Programlama Temelleri

TEMEL BİLGİSAYAR KAVRAMLARI

Donanım:

Giriş birimleri- klavye, fare, okuyucular, mikrofon, dokunmatik aletler

Ana işlemci ve yardımcılar

Saklama birimleri (CD-ROM, diskler, bellekler...)

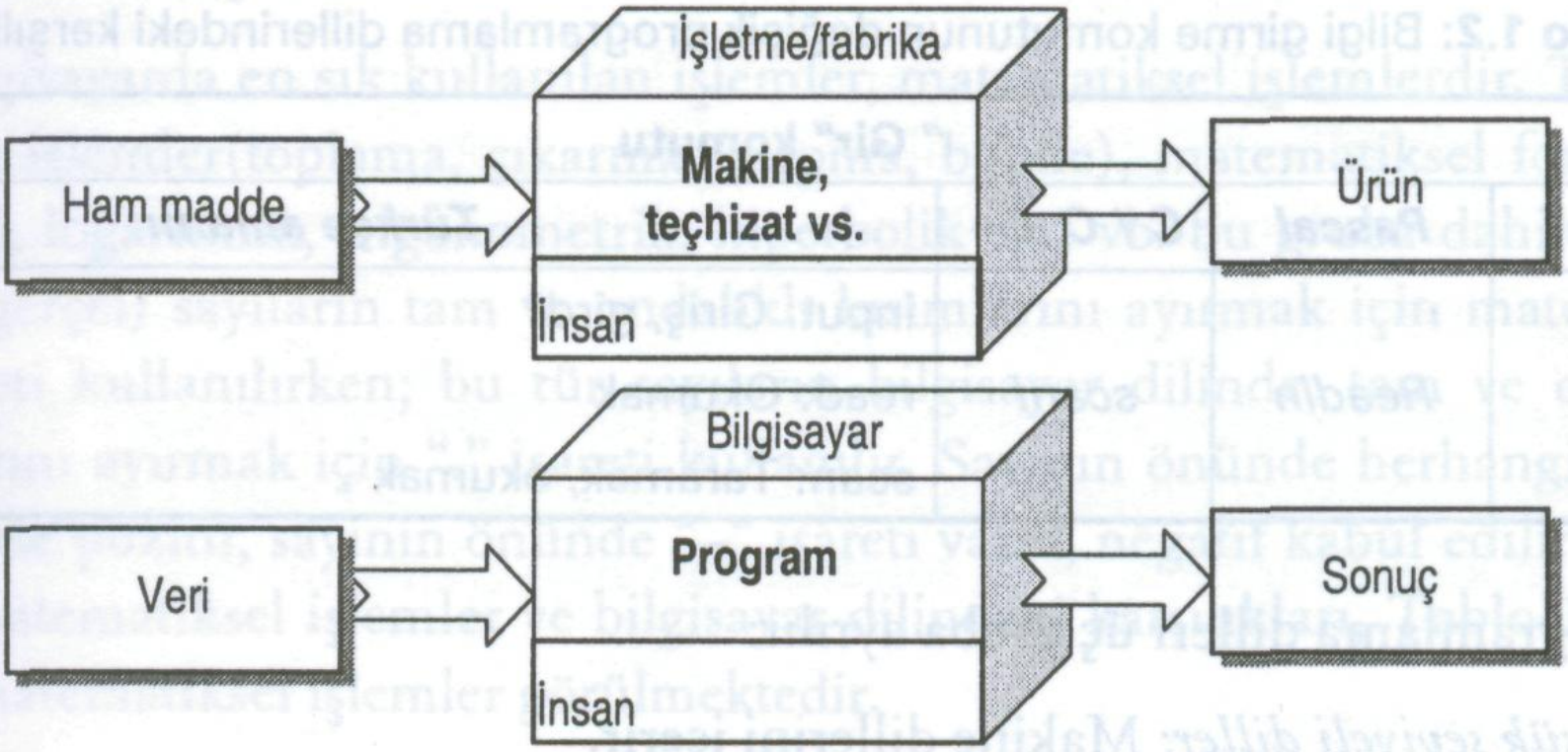
Çıkış birimleri (ekran, yazıcı-çizici, hoparlör...)

Yazılım (program):

Sistem yazılımları (WINDOWS, MAC OS, LINUX, Derleyiciler)

Uygulama yazılımları: Ofis yazılımları, İnternet yazılımları, oyunlar, muhasebe...

Program ve programlama dili nedir?



