

# YAZILIM PROJESİ GELİŞTİRME DERS NOTLARI

Dr. Öğretim Üyesi Yüksel BAL

# CPM UYGULAMA-2

Dr. Öğretim Üyesi Yüksel BAL

## UYGULAMA-2

Bir yazılım projesinde, aşağıdaki tabloda belirtildiği gibi, 9 ayrı faaliyetten oluşan geliştirme çalışmalarının tamamlanması gerekmektedir.

Her bir faaliyet için, tamamlanma zamanları ve gerçekleştirilen faaliyetlerin aralarındaki bağımlılıkları da tabloda verilmiştir.

Bu proje için;

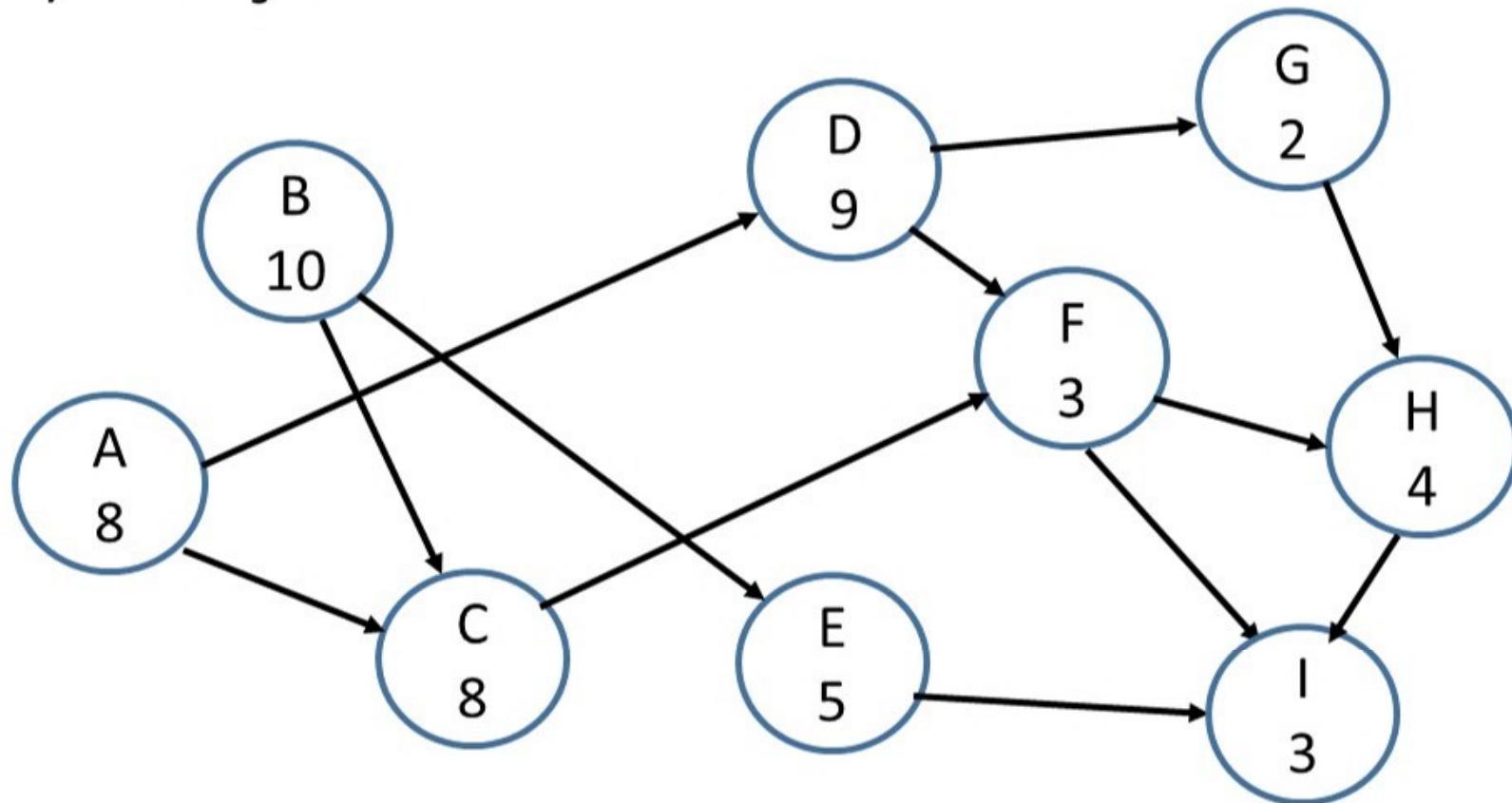
- CPM metodunu kullanarak projenin kritik faaliyetlerini ve kritik yolunu,
- Projenin tamamlanma süresini,
- A faaliyetinde 1 haftalık, F ve G faaliyetlerinde 2'şer haftalık gecikmeler yaşanırsa bu faaliyetlerin proje süresine etkisi nasıl olur?  
Yorumlayınız.

