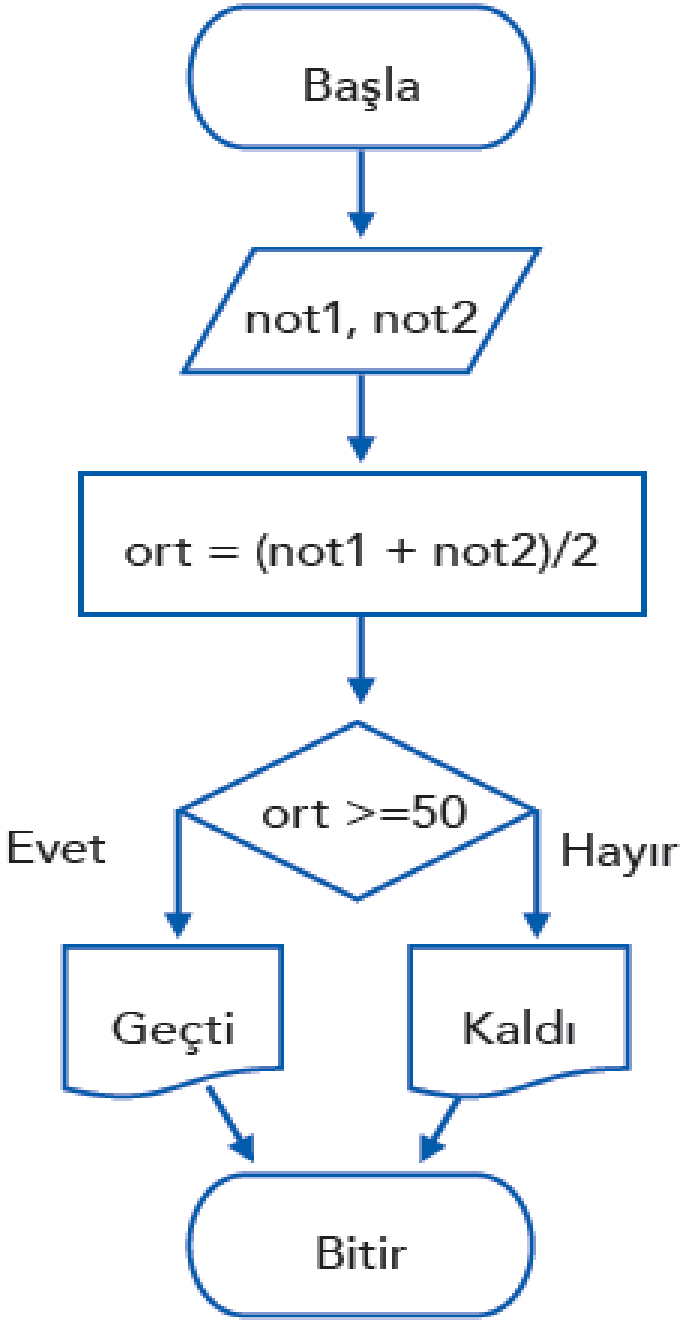
Algoritma	Akış Şeması
1. Başla. 2. Notları Oku. 3. Ortalamayı Hesapla. 4. Ortalamayı Yaz. 5. Bitir.	

Örnekler:

1- Birbirinden farklı olarak verilen iki adet sayıdan, büyük olanı bulup gösteren algoritma ve akış diyagramını tasarlayınız.