Şekil 1.1: İşletme/fabrika ve bilgisayardaki işlem akışı

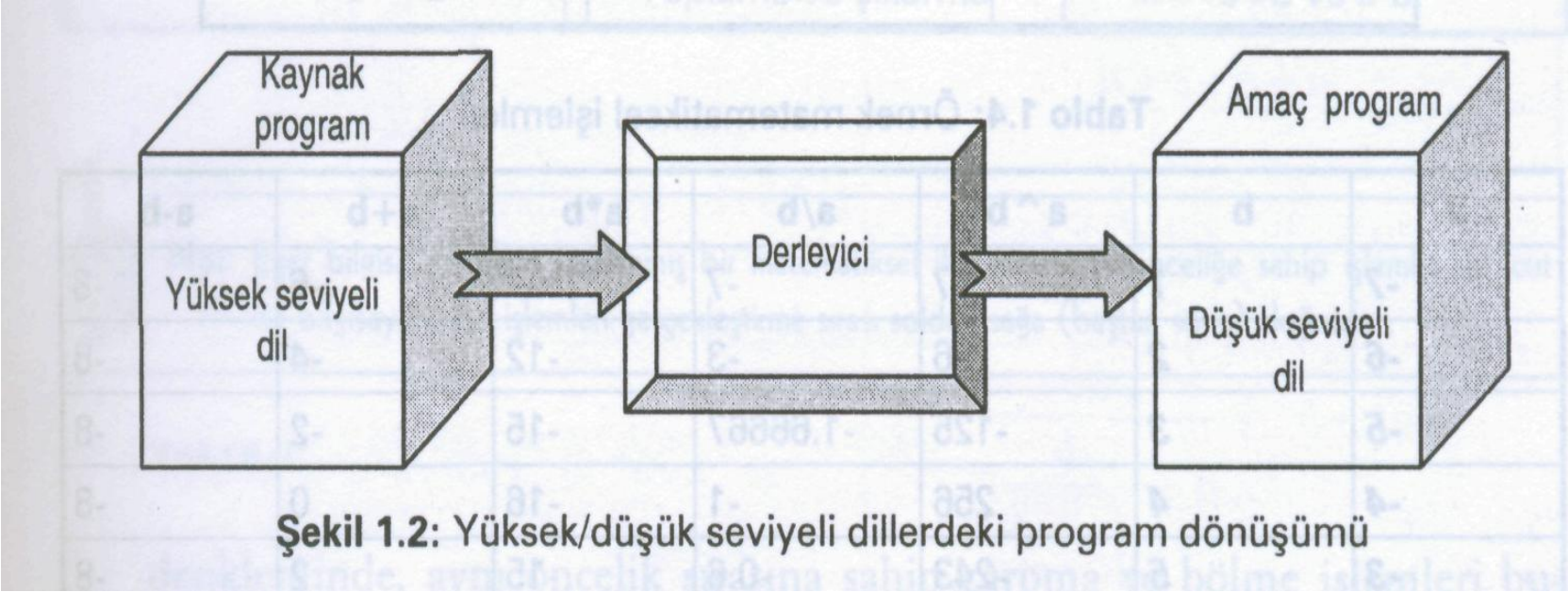
Program ve programlama dili nedir?

- İnsanla bilgisayar arasındaki iletişim aracı
- Programlama dili, programcı ile bilgisayar arasındaki iletişimi sağlayan bir araç olup programların yazılımında kullanılan bir notasyondur (simgeler ve özel komutlar, komut parçacıkları).

Program dönüşümü



- Düşük seviyeli diller (assembly : MOV AL, 61h)
- Orta seviyeli diller (PIC programlama: SUBWF f,d [d = f − W])
- Yüksek seviyeli diller (C/C++, Pascal, QBasic : printf(), writeln(), PRINT)



Şekil 1.2: Yüksek/düşük seviyeli dillerdeki program dönüşümü

İşlemler

- Matematiksel (aritmetik) işlemler

(toplama, çıkarma, çarpma, bölme, üs alma)

- Karşılaştırma (karar) işlemleri

(eşit, eşit değil, büyük, küçük, büyü eşit, küçük eşit)

- Mantıksal (lojik) işlemler

(ve, veya, değil)

Bilgisayarda problem çözme aşamaları:

- Problemin tanımı
- Çözüm yolunun tespiti
- Algoritmanın hazırlanması
- Akış diyagramının çizilmesi
- Programın hazırlanması
- Hazırlanan programın test edilmesi
- Uygulama

ALGORİTMA

Algoritma

- 9. yüzyılda Farslı matematikçi Al-Khowarizmi (Al-Harezmi) tarafından bulunmuş ve onun adında türetilmiştir.

Algoritma

- Problem çözmek için geliştirilen **Adım – adım** çözüm yöntemidir.
- Verilen bir problemi çözmek için tasarlanan talimatlar bütünüdür.
- Bir sorunun çözümü için sunulan mantıksal ve sembolik anlatımdır.

Algoritma çalışmasında iki önemli görev vardır:

1. Belirli bir problemi çözmek için bir algoritma tasarlamak
2. Verilen bir algoritmayı analiz etmek

   - Bu iki görev birbiriyle ilişkilidir.
   - Algoritmanın analizi yeni algoritmaların tasarlanmasını sağlar.

Program geliştirme yöntemi

- Bilgisayar programcılarının program geliştirme yöntemi aşağıdaki adımlardan oluşur:
  - Problem
  - Analiz
  - Tasarım
  - Uygulama
  - Sınama
  - Bakım

Problem

- Bu ilk adımda programcı, problemi anlamaya çalışır.
- Problemin ne olduğunu, çözümde nelerin gerekli olduğunu, lazım olan ön bilgileri belirler.
- Problemi anlamanın çözümün yarısı olduğu unutulmamalıdır!

Analiz

- Bu adımda problemi çözüm için verilen değerler ile programın elde etmesi gereken değerlerin ne olduğu belirlenmelidir.
- Girdiler ve çıktılar madde madde yazılır.
- Bu girdi ve çıktı arasındaki ilişki belirlenir.
- Belirlenen ilişki formüller ile açıklanabilir.
- Problem adımı başarılı bir şekilde gerçekleşmeden analiz adımına geçilmemelidir.
- Problem çözümünde ilk iki adım çok önemlidir.
- Çözüm için bir fikir bulunamamışsa problem tekrar gözden geçirilmelidir.

ÖRNEK: Taban ve yükseklik değerleri verilen üçgenin alanını hesaplayınız.