# UYGULAMA-2

## FAALİYET SÜRELERİ VE BAĞIMLILIKLAR TABLOSU

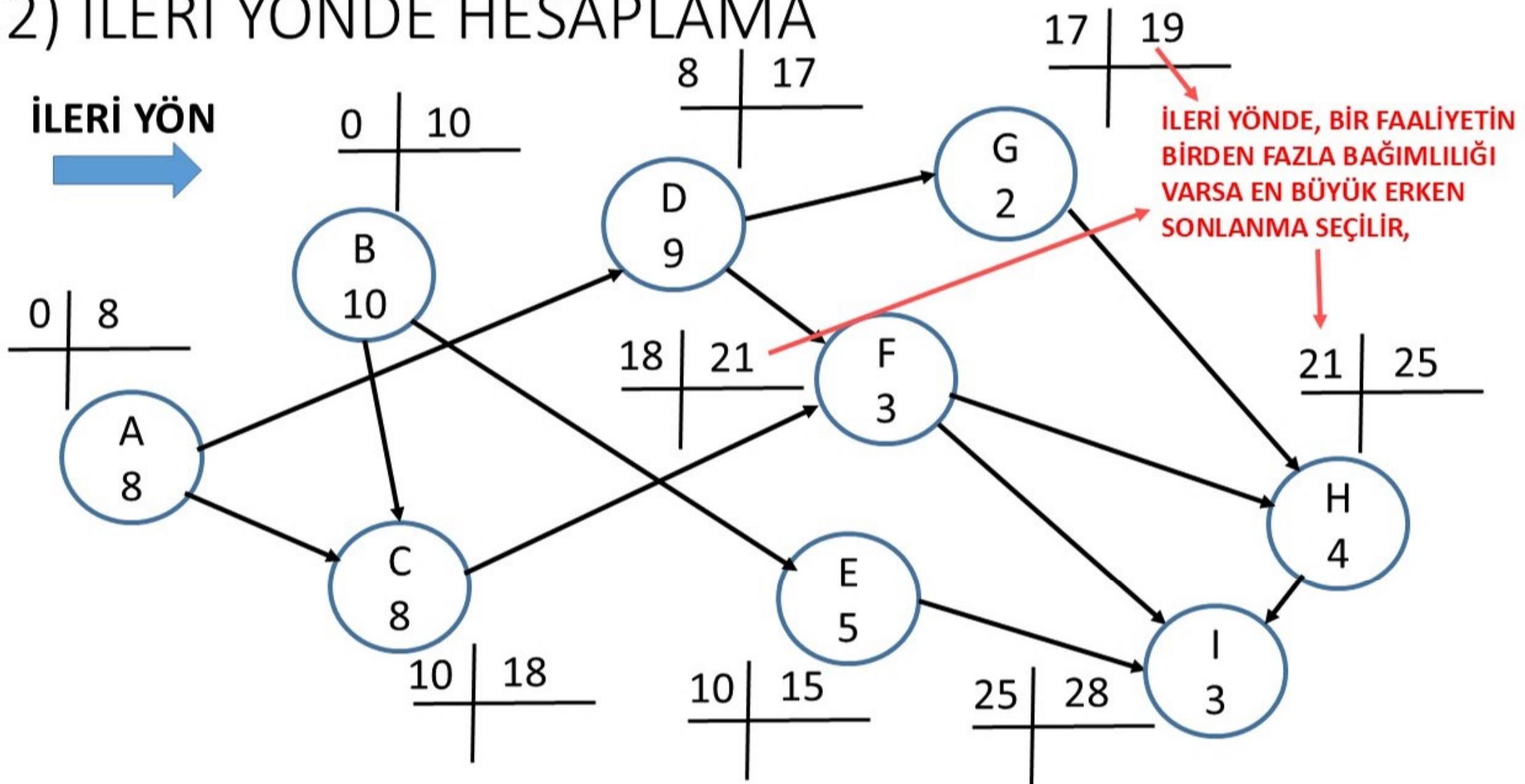
FAALİYET ADI	FAALİYET SÜRESİ (HAFTA)	BAĞIMLILIKLAR (DEPENDENCIES)
A	8	
B	10	
C	8	A, B
D	9	A
E	5	B
F	3	C, D
G	2	D
H	4	F, G
I	3	E, F, H

