Bilgisayara Programlama 1

4. Hafta Algoritmalara Giriş

Öğr.Gör.Dr.Yalçın Ezginci

Problem Çözme

- Günlük hayatımızda ya da bilgisayarda çözmemiz gereken bir problem ile karşılaştığımızda yapılması gerekenler problem çözme tekniği olarak adlandırılmaktadır.
- Çözüm aşamaları :
  - Problemin hiçbir kuşkuya yer kalmaksızın anlaşılması
  - Düzenli bir biçimde düşünün; anlaşılması en kolay olan şeylerle başlayıp yavaş yavaş daha zor ve karmaşık olanlara doğru ilerleyin.
  - Çözüme ulaşmak için mümkün olduğu kadar çok parçaya bölün. Böylece gerekli ara işlemleri belirleyerek sıralayın
  - Ara işlemlerin sırasıyla uygulanması ve sonucun elde edilmesi

Problem çözme sırası

1. Problemi anlama (Understanding, Analyzing),
2. Bir çözüm yolu geliştirme (Designing),
3. Algoritma ve program yazma (Writing),
4. Tekrar tekrar test etme (Reviewing)

Yazılım Geliştirme Aşamaları

1. **Problemin Analizi:** Problemin anlaşılma, çözümünden neler beklendiği ve yaratacağı çözümün girdi ve çıktılarının neler olacağı kesin olarak belirlenmeli

2. **Tasarım:** Problemi çözmek için kullanılacak çözüm adımlarını gösteren bir liste yapılır. Problemin çözüm adımlarını gösteren bu listeye algoritma denir. Böyle bir liste tasarlanırken, ilk önce problemin ana adımları çıkarılır; daha sonra her adım için, ayrıntılı bir çözüm tasarlanır.

3. **Kodlama:** Kağıt üzerinde geliştirilen algoritma, programcının tercih ettiği bir programlama diline çevrilir.

4. **Test :** Program değişik girdiler ile çalıştırılarak ürettiği sonuçlar kontrol edilerek test işlemi gerçekleştirilir. Sonuçlar beklendiği gibi değilse doğru çalışmayan kısımları tespit edilerek düzeltilir.

5. **Belgeleme:** Bütün bu çalışmaların belli bir dosyalama sistemi içinde belgeler halinde saklanmasının sağlandığı aşamadır.

6. **Bakım:** Programın güncellenmesi, oluşan istek veya hatalara göre yeniden düzenlenmesi, yeni şartlara uygun hale getirilmesidir.

Program

Program: belirli bir problemi çözmek için bir bilgisayar dili kullanarak yazılmış deyimler dizisidir.

- Bir problemi bilgisayar ile çözmek için geliştireceğimiz programın yazımında izleyeceğimiz adımlar.
- Problemin ne olduğunun kavranması
- Çözüm için gereksinimlerin belirlenmesi
- Problemin girdi, çıktı ve işlemlerin belirlenmesi
- Problemin çözümünü veren algoritmayı yaz
- Algoritmayı bir programlama dili ile yaz
- Programın doğruluğunu test et.

Problemi çözmek

DÜŞÜN

Problemi anlamak

Problemi çözmek

Algoritmaların ifade edilmesi

1. Pseudo Code (Kaba Kod veya Yalancı Kod veya Sözde Kod), bir algoritmanın yarı programlama dili kuralı, yarı konuşma diline dönük olarak ortaya koyulması/

tanımlanmasıdır. Bu şekilde gösterim algoritmayı genel hatlarıyla yansıtır.

2. Akış şeması, algoritmanın görsel/şekilsel olarak ortaya koyulmasıdır. Problemin çözümü için yapılması gerekenleri, başından sonuna kadar, geometrik şekillerden oluşan simgelerle gösterir.

Algoritma

- Verilen herhangi bir problemin çözümüne ulaşmak amacıyla uygulanması gereken adımların hiçbir karışıklığa meydan vermemesi için sıralı ve düzenli bir şekilde yazıyla ifade edilmesine algoritma denilmektedir.
- Her algoritmanın sadece bir başlangıcı ve bir bitişi bulunmaktadır.
- Algoritmada adımlar sırasıyla işlenmektedir.