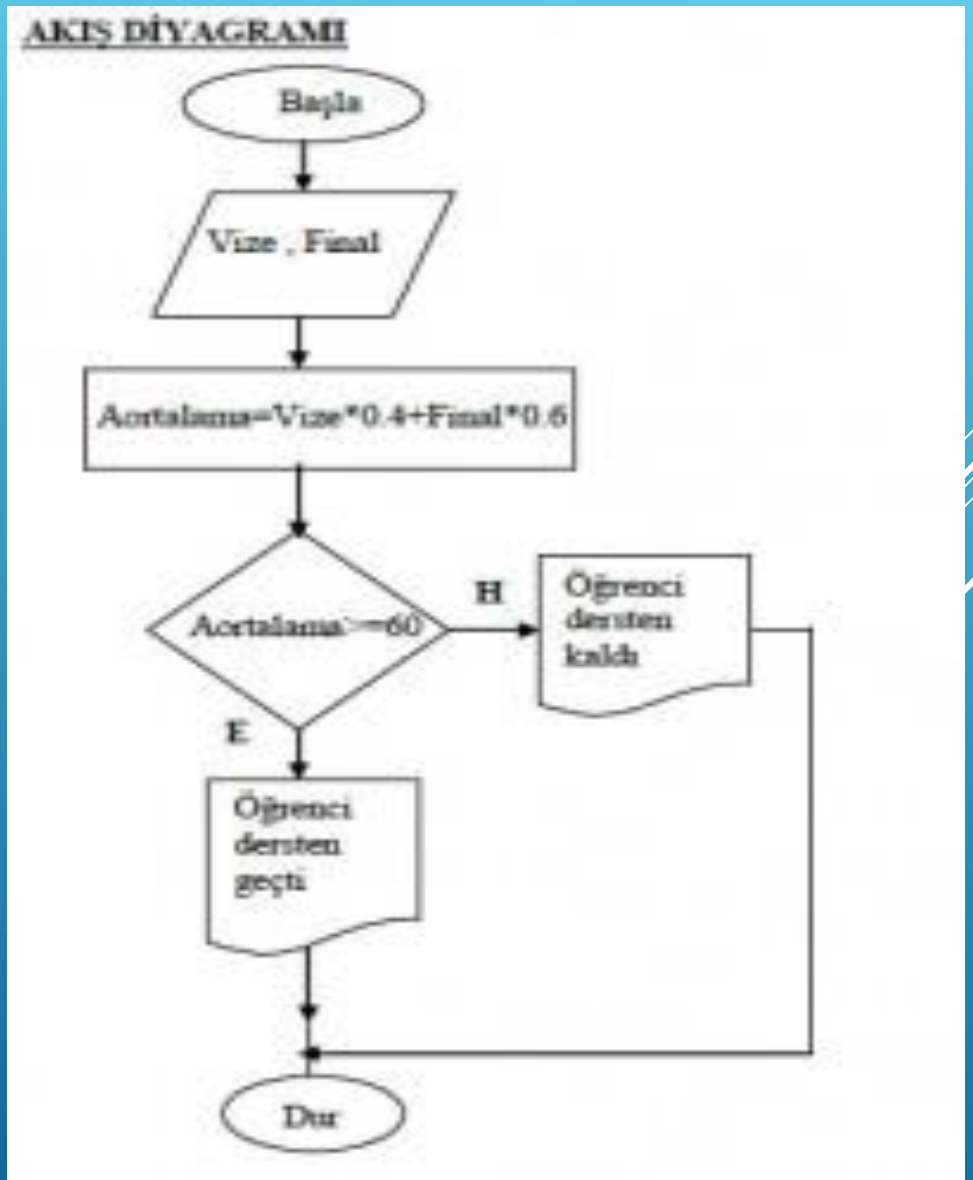
BAŞLA
OKU sayi1
OKU sayi2
EĞER sayi1> sayi2 İSE
YAZ sayi1 Büyük
DEĞİLSE
YAZ sayi2 Büyük
BİTİR



Algoritma	Akış Şeması
1. Başla. 2. Notları oku. 3. Ortalamayı hesapla. 4. Eğer ortalama >=50 ise "Geçti" yaz. Değilse "Kaldı" yaz. 5. Bitir.	

Girilen vize ve final notlarına göre öğrencinin dersten geçip geçmediğini bulan algoritma ve akış diyagramını tasarlayınız.

BAŞLA
YAZ ("Vize notunu gir")
OKU vize
YAZ ("Final notunu gir")
OKU final
*ortalama=vize * 0.40 + final * 0.60*
EĞER ortalama >= 60 İSE
YAZ "Öğrenci Dersten Geçti"
DEĞİLSE
YAZ "Öğrenci Dersten Kaldı"
BİTİR



Verilen tamsayının sıfır, pozitif ya da negatif olup olmadığını bulan algoritma ve akış diyagramını tasarlayınız.

BAŞLA
OKU sayi
EĞER Sayı>0 İSE YAZ "Bu sayı Pozitiftir"
EĞER Sayı<0 İSE YAZ "Bu sayı Negatiftir"
EĞER Sayı=0 İSE YAZ "Bu sayı Sıfırdır"
BİTİR