# 1) AĞ ÇİZİMİ

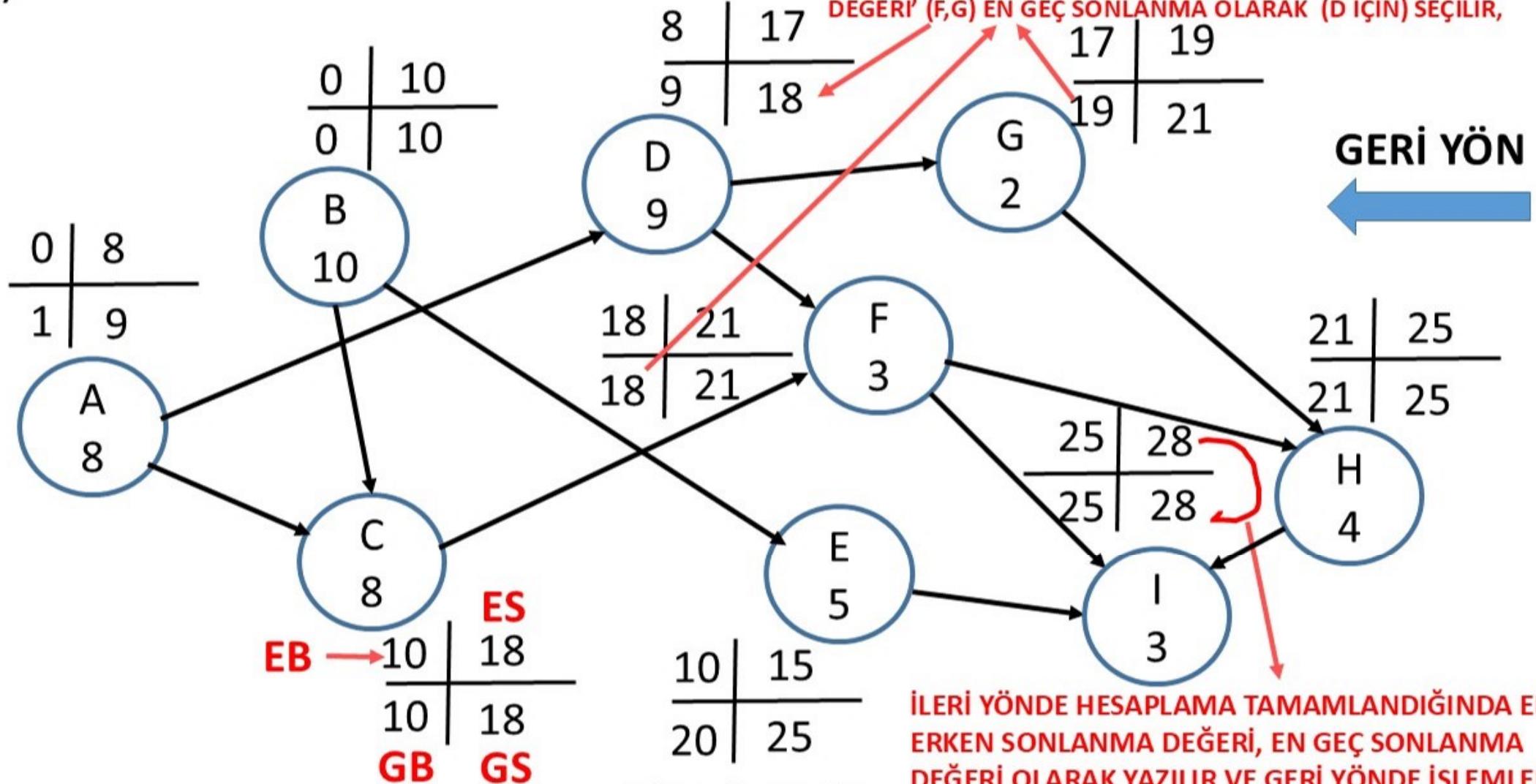


## 2) İLERİ YÖNDE HESAPLAMA

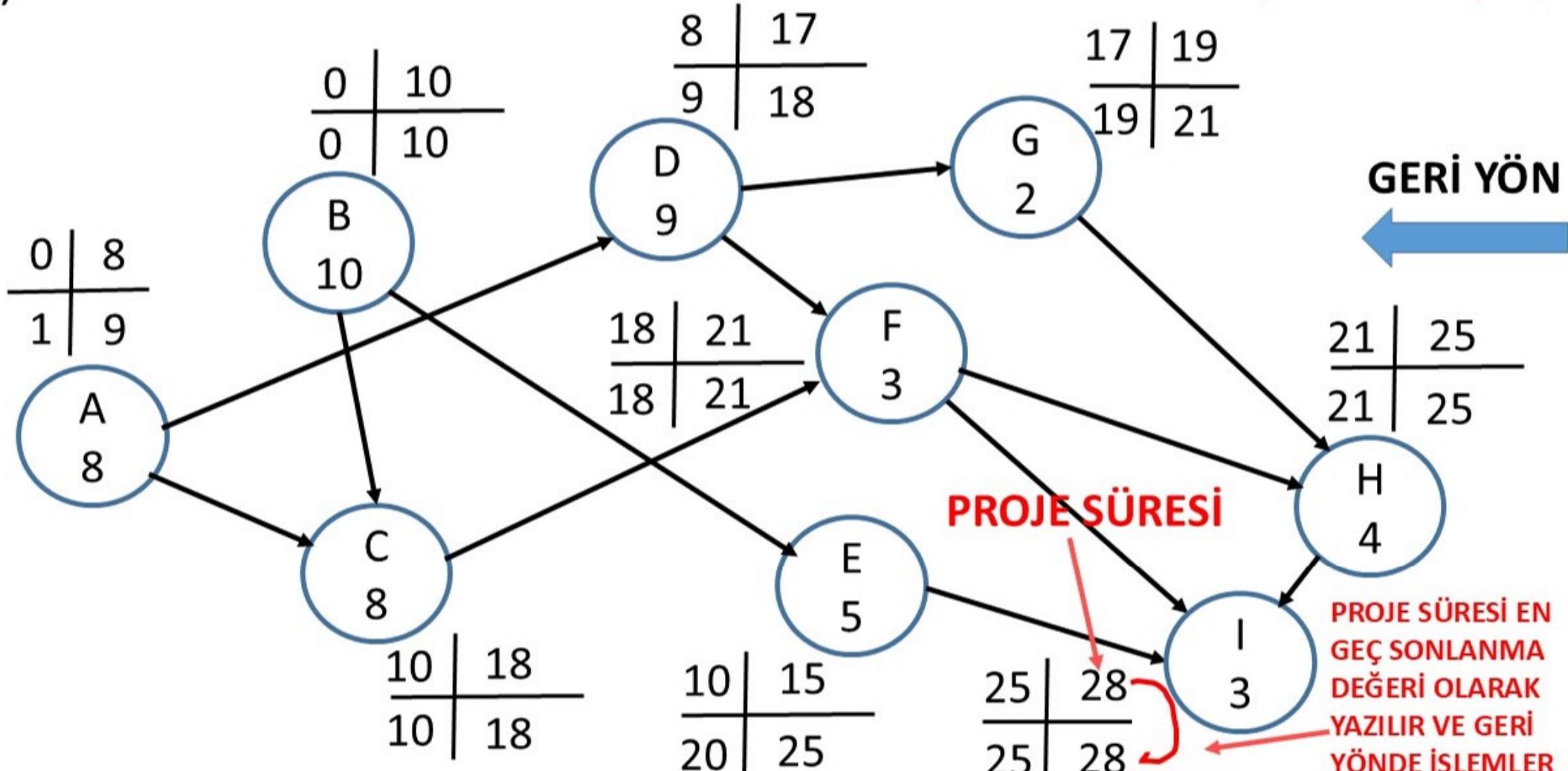
İLERİ YÖN



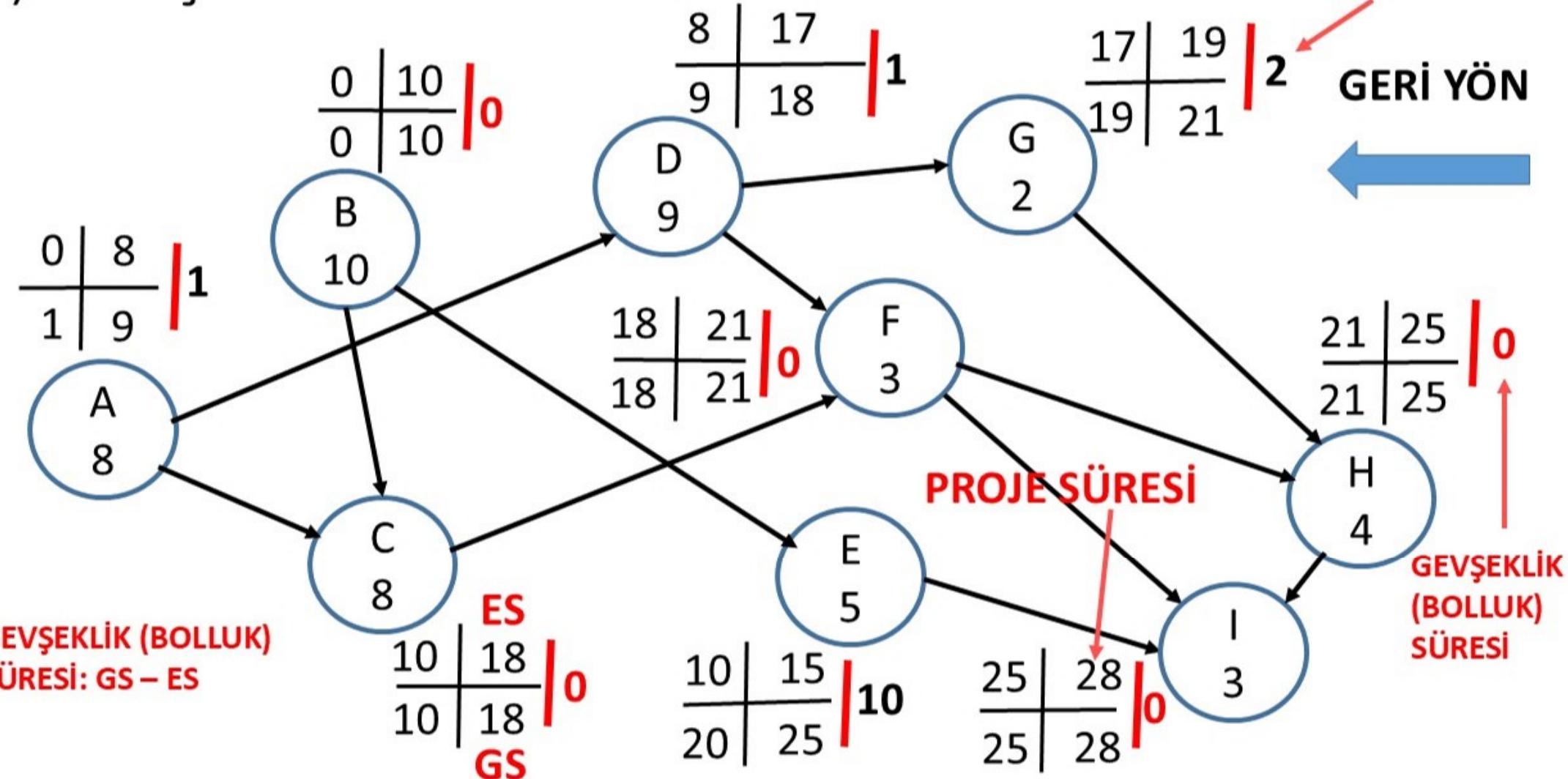
### 3) GERİ YÖNDE HESAPLAMA



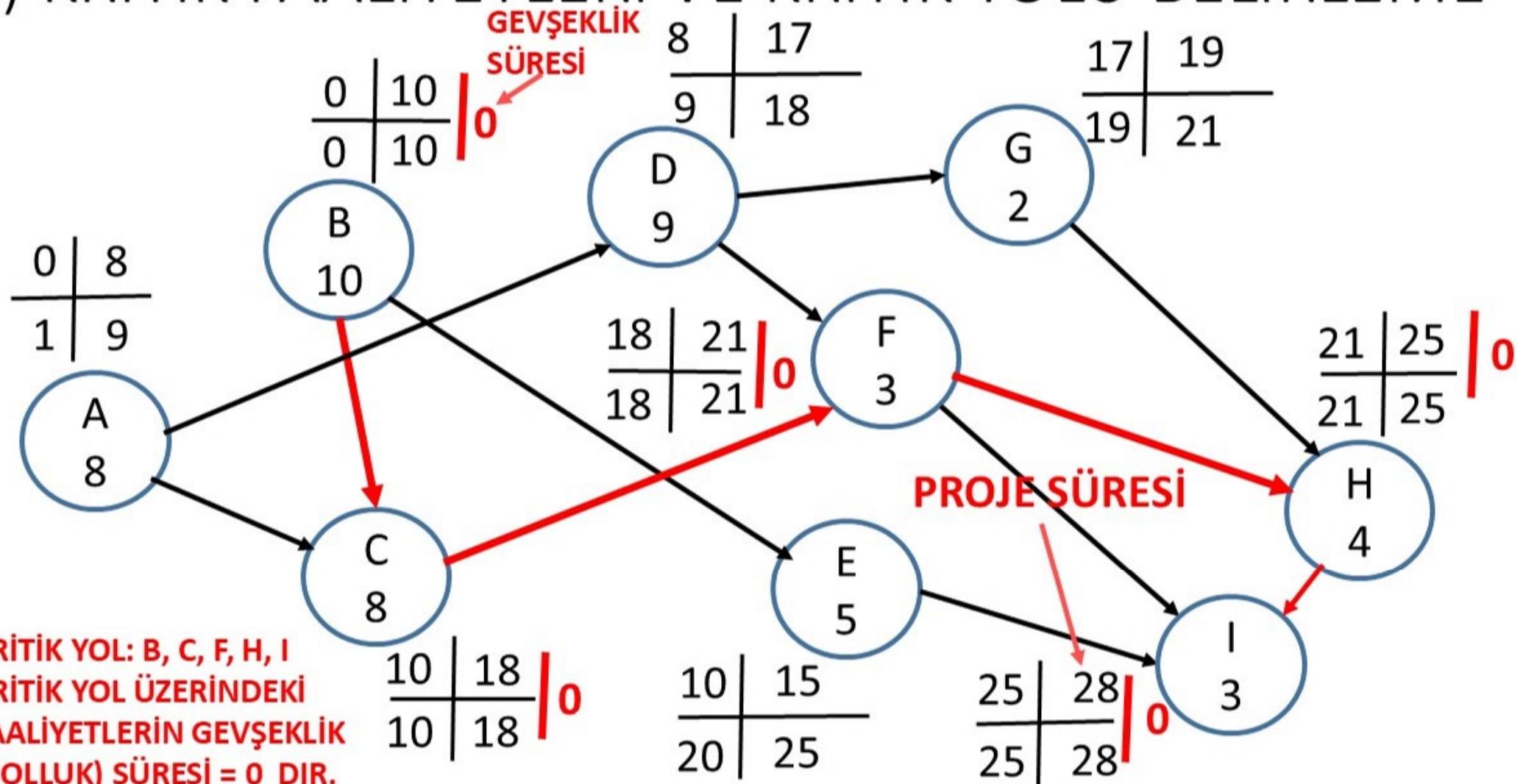
### 3) PROJE SÜRESİ HESAPLAMA



#### 4) GEVŞEKLİK SÜRELERİİNİ BELİRLEME



## 5) KRİTİK FAALİYETLERİ VE KRİTİK YOLU BELİRLEME



## 6) DİĞER DEĞERLENDİRMELER

- Projenin kritik yolu: **B-C-F-H-I** olarak belirlenmiştir. Bu faaliyetlerde esneklik / gevşeklik süresi (bolluk) sıfır olduğundan (kritik yol üzerinde) **hiç gecikme olmamalıdır**.
- A faaliyeti kritik yol üzerinde olmadığından ve 1 **haftalık gevşeklik** (bolluk) zamanına sahip olduğu için 1 haftalık gecikme **proje süresini etkilemeyecektir**.
- G faaliyetindeki 2 haftalık gecikme de bu faaliyetin **gevşeklik süresinin 2 hafta oluşu sebebiyle proje süresini geciktirmeyecektir**.