Algoritmanın hazırlanması

- Diğer bir deyişle algoritma, verilerin, bilgisayara hangi çevre biriminden girileceğinin, problemin nasıl çözüleceğinin, hangi basamaklardan geçirilerek sonuç alınacağının, sonucun nasıl ve nereye yazılacağının sözel olarak ifade edilmesi biçiminde tanımlanabilir.
- Algoritma hazırlanırken, çözüm için yapılması gerekli işlemler, öncelik sıraları göz önünde bulundurularak ayrıntılı bir biçimde tanımlanmalıdırlar.

algoritma

- Adım adım işlem basamaklarının tasarlanmasıdır.
- Algoritmalar bir görevi yerine getirmek için sonlu sayıdaki işlemler dizisidir.
- Her algoritma aşağıdaki kriterleri sağlamak zorundadır.
- 1. Girdi: Sıfır veya daha fazla değer dışarıdan verilmelidir.
- 2. Çıktı: En azından bir değer üretmeli.
- 3. Açıklık: Her işlem açık olmalı ve farklı anlamlar içermelidir.
- 4. Sonluluk: Her türlü olasılık için algoritma sonlu adımda bitmeli.
- 5. Etkinlik: Her komut kişinin kalem ve kağıt ile yürütülebileceği kadar basit olmalıdır.

Sorunun, problemin sözel ifade edilmesi

Aslında hayatımızın akışı içinde iş yapma alışkanlıklarımız, o işi çözüm şeklimiz vardır. İşte bunlar birer algoritmadır. Her durum için öğrenerek, kendi algoritmamızı geliştiririz. Ama doğru algoritma en iyi çözümü verecektir. Mesela çay demleme algoritması ;

1. Çay içme isteği ve mümkün olma durumu (Başla)
2. Çaydanlığa su koy
3. Ocağı yak
4. Çaydanlığı ocağa koy.
5. Kaynayıncaya kadar bekle
6. Demliğe sıcak suyu doldur
7. Çaydanlığa su ilave et
8. Demliğe uygun miktarda çay koy
9. Çaydanlığı ocağa koy
10. Suyu kaynayınca altını kısık hale getir ve en az 15 dk bekle
11. Çay demini aldığında servis et, afiyet olsun (Bitti)

Sözde (Pseudo) Kod

Sözde programlar, doğrudan konuşma dilinde ve programlama mantığı altında, eğer, iken gibi koşul kelimeleri ve > = < gibi ifadeler ile beraber yazılır. İyi bir biçimde yazılmış, sözde koddan, programlama diline kolaylıkla geçilebilir.

Örnek 1: Ekrana yazı yazma algoritması

- Adım 1: Başla
- Adım 2: Ekrana ‘Merhaba’ yazdır
- Adım 3: Dur

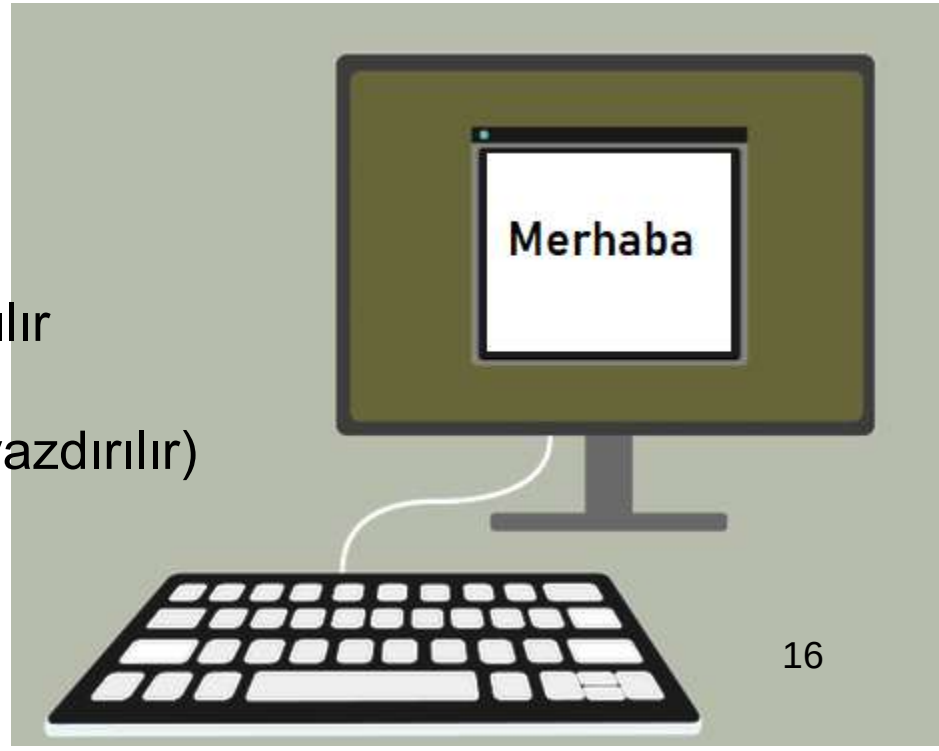
Aslında Neler oluyor:

Adım 1, Program başlatılır

Adım 2: Klavyeden bilgisayara giriş yapılır

Adım 3:Ekrana çıkış yapılır (‘Merhaba’ yazdırılır)

Adım 4: program bitti, Dur



Algoritmalarda Kullanılan Terimler-
Tanımlayıcı

- Programcı tarafından oluşturulur.
- Programdaki değişkenleri, sabitleri, kayıt alanlarını, özel bilgi tiplerini vs. adlandırmak için kullanılan kelimeler
- Tanımlayıcılar, yerini tuttukları ifadelere çağrışım yapacak şekilde seçilmelidir.
- Sözde kodlamada Türkçe karakterleri kullanabilirsiniz. Ama C programlama da ingiliz alfabesindeki A-Z veya a-z arası 26 harften 0-9 arası rakamlar kullanılır
- Sembollerden sadece alt çizgi (_) kullanılabilir.
- Tanımlayıcı isimleri harfle veya alt çizgiyle başlayabilir.
- Tanımlayıcı ismi, rakamla başlayamaz veya sadece rakamlardan oluşamaz.

Algoritmalarda Kullanılan Terimler- Değişkenler

- Programın her çalıştırılmasında, farklı değerler alan bilgi/bellek alanlarıdır.
- Değişken isimlendirilmeleri, yukarıda sayılan tanımlayıcı kurallarına uygun biçimde yapılmalıdır.

Örneğin ;

Bir ismin aktarıldığı değişken ; ad

Bir isim ve soy ismin aktarıldığı değişken; adsoyad

Ev telefon no sunun aktarıldığı değişken; evtel

Ev adresinin aktarıldığı değişken; evadres

İş adresinin aktarıldığı değişken; isadres

Algoritmalarda Kullanılan Terimler- Aktarma

- Herhangi bir bilgi alanına, veri yazma; herhangi bir ifadenin sonucunu başka bir değişkende gösterme vb görevlerde “aktarma” operatörü kullanılır.

Değişken = ifade

- Değişken yazan kısım herhangi bir değişken ismidir.
- İfade yazan kısımda ise matematiksel, mantıksal veya alfa sayısal ifade olabilir.
- = sembolü, aktarma operatörüdür ve sağdaki ifadenin/işlemin sonucunu soldaki değişkene aktarır. Bu durumda değişkenin eğer varsa bir önceki değeri silinir.

Değişken ve Sayı atama

Sayı kelimesini S ile, toplam ifadesini T ile temsil edebiliriz.