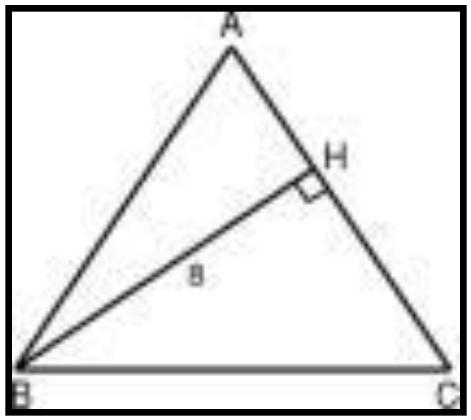
PROBLEM:

- Bu problemde üçgen alanını bulmak için gerekli olan formülün bulunması zorunludur.
- Formül olduğunu düşünmek çözüm yöntemi hakkında bir fikir oluştuğu anlamına gelir.

ÖRNEK: Taban ve yükseklik değerleri verilen üçgenin alanını hesaplayınız.

ANALİZ

- Girdiler:
  - Taban ve Yükseklik
- Çıktılar:
  - Alan



ÖRNEK: Taban ve yükseklik değerleri verilen üçgenin alanını hesaplayınız.

- Verilen değerler ile çıktı arasındaki formüle edilmiş ilişkiyi belirlemek gerekir.
- Gerekli olan girdileri ve birbirleri arasındaki ilişkileri belirleyerek bir problemi modelleme işlemine *soyutlama* (abstraction) adı verilir.

Alan=(Taban X Yukseklik)/2

Tasarım

- Problemin çözümü için gerekli olan çözüm aşamalarını mantıksal sıra içerisinde yazılması aşamasıdır.
- Yapılan işleme algoritma adı verilir.
- İyi bir algoritmanın iki şartı vardır.
  - Bütün adımlar mantıksal bir düzen ve sıra içerisinde verilmeli
  - Verilen adımların tamamı çalıştırılabilir olmalıdır.

Tasarım

- Algoritma tasarlanırken bütün problem parçaları aynı anda çözülmeye çalışılmaz.
- Problem kendi içinde önemli parçalara ya da alt problemlere ayrılır ve sonra bu alt problemler çözülerek esas problemin çözümüne gidilir.
- Bu yönteme *yukarıdan aşağı tasarım* (top-down design) veya böl ve yönet (divide&conquer) adı verilir.
- Algoritma bütün tanımlamaları içerecek şekilde adım adım yazılır.
- İlk adım "BAŞLA", son adımda her zaman "DUR" olur.

Tasarım

- Algoritma tasarımının önemli bir parçası da algoritmanın doğru çalışıp çalışmadığının adım adım izlenerek kontrol edilmesidir.
- Bu kontrol etme esnasında erkenden fark edilen hatalar programcıya hem zaman kazandıracak, hem de boş yere enerji harcamasına engel olacaktır.

Aşağıda örnek bir algoritma çalışması verilmiştir.

- Örnek: Problem “tahtaya adını yazma" işidir.
- Algoritma aşağıdaki gibi yazılabilir.

Tahtaya adını yazma

1. BAŞLA
2. Yerinden kalk
3. Yönün tahtaya doğru mu?
   - Hayırsa Tahtaya Dön
   - Evetse adım 4 e git
4. Tahtaya doğru yürü
5. Tahtaya geldin mi?
   - Hayırsa adım 4‘e
   - Evetse adım 6'ya git
6. Kalemi al
7. Adını yaz
8. DUR



Tahtaya adını yazma

- Burada emirler, belli sorgulamalar yapılarak ve mantıksal bir sıra içinde verilmiştir.
- Yerinden kalk emri verilmeden ondan yürümesi istenemez.
- Kalemi almadan adını yaz emrinin verilmesi doğru olmaz.
- Sorgulamalarla da işlemi yapıp yapmadığı kontrol edilmiştir.

Tahtaya adını yazma

- Bu örnek, sadece adımların tutarlılığını ve mantıksal sırasını göstermek içindi.
- Yoksa bilgisayar bu tür işleri yerine getiremez.
- Esasında kullanıcılar bilgisayarlara belli girdiler verir.
- Onlarda programcının verdiği adımlara göre bu girdiler üzerinde matematiksel ve mantıksal işlemler yaparak bir çıktı üretirler.

Örnek: Üçgenin Alanı

- Yukarıdaki verilen üçgen alanını hesaplama işlemi için gerekli olan algoritma, sözde kod (pseudocode) kullanarak aşağıdaki gibi oluşturulabilir.
  1. BAŞLA
  2. Taban değerini al
  3. Yükseklik değerini al
  4. Taban ile yüksekliği çarp sonucu ikiye böl
  5. Sonucu yaz
  6. DUR

Üçgenin Alanı

- Önceki algoritma kendi içerisinde tutarlı ve mantıksal sıra ile yazılmıştır.
- Değerler verilmeden alan hesabı işlemini yaptırmak doğru olmazdı.
- Fakat algoritmalarda daha öncede belirtildiği gibi kısa ve basit kelimelerle emirler verilecektir.
- Böylece uygulama yapılacak program kodlarına adapte olmak kolaylaşacaktır.

Üçgenin Alanı

1. BAŞLA
2. OKU taban
3. OKU yükseklik
4. alan= (Taban X Yükseklik)/2
5. YAZ alan
6. DUR

Üçgenin Alanı

- Uzun cümleler yazmak yerine kısa basit kelimelerle ve formülize edilmiş işlemlerle çözüm sağlanmış oldu.
- Her algoritma BAŞLA emri ile başlar ve DUR emriyle de biter.
- Dışarıdan verilmesi gereken bilgiler için OKU, sonuçları göstermek içinde YAZ emri kullanılabilir.

Örnek: İki sayıyı toplamak için gerekli programa ait algoritmanın oluşturulması.