- Ancak F faaliyetindeki 2 haftalık gecikme bu faaliyetin **kritik yol üzerinde olmasından dolayı** (gevşeklik süresi: bolluk süresi=0) projenin süresi de **2 hafta uzayacaktır**.

# YAZILIM PROJESİ PLANLAMA

Dr. Öğretim Üyesi Yüksel BAL

# MALİYET KESTİRİM YÖNTEMLERİ - SINIFLANDIRMA

Dr. Öğretim Üyesi Yüksel BAL

# MALİYET KESTİRİM YÖNTEMLERİ

Maliyet kestirim modelleri çeşitli biçimlerde sınıflandırma yapılarak guruplandırılabilir. Bu kestirim modelleri aşağıda beş şekilde guruplandırılmıştır.

**Sınıflandırma-1:** Bu sınıflandırmada, yöntemler proje boyut türüne göre guruplandırılır.

- Proje büyüklüğünü (satır sayısı, işlev nokta sayısı vb.) kestiren yöntemler,
- Proje zaman ve iş gücünü kestiren (kişi/ay, ay vb.) yöntemler olarak iki guruba ayrırlırlar.

# MALİYET KESTİRİM YÖNTEMLERİ

**Sınıflandırma-2:** Yöntemler uygulandıkları projelerin büyüklüğüne göre sınıflandırılır.

- Makro yöntemler büyük boyutlu projeler için (ör; 30 kişi / yıl ve daha fazla),
- Mikro yöntemler orta ve küçük boyutlu projeler için kestirimleri içerir.

**Sınıflandırma-3:** Yöntemler uygulanış biçimlerine göre sınıflandırılmaktadır. Buna göre yöntemler;

- Çok yalın düzeyde,
- Orta ayrıntı düzeyinde
- Çok ayrıntı düzeyinde

uygulanabilen yöntemler olarak üç guruba ayrılabilirler.

# MALİYET KESTİRİM YÖNTEMLERİ

**Sınıflandırma-4:** Bu sınıflandırmada yöntemler, projenin değişik aşamalarında kullanılabilirliklerine göre sınıflandırılabilir.

Buna göre yöntemler;

- Planlama ve çözümleme aşamasında kullanılabilecek yöntemler,
  - Tasarım aşamasında kullanılabilecek yöntemler
  - Gerçekleştirim aşamasında kullanılabilecek yöntemler
- olarak üç sınıfta toplanır.

# MALİYET KESTİRİM YÖNTEMLERİ

**Sınıflandırma-5:** Yöntemler, yapılarına göre sınıflandırılabilir.

Buna göre;

- Uzman deneyimine fazlasıyla gereksinim duyan doğrusal kestirim yöntemleri
- Önceki projelerden edinilen geri bildirimleri kullanarak üretilen katsayıları kullanan, genellikle doğrusal olmayan denklemleri kullanan kestirim yöntemleri  
olmak üzere iki gurupta toplanır.

# MALİYET KESTİRİM YÖNTEMLERİ

Yöntemler ayrıntılı olarak incelendiğinde hepsinde **ortak olan özellikler**:

- Projenin uygulandığı aşamaları temel olarak girdi alması,
- Projeye ilişkin çevresel özellikleri girdi olarak alması,
- Doğrusal ya da doğrusal olmayan denklemler kullanması,
- Projeye ilişkin; satır sayısı, işlev nokta sayısı, zaman, iş gücü (kişi / ay), parasal maliyet gibi çıktılar vermesi,

olarak göze çarpar.