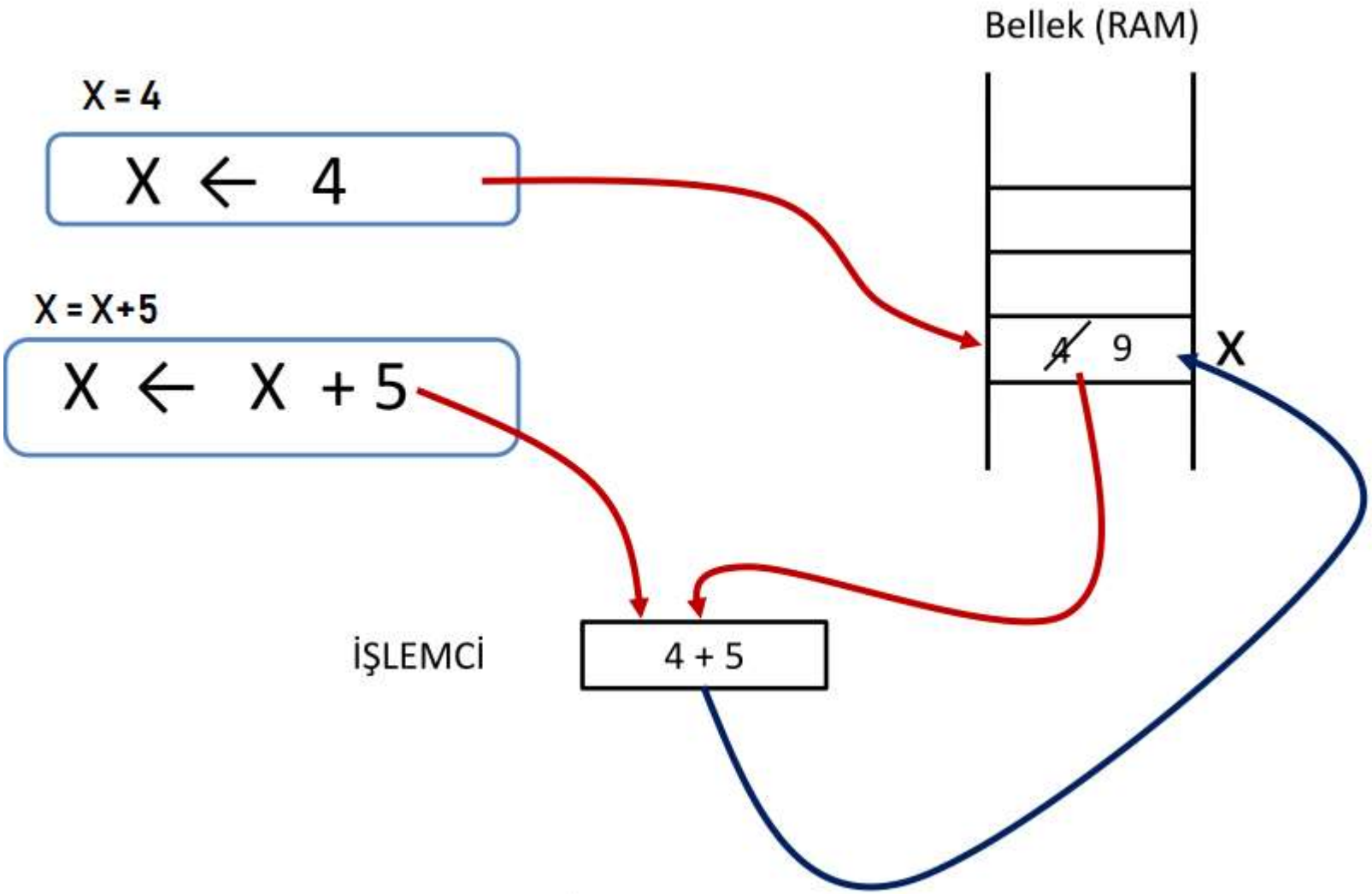
Değişkenlere değer atayabiliriz:

S=12

C dilinde ise değişkenin tipini de tanımlarız:

```
int s = 5;
float sayi= 1.56;
char c='A';
```

Değişken ve Atama işlemi ve bellek modeli



işlemler

1. Matematiksel (aritmetik) işlemler

2. Karşılaştırma (karar) işlemler

3. Mantıksal (lojik) işlemler

İşlemlerde öncelik sıraları:

1. Parantez ((……))
2. Üs alma a^b
3. Çarpma ve bölme a*b ve a/b
4. Toplama ve çıkarma a+b ve a-b
5. Karşılaştırma işlemleri (< > ..)
6. Mantıksal işlemler (değil, ve, veya, ~, &, |)

işlemler

Aritmetik İşlemler;

Programlamadaki en temel işlemlerdir. Aritmetik işlemler için kullanılan işleçlere operatör adı verilir.