Algoritma:

1. Başla
2. Birinci sayıyı gir
3. İkinci sayıyı gir
4. İki sayının toplamını yap
5. Toplamın değerini yaz
6. Bitir

1. Başla
2. X değerini gir
3. Y değerini gir
4. toplam = X+Y
5. toplam' ı yaz
6. Bitir

Örnek :

- Beş sayının toplamını ve ortalamasını veren programa ait algoritmanın oluşturulması
- Toplam adı için **Toplam**
- Ortalama adı için **Ort**
- Girilen sayılar için **X**
- Arttırma için **Sayac** kullanılırsa

Algoritma:

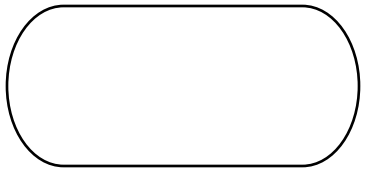
1. Başla
2. Toplam = 0, Sayac = 0
3. X'i gir
4. Toplam= Toplam+X
5. Sayac = Sayac +1
6. Eğer Sayac <5 ise 3'e git
7. Ort= Toplam/5
8. Toplam ve Ort değerlerini yaz
9. Bitir

Akış Şemaları

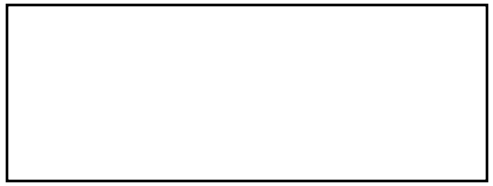
Akış Şemaları

- Bir süreci belirleme veya bir projenin adımlarını planlama için görsel bir sunum oluşturur.
- Proje ve süreç ile ilgili kişilere ortak bir dil veya bir referans noktası sağlar.
- Algoritmadaki ifadeler şekillerin içerisine yazılır.
- BAŞLA ve DUR şekilleri hariç her bir şeklin anlamı belli olduğundan içlerine tekrar anlamla aynı olacak kelime ve ifadeler yazılmaz.