Projenin zaman ve bütçe planlaması açısından özellikle yukarıda belirtilen son özellik büyük önem taşımaktadır.

# MALİYET KESTİRİM YÖNTEMLERİ

Algoritmik yöntemler (ör: Constructive Cost Model (COCOMO), Putnam yöntemi, işlev puanı analizi, doğrusal modeller, vs.)

Algoritmik olmayan yöntemler (uzman görüşü, analogi yöntemiyle tahmin, kazanmak için fiyatlama, Parkinson yasası, makine öğrenmesi yöntemleri, vs.)

**Algoritmik yöntemlerde;** geçmiş veriler ve deneyimlerle oluşturulan matematiksel denklemler kullanılır.

**Algoritmik olmayanlarda ise;** benzer geçmiş projelerdeki tecrübe ve gerçekleşen değerlerden faydalанılır.

# MALİYET KESTİRİM YÖNTEMLERİ

Yazdırılacak bir programın değerini saptamak için bir şekilde onu ölçmemiz gereklidir.

Kolaylığı ve doğrudan ölçülebilirliği açısından en fazla kullanılan yazılım ölçme yöntemlerinin temeli satır sayısıdır.

Bir programın büyüklüğü denince ilk akla gelen kaç satırlık kaynak kodu ile üretildiğiidir. Bazen bu, programın karmaşıklığı için de bir ölçüm olarak ifade edilir.

Ancak, benzer işlevi değişik programcılar farklı büyüklükte kodlarla temin edebilirler. Programın satır sayısı büyüklüğü, onun karmaşıklığı hakkında tam doğru bir fikir vermeyebilir.

Sonuçta önemli olan verilen para karşısında ‘ne kadar’ işlevsellik alındığıdır.

# MALİYET KESTİRİM YÖNTEMLERİ

Bu düşünce ile geliştirilen '**işlev puanı**', **satır sayısına karşı bir seçenek** olarak karşımıza çıkmış bir **yazılım ölçüsü** birimidir.

Bu ölçü, satır sayısı gibi açıkça tanımlanmış bir doğrudan ölçüm birimi değilse de yazılımı değerlendirmek için yaygınca bir şekilde daha anlamlı bulunmaktadır.

Bu iki en yaygın temel ölçüden başka yöntemlerle de program karmaşıklığı ve büyülüğü ölçülmektedir.

# MALİYET KESTİRİM YÖNTEMLERİ

Yazılım Ölçümlerinde doğrudan ve dolaylı ölçülebilen büyüklükler vardır.

**Doğrudan ölçülebilen büyüklüklerden** bazıları:

- Zaman
- Satır sayısı
- Maliyet
- İş gücü, Çaba
- Bildirilen hata sayısı
- Çalışan kişi sayısı

# MALİYET KESTİRİM YÖNTEMLERİ

**Dolaylı ölçülebilen değerlerden bazıları:**

- Karmaşıklık
- Kalite
- İşlevsellik
- Güvenilirlik
- Bakım kolaylığı
- Kullanılabilirlik
- Performans
- Güncelleştirilebilirlik

# MALİYET KESTİRİM YÖNTEMLERİ

**Yazılım geliştirme şirketleri;** proje adı, satır sayısı, çaba, maliyet, doküman sayfa sayısı, hata sayısı, bozukluk sayısı, personel sayısı ve başka değerleri de içine alan bilgileri tutmaya çalışırlar.

**Buna dayanarak ileride alacakları projeler için kestirimde bulunurlar.** Bu bilgileri kullanarak personel maliyeti, program satırı maliyeti, hatalılık gibi sonuçları basit matematiksel hesaplamalarla bulabilirler.

# MALİYET KESTİRİM YÖNTEMLERİ

**Bunların yanında;**

- “hata/adam.ay”,
- “satır sayısı/adam.ay”,
- “doküman sayfası/kod satır sayısı”,
- “doküman sayfası/adam.ay”

gibi değerlere de ulaşabilirler.

**Doğrudan kod satır sayısı yerine genellikle daha geçerli olan 1000 satırı  
karşı düşen KLOC birimi**, yazılım ölçümlerinde bir standarttır.

Bir başka önemli nokta da verilen bilgilerin sadece programlama değil, projenin analiz, tasarım, onarım gibi bütünü için hesaplandığıdır.

# YAZILIM PROJELERİNDE MALİYET KESTİRİM YÖNTEMLERİ

Dr. Öğretim Üyesi Yüksel BAL

# Satır Sayısı Yöntemi İle Kestirim

## 1- Satır Sayısı Yöntemi ile Kestirim

Bu yöntemde, proje tahmin edilen alt birimlerine ayrıstırılır. Parçala - yönet stratejisi sonucunda ortaya çıkan, üzerinde tahmin yapılması daha kolay olan daha küçük her birim için satır sayıları önerilir.