Aritmetiksel Operatörler

İşleç	Adı	Örnek	Açıklama
+	Toplama	C=A+B	A ve B sayılarını toplar ve sonucu C ye aktarır.
-	Çıkarma	C=A-B	A sayısından B yi çıkarır ve sonucu C ye aktarır.
*	Çarpma	C=A x B	A ve B sayılarını çarpar ve sonucu C ye aktarır.
/	Bölme	C=A / B	A sayısını B sayısına böler ve sonucu C ye aktarır.
%	Mod	C=A % B	A sayısını B sayısına böler ve kalanı C ye aktarır.
=	Atama	C=A+B	A ve B sayılarını toplar ve sonucu C ye aktarır.

Karşılaştırma operatörleri

Bir programın akışı içerisinde, belirli bir koşula bağlı olarak akışın hangi yönde ilerleyeceğine karar vermede mantıksal operatörler kullanılır. Bu operatörler karşılaştırma işlemlerinde kullanıldıklarında sonuç olarak sadece true(doğru) yada false(yanlış) değerleri üretilir.

İşleç	Adı	Örnek	Açıklama
>	Büyük	A > B	A değişkeni B değişkeninden büyükse true, değilse false değeri üretir..
<	Küçük	A < B	A değişkeni B değişkeninden küçükse true, değilse false değeri üretir..
==	Eşit	A == B	A değişkeni B değişkeninden eşitse true, değilse false değeri üretir..
!=	Eşit Değil (Farklı)	A != B	A değişkeni B değişkeninden eşit değilse true, değilse false değeri üretir..
>=	Büyük eşit	A >= B	A değişkeni B değişkeninden büyük yada eşitse true, değilse false değeri üretir..
<=	Küçük eşit	A <= B	A değişkeni B değişkeninden küçük yada eşitse true, değilse false değeri üretir..

Operatörler

- Aritmetik işlem operatörleri:

  + - * / % → mod alma

- Karşılaştırma operatörleri:

  < <= > >= == → eşit != → eşit değil

- Mantıksal operatörler:

  && → ve || → veya

- Atama operatörleri:

  = += -= *= /= ++ --

Değişkenin değerini 1 arttırma, 1 azaltma

Değişkenin değerini arttırma, azaltma, vs için kullanılırlar.
Örneğin: A += 5; ile A = A + 5; aynıdır.

Ardışık işlemler

Programlarda, aynı değerin üzerine yeni değerler eklemek için kullanılır.

- Toplam değişkeni=Toplam değişkeni + Sayı

a=4

a=a+3 (a=7oldu) Veya a+=3 (a=7oldu)

- Çarpım değişkeni=Çarpım değişkeni * Sayı

b=3

b=b*2 (b=6 oldu) Veya b*=2 (b=6 oldu)

Benzer şekilde diğerleri..

İlave atama işlemleri:

c=5

c++ (c=6 oldu)

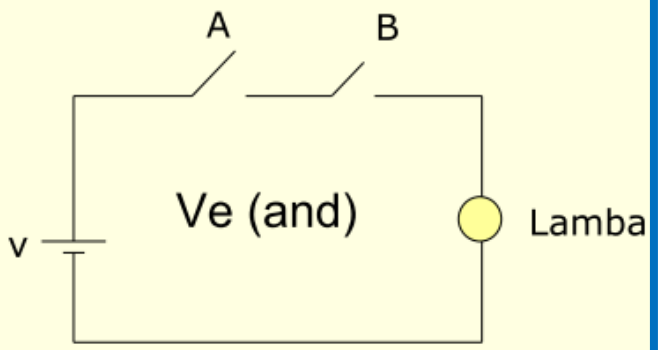
% kalan (mod) işlemleri

a=6 ise a % 2=0 (a çift sayıdır)

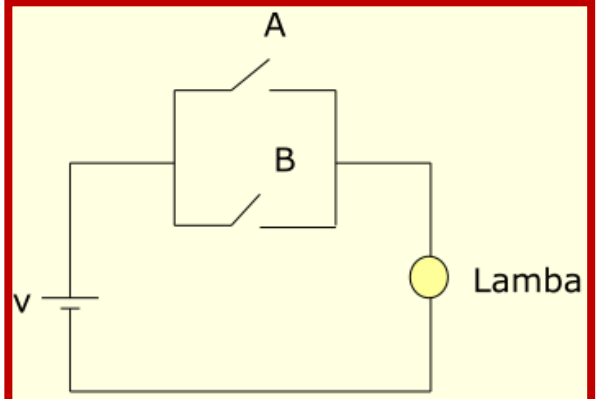
b=7 ise b % 2=1 (b tek sayıdır)

c=43 ise c%3=1 (43 sayısının 3'e bölümünden kalan 1'dir)