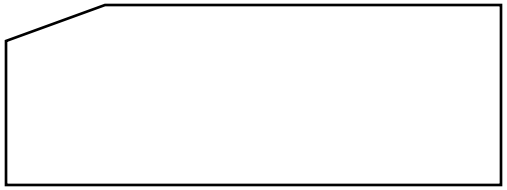
Akış Şeması Şekilleri



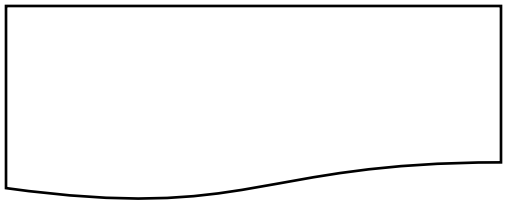
Başlama ve bitiş işlemleri



Atama ve hesaplama işlemleri

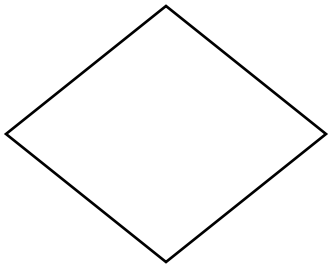


giriş – okutma işlemleri



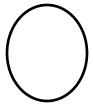
görüntüleme – yazdırma işlemleri

Akış Şeması Şekilleri

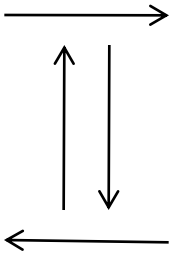


Karar ve kontrol işlemleri

{eğer (koşul bildirimi veya karar)}



Bağlantı işlemleri



Akış yönünü belirten işlemler

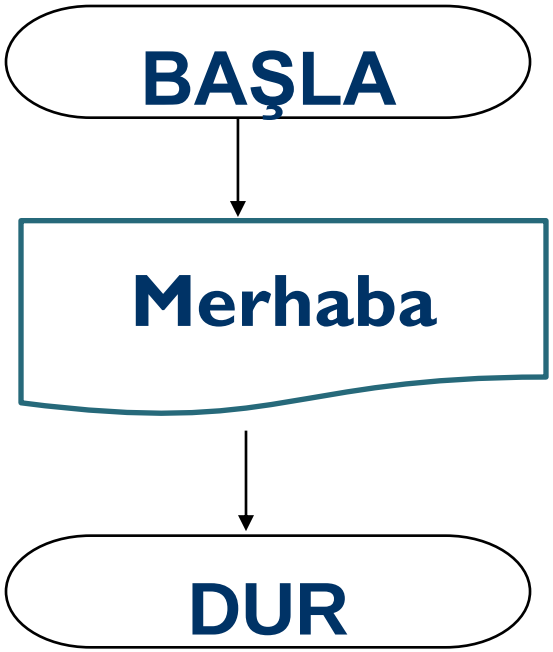
Örnek

- ✓Ekrana "Merhaba" yazdıran program

1-Başla

2-Merhaba yaz

3-Dur



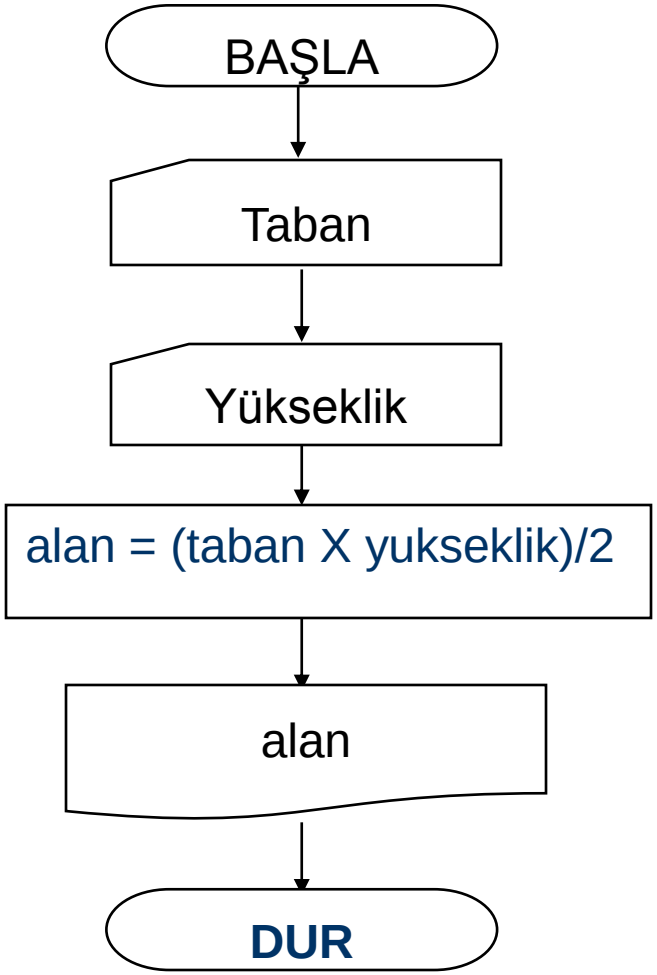
Program-1

ÖRNEK: Taban ve yükseklik değerleri verilen üçgenin alanını hesaplayınız.

ALGORİTMA

1. BAŞLA
2. OKU taban
3. OKU yukseklik
4. alan= (Taban X Yukseklik)/2
5. YAZ alan
6. DUR

AKIŞ ŞEMASI



Uyarılar...

- Akış şemalarında tek bir başlangıç simgesi olmalıdır
- Bitiş simgesi birden çok olabilir.
- Karar simgesinin haricindeki simgelere her zaman tek giriş ve tek çıkış yolu bulunur.
- Bağlaç simgesi sayfanın dolmasından ötürü parçalanan akış şemasının öğelerini birleştirmede kullanılır.
- Simgeler birbirleri ile tek yönlü okla bağlanırlar.
- Okların yönü algoritmanın mantıksal işlem akışını tanımlar.

Uygulama

- Algoritması oluşturulan programın herhangi bir dilin (Basic, C, Fortran…) kurallarına uyarak ve komutlarını kullanarak yazılma aşamasıdır.
- Sözde kod (Pseudocode), kullanılan programlama diline dönüştürülür.

Sınama

- Program yazıldıktan sonra, uygun aşamalardan geçirilip, verilen girdilere göre uygun **çıktıları** üretip üretmediği kontrol edilir.
- Bu aşamada program girdi olarak verilebilecek tüm değerlere göre; özellikle de uç değerlere göre sınanır.
- Örneğin girdi olarak bir tamsayı girilmesi gereken programda, girilen sayının negatif olması halinde yapılacak işlemler tanımlanmamış olabilir.
- Bu gibi durumlar bu aşamada gözden geçirilmelidir.

Bakım

- Sınama aşamasından sonra ortaya çıkan aksaklıklar varsa bunlar giderilir.

ÖRNEK :

- Çalıştığı gün sayısı ve yevmiyesi girilen işçinin maaşını hesaplayan algoritmayı ve akış diyagramını yapınız.

PROBLEM

- Bu program çalıştığı gün sayısı ve günlük yevmiyesi verilen işçinin maaşını bulacaktır.
- Maaş hesaplanırken işçinin çalıştığı gün sayısı ile günlük yevmiyesi çarpılarak maaş hesaplanabilir.

ANALİZ

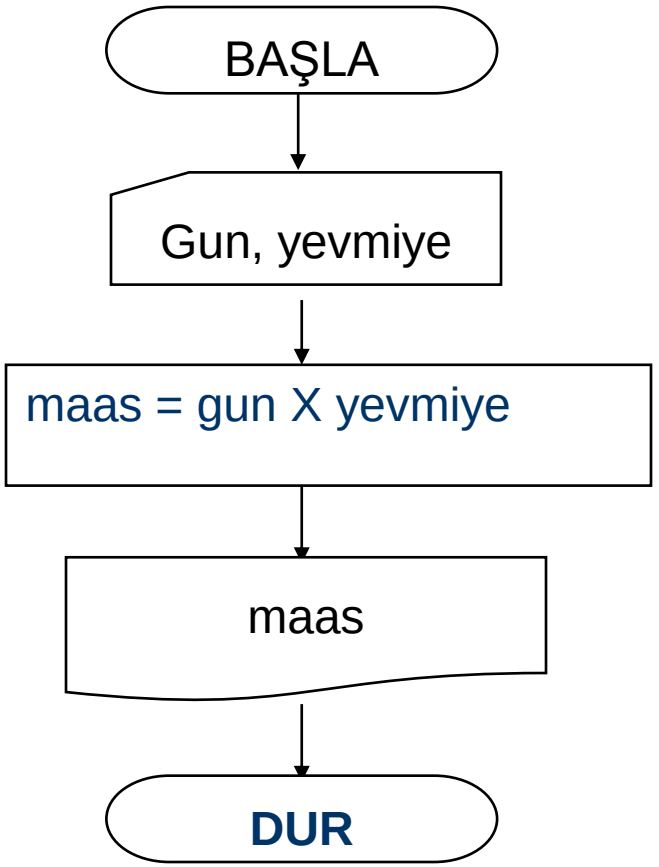
- Girdiler:
  - Gün
  - Yevmiye
- Çıktılar:
  - Maaş
- İlişki
  - Maas=Gun X Yevmiye

TASARIM

ALGORİTMA

1. BAŞLA
2. OKU gun, yevmiye
3. maas=gun x yevmiye
4. YAZ maas
5. DUR

AKIŞ ŞEMASI

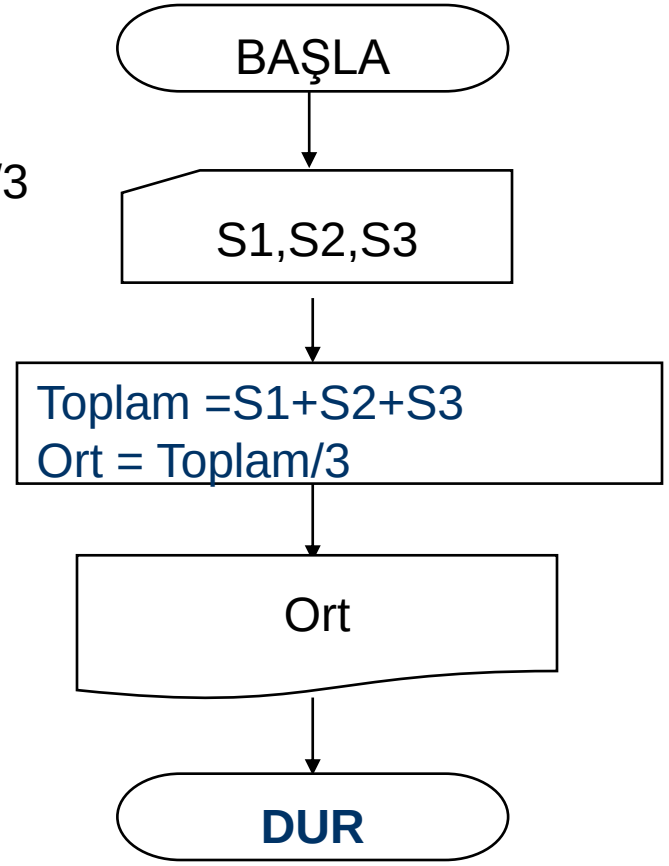


Örnek: Klavyeden girilen 3 sayının aritmetik ortalamasını bulan programın algoritma ve akış şemasını yazın.

Algoritma;

1. Oku S1,S2 ve S3
2. Toplam=S1+S2+S3
3. Ort =Toplam/3
4. Yaz “Ortalama=”;Ort
5. Dur

Yerine
Ort = (S1+S2+S3)/3
yazılabilir.



10,15,8 değerleri için bellekteki durum ve ekran çıktısı:

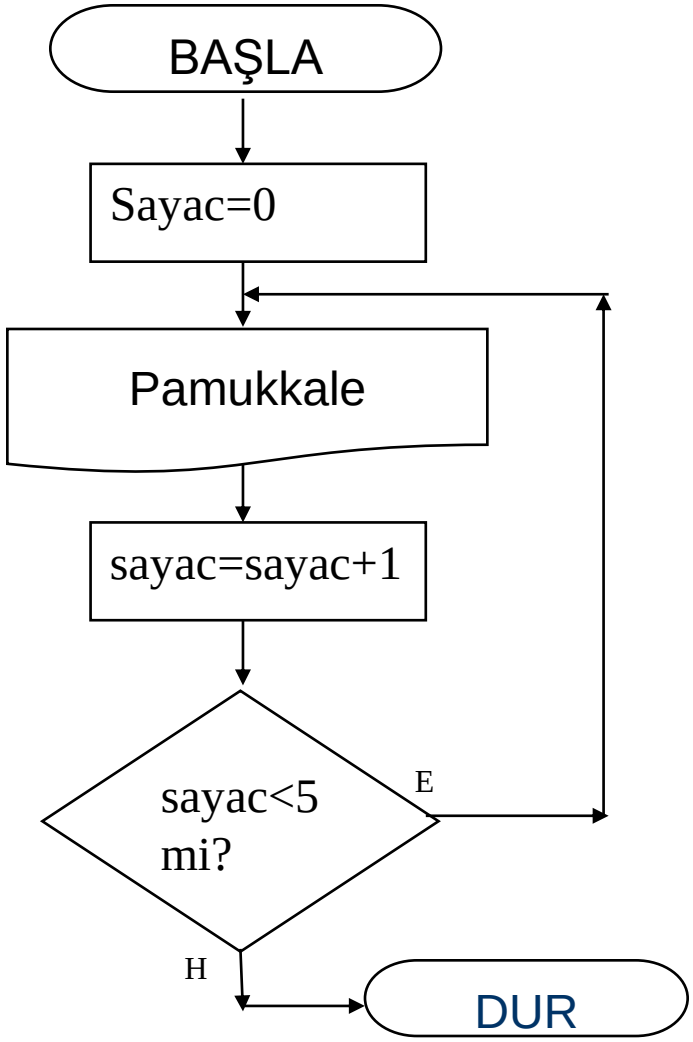
S1	S2	S3	Toplam	Ort
10	15	8	33	11

Ekran Çıktısı
Ortalama=11

Örnek:Ekrana 5 defa "Pamukkale" yazdıran programın algoritma ve akış şemasını yazın?