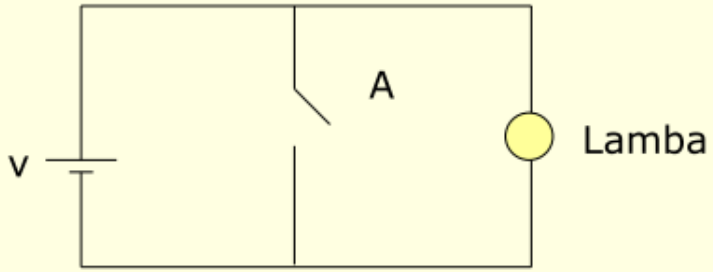
Mantıksal işlemler



A	B	A ve B
0	0	0
0	1	0
1	0	0
1	1	1



A	B	A veya B
0	0	0
0	1	1
1	0	1
1	1	1



A	A DEĞİL (A')
0	1
1	0

işlemler

Eğer (x>5 VE x<8) ise işlem(ler)

Eğer (x=5 VEYA x=6 VEYA x=7) ise işlemler

Eğer (final<50 VEYA ortalama<35) ise Harf_Notu=FD

Hata kontrolü:

Eğer (vize< 0 VE vize > 100) ise

YAZ “ vize notu [0-100] aralığında olmalıdır. “

Örnek sözde kod aşamaları

Verilen bir sıcaklık derecesine göre suyun durumunu belirten bir program yazınız.

1. aşama: Örnek tanımlama ve Giriş/Çıkış işlemleri:

–Bu Program, Sıcaklığa göre suyun durumunu gösterir

–Suyun durumları Sıvı, Buz, Buhar dır

–Derece cinsinden sıcaklığı giriniz: örnek giriş: 125 derece

–örnek çıkış: Su, BUHAR halinde

2. Aşama: İstenilen programın Pseudo Kodu:

1. Başla
2. Ekrana yaz

   (Klavyeden sıcaklık değeri gir –derece olarak)
3. Girilen Sıcaklığı Oku.
5. Eğer Sıcaklık < 0 ise Durum=“Buz”
6. Eğer Sıcaklık>= 100 ise Durum=“Buhar”
7. Eğer Sıcaklık>= 100 Değilse Durum =“Sıvı”
8. Yaz Durum
9. Dur

Örnek 2: iki sayının toplanması

Yöntem, problemi çözmek için önce doğru ve tam anlayalım sonra sözlü basit adımlara bölelim :

1. Problem matematiksel bir toplama işlemidir.
2. Çözüm için iki sayıya ve bu sayıları saklayacağımız değişkenlere ihtiyacımız var.
3. Sayıların belirlenmesi ve toplama işleminin yapılması.

Örnek 2: iki sayının toplanması, devam

4.Algoritma:

Değişkenler T, s_1, s_2

s_1=?

s_2=?

$T=s_1+s_2$

T

5.Program

C program
(daha sonra)

6. Kontrol et, adımları işlet:
mesela, $s_1=2$, $s_2=3$ olsun
$T=s_1+s_2=2+3=5$

Düzenlenmiş algoritma

Soru: iki sayının toplamını bulan algoritmayı hazırlayınız.