Algoritma;

1. Basla
2. sayac=0
3. YAZ "Pamukkale", sayac
4. sayac=sayac+1
5. Eğer sayac<5 GİT 3
6. DUR

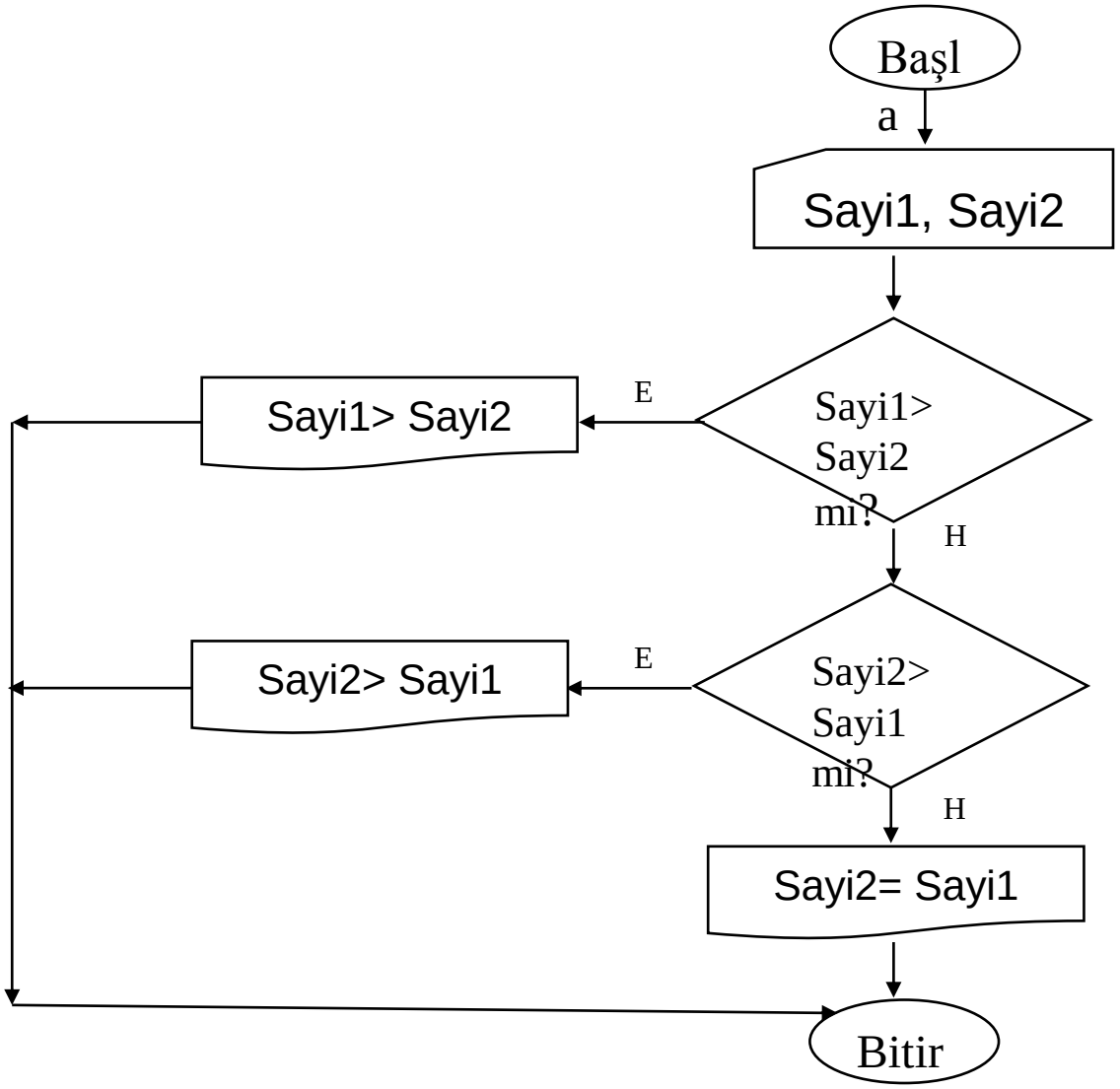


Örnek: Klavyeden girilen 2 sayıyı karşılaştırıp sonucu ekrana yazdıran algoritma ve akış şemasını yazın?

Algoritma;

1. BAŞLA
2. OKU sayi1,sayi2
3. EĞER sayi1>sayi2 İSE YAZ “Sayi1 sayi2’den büyüktür”
4. Değilse EĞER sayi2>sayi1 İSE YAZ “Sayi2 sayi1’den büyüktür”
5. DEĞİL İSE YAZ “Sayi1 sayi2’ye eşittir”
6. BİTİR

Örnek: Klavyeden girilen 2 sayıyı karşılaştırıp sonucu ekrana yazdıran algoritma ve akış şemasını yazın?



Algoritmada kullanılan terimler

- Tanımlayıcı (programdaki değişkenleri, sabitleri, kayıt alanlarını, özel bilgi tiplerinin adlandırılması veya belirlenmesi)
- Değişken (x, ad, tel_no, sayi1 vs.)
- Aktarma (değişken = ifade)
- Sayaç (sayac = sayac + 1, x = x+3, s=s-5)

Algoritmada kullanılan terimler

- Döngü
  - Döngü değişkeninin başlangıç ve bitiş değeri,
  - Artma ya da azalma miktarı belirlenir.
- Ardışık toplama ve çarpma
  - top_degis= top_degis+sayi
  - carp_degis= carp_degis*sayi

Örnek:

Bir öğrencinin numarası, adı ve soyadı, vize notu ve final notu girilmektedir. Verilere göre vize notunun %30'ü ile final notunun %70'i alınarak ortalaması hesaplanmak isteniyor. Algoritmayı maddeler halinde yazınız.

Çözüm:

1. Başla
2. Öğrencinin numarasını (**No**) gir
3. Öğrencinin adını ve soyadını (**Adsoy**) gir
4. Öğrencinin vize notunu (**Vize**) gir
5. Öğrencinin final notunu (**Final**) gir
6. **Ort** = 0.3*__Vize__ + 0.7*__Final__
7. Numara (**No**) ve ortalamayı (**Ort**) yaz
8. Dur

No	Adsoy	Vize	Final	Ort	Ekrana yazılan	
9801.0033	Ahmet	30	70	58	9801.0033 – 58	→ 1. çalıştırma
9804.0099	Mehmet	70	90	84	9804.0099 – 84	→ 2. çalıştırma
9712.0123	Ali	60	30	39	9712.0123 – 39	→ 3. çalıştırma

Örnek:

Klavyeden girilen herhangi bir N sayısının faktöriyelini bulan programın algoritmasını ve çalışmasını inceleyelim..

Çözüm:

```
1. Başla
2. N sayısını gir
3. Fak=1
4. S=0
5. Eğer S>N-1 ise git 9
6. S=S+1
7. Fak=Fak*S
8. Git 5
9. Yaz Fak
10. Dur
```

Tablo 2.9 Örnek-2.7 için uygulama sonuçları.

Eski S	Yeni S	Eski Fak	Yeni Fak	Sonuç
0	0+1=1	1	1*1=1	---
1	1+1=2	1	1*2=2	---
2	2+1=3	2	2*3=6	---
3	3+1=4	6	6*4=24	---
4	4+1=5	24	24*5=120	120

Ev Ödevi 1:

1. Klavyeden girilen 1-1000 arasındaki asal sayıları ve kaç adet olduğunu bulup bunları ekrana yazdıran programın algoritmasını maddeler halinde yazınız.
2. Klavyeden 3 kenarı da girilen bir üçgenin çeşidini (eşkenar, ikizkenar, çeşitkenar) bulan programın algoritmasını maddeler halinde yazınız.
3. 1-100 arasındaki
   i. Tüm sayıların toplamını
   ii. Çift sayıların toplamını
   iii. Tek sayıların toplamını bulan programın algoritmasını maddeler halinde yazınız.

Ev Ödevi 1: (devam)

4. Girilen sıcaklık değerine göre bir suyun katı, sıvı ve gaz olma durumunu gösteren algoritmanın akış şemasını çiziniz.
5. Bankaya aylık getirisi %8 ile 1000 TL yatırılmıştır. Buna göre;
   a) 10 ay sonrasında paranın ne kadar olacağını ve aynı zamanda her ay sonunda ne kadar olduğunu ekrana yazan algoritmanın akış şemasını çiziniz.
   b) Paranın kaç ay sonra 5000 TL olacağını hesaplayan algoritmanın akış şemasını çiziniz.
6. Dışarıdan girilen n adet sayının karesinin toplamını alan algoritmanın akış şemasını çiziniz.
7. Dışarıdan girilen 20 adet sayıdan çift olanları toplayan algoritmanın akış şemasını çiziniz.

Programlama Dili Elemanları

- Değişkenler
- Noktalama işaretleri
- Sabitler
- Operatörler

Programlama Dili Elemanları

- Değişkenler; bellekte belli bir yer işgal eden ve içerisinde bir değeri tutan bellek alanlarıdır.

- Değişken Özellikleri
  - İsim
  - Tür
  - Değer

Değişken İsimleri

- İlk karakter harf olmalıdır.
- İlk karakterden sonra alt çizgi yada sayı kullanılabilir.
- Değişken isimlerinde Türkçe harfler bulunmamalıdır.
- Değişken isimlerinde boşluk bulunmamalıdır.
- C dili büyük-küçük harf duyarlı bir dildir.
- Özel karakterler kullanılmamalıdır.

‘ ’ Boşluk	, Virgül	! Ünlem
. Nokta	; Noktalı virgül	: İki nokta
‘ Tırnak	“ Çift tırnak	| Duvar
(Sol parantez	) Sağ parantez	[Sol köşeli parantez
] Sağ köşeli parantez	{ Sol küme	} Sağ küme
/ Bölme (slaş)	\ Ters bölme	~ Tilde
+ Artı	- Eksi	= Eşit
< Küçük	> Büyük	# Diyez
* Yıldız	& Ve	^ Üst ok (karet)

Değişken İsimleri

- Programın kullandığı isimler kullanılmamalıdır.

asm	auto	break
char	const	continue
double	else	enum
float	for	goto
int	interrupt	long
register	return	short
static	struct	switch
unsigned	void	while
_cs	_bp	_sı
_es	_ax	_bh

Değişken İsimleri

- Program yazılırken yapılan tanımlamalar için isimler kullanılır.
- Örnek:
- maas=gunxyevmiye
- ifadesinde maas, gun ve yevmiye birer değişken ismidir.

Değişken İsmi	Açıklama
ogrenci_yasi	
Short	
06_yili_ucreti	Değişken ismi sayı ile başlamaz
Ayse?veli	Değişken ismi özel karakter içermez
short	Değişken ismi özel amaçlı bir sözcük olamaz
ogrenci yasi	Değişken ismi boşluk içermez
Meriç	Değişken ismi Türkçe karakter içermez

Veri Tipleri

Tip	Açıklama	Boyut (Bit)	Aralık
İnt	tamsayı	16 32	-32.768 ..... 32.767 -2.147.483.648…. -2.147.483.647
Float	reel sayı	32	+-3.4E -38 ..... +-3.4E 38 Yaklaşık 6 basamak duyarlı
Double	Çift duyarlıklı reel sayı	64	+-1.7E-308.....+-1.7E308 Yaklaşık 12 basamak duyarlı
short int	Kısa tamsayı	8	-128....+127
char	karakter	8	0.....255
unsigned int	İşaretsiz tamsayı	16	0.....65.535

NOT:1) 8 Bit=1 Byte
2) Sayıların ondalık kısımları nokta ile ayrılır.
3) $3.4e\text{-}38=3.4x10^{-38}$

AÇIKLAMA SATIRLARI

- Kodlarla ilgili açıklamalar yazmak için açıklamanın başına
  - // tek satır için
- Örnek:

  // bu bir C++ programidir .
  - Başına /* sonuna da */
- Örnek:

  /* bu bir C++ programidir*/

  yazmak gereklidir.