Adım 1: Başla

Adım 2: Toplamı, birinci ve ikinci sayıları tanımla

Adım 3: Birinci sayıyı oku

Adım 4: İkinci sayıyı oku

Adım 5: Sayıları topla ve toplam değişkenine at

Adım 6: Toplamı görüntüle

Adım 7: Dur

klavyeden OKU,
ekrana YAZ

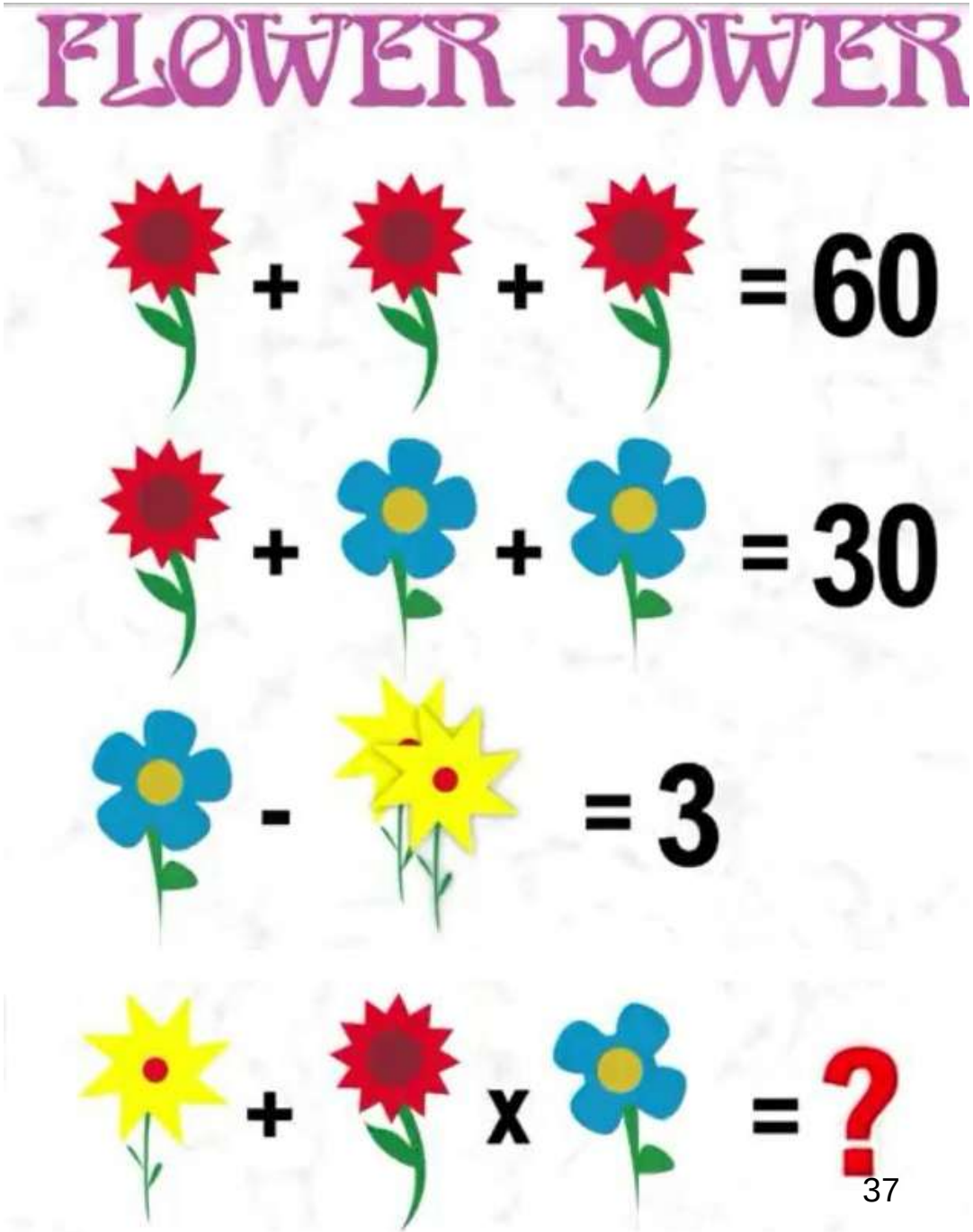
1 Başla
2 Tanım T, s_1, s_2
3 Oku s_1
4 Oku s_2
5 T= s_1 + s_2
6 Yaz T
7 Dur

Problemin çözümü

Yanlış çözüm

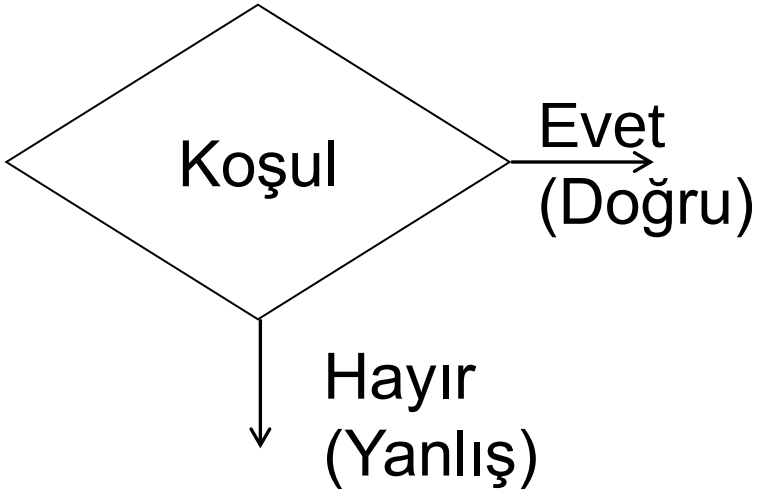
Nasıl çözersiniz?

İşlemleri basamaklandırın



Karar, Şarta bağlı dallanma

- mantıksal kararlar
- Verilen koşulun doğru yada yanlış olmasına göre iş akışı yönlendirilir.
- Eğer .. İse .. Değilse … şeklinde ifade edilir.
- «Eğer (şart) işlem» şeklinde formülize edilir
- ‘Eğer Sayı1, sıfırdan büyükse pozitiftir



Karar (Eğer) yapısı

1. **Eğer** (Şart) işlem/ler **son**

2. **Eğer** (Şart) 1.işlem değilse 2. işlem **son**

3. **Eğer** (Şart) 1.işlem
   değilse eğer 2.işlem
   değilse eğer 3.işlem ..
   değilse n.işlem **son**

Klavyeden girilen bir sayının pozitif, negatif veya sıfıra eşit olma durumunu hesaplayıp yazdıran algoritma.

1- Başla

2- OKU X

3- Eğer(x>0) YAZ “Pozitif”

4- Eğer(x<0) YAZ “Negatif”

5- Eğer(x=0) YAZ “Sıfır”

6- Bitir

Döngüsel Akış Şemaları

- Sorunun çözümü için, çözümde yer alan herhangi bir adım ya da aşamanın birden fazla kullanıldığı, yani tekrarlı işlemler olduğunda döngülere başvurulur. Program ilerlerken bir kontrol noktasından gerideki bir noktaya atlanır.
- Döngü genel olarak;

  giriş ya da başlangıç değeri verme,

  hesaplama,

  kontrol

  ve çıkış biçiminde olmaktadır.

Algoritmalarda Kullanılan Terimler-Sayaç

Programlarda bazı işlemlerin belirli sayıda yaptırılması veya işlenen/üretilen değerlerin sayılması gerekebilir.

say=say+1

Bu işlemde sağdaki ifadede değişkenin eski değerine 1 eklenmekte; bulunan sonuç yine kendisine yeni değer olarak aktarılmaktadır.

Bu tür sayma işlemlerine algoritmada sayaç adı verilir.

Sayacın genel formülü;

Sayaç değişkeni=sayaç değişkeni+adım

Örnek; X=X+3

Örnek; S=S-5

Sayaç kullanımı:

Aşağıdaki algoritmada 1-5 arası sayılar, ekrana yazdırılmaktadır. 1-5 arası sayıları oluşturmak için sayaç(s=s+1) kullanılmıştır.

1. **Başla**
2. **S=0**
3. **Eğer s>4 ise git 7**
4. **S=S+1**
5. **Yaz S**
6. **Git 3**
7. **Dur**

	Eski S	Yeni S	Ekrana Yazılan
1. çevrim	0	0+1=1	1
2. çevrim	1	1+1=2	2
3. çevrim	2	2+1=3	3
4. çevrim	3	3+1=4	4
5. çevrim	4	4+1=5	5

20’den 25’e kadar olan sayıların toplamını bulup yazdıran algoritmayı yazınız

1- BAŞLA

2- toplam = 0

3- sayı = 20

4- toplam = toplam + sayı

5- sayı = sayı + 1

6- Eğer (sayı<25) ise GİT 4

7- YAZ toplam

8- Dur