Bu kestirimler yapılırken de en küçük, en olası, ve en büyük ihtimaller belirlenip bunlarla bir ortalama işlemi yapılabilir. Bir faaliyet (birim-parça) için tahmin edilecek en küçük satır sayısına “minimum”, en olası satır sayısı tahminine “olası”, ve en büyük tahmin değerine de “maksimum” denenecek olursa, o faaliyet (birim-parça) için satır sayısı kestirimi aşağıdaki gibi hesaplanır:

$$\underline{(\text{minimum} + (4 \times \text{olası}) + \text{maksimum})}$$

# Satır Sayısı Yöntemi İle Kestirim

- Her bir faaliyet (parça) için ayrı ayrı tahminler yapılır ve daha önceki deneyimlerden benzeri birimlerin geliştirilmesindeki değerler de kullanılarak satır sayısı tahminlerinden; çaba, zaman ve maliyet kestirimlerine ulaşılır.
- Faaliyetlerin; çaba, zaman ve maliyet kestirimleri toplanarak projenin bütünü için değerler elde edilir.
- Faaliyetlerin satır sayıları toplanarak proje bütünü hakkında çaba ve zaman gibi kestirim hesaplarını bir defa yapıp kesinleştirmek doğru değildir.
- Satır sayısı büyüklüğü ile diğer sonuç değerlerinin doğrusal olmayan bir ilişki ile bağlantılı oldukları hatırlanmalıdır.

# Satır Sayısı Yöntemi İle Kestirim

Modül (Parça)	En İyi Ölçü	En Olası Ölçü	En Kötü Ölçü
1	20	30	50
2	10	15	25
3	25	30	45
4	30	35	40
5	15	20	25
6	10	12	14
7	20	22	25

$$\frac{(\text{minimum} + (4 \times \text{olasi}) + \text{maksimum})}{6}$$

$$M1 \quad (20 + 4 \times 30 + 50) / 6 = 31,66$$

$$M2 \quad (10 + 4 \times 15 + 25) / 6 = 15,83$$

$$M3 \quad (25 + 4 \times 30 + 45) / 6 = 31,66$$

$$M4 \quad (30 + 4 \times 35 + 40) / 6 = 35$$

$$M5 \quad (15 + 4 \times 20 + 25) / 6 = 20$$

$$M6 \quad (10 + 4 \times 12 + 14) / 6 = 12$$

$$M7 \quad (20 + 4 \times 22 + 25) / 6 = 22,17$$

TOPLAM = **168,32 LOC**

$$\text{STANDART SAPMA} = \sqrt{\frac{(50-20)^2 + (25-10)^2 + (45-25)^2 + (40-30)^2 + (25-15)^2 + (14-10)^2 + (25-20)^2}{6^2}}$$

$$\text{STANDART SAPMA} = (1766/36)^{1/2} = (49,05)^{1/2} = 7$$

LOC (+/-) SS = 161,32 – 175,32 Kod Satır Sayısı Aralığı

Dr. Öğretim Üyesi Yüksel BAL

Küme parantezinin içi varyansı belirtmektedir. Varyansın karekökü SS dir.

# Satır Sayısı Yöntemi İle Kestirim

Verimliliğe etki eden bazı faktörler:

- Yazılım geliştirme ortamı
- Geliştirilen uygulamanın karmaşıklığı
- Takımın yetenekleri ve deneyimi
- Yazılım mühendisliği uygulanmasının niceliği
- Kullanılan programlama dilinin soyutlama düzeyi
- Süreç olgunluğu ve yönetim uygulamaları

# İşlev Noktaları Yöntemi İle Kestirim

## 2- İşlev Noktaları Yöntemi

Bir yazılım projesinde; **girdiler, çıktılar, arayüzler** gibi yapılacak geliştirmenin özellikleri tahmin edilebiliyorsa bu özellikler kullanılarak geliştirilecek sisteme ait bir değer elde edilebilir.

Satır sayısı tekniğinin tersine bu yöntemde bir yazılım birimi için doğrudan büyülüklük tahmini yapma zorluğu yoktur. Aksine, ihtiyaçların belirlenmesi çalışmalarında ortaya çıkabilecek değerler kullanılarak sonuca varılabilir.

# İşlev Noktaları Yöntemi İle Kestirim

İşlev puanı hesaplamak için karmaşıklığa etki eden faktörlerin ağırlıklandırılması ve sonuçta **karmaşıklığa olan etkileri denenerek** bu ağırlıkların ayarlanması şeklinde yöntemler uygulanmıştır. Aşağıda bu yöntemlerle elde edilen işlev noktası başına ortalama satır sayısı kestirimlerini veren bazı çalışmalara ait veriler verilmiştir.

# Bazı Diller İçin İşlev Noktası Başına Ortalama Satır Sayısı

Bu çalışmada işlev noktası başına dillere göre ortalama satır sayısı yandaki tabloda verilmiştir.

Programlama Platformu	Ortalama Satır Sayısı / İN
Assembly dili	300
COBOL	100
FORTRAN	100
Pascal	90
C	90
Ada	70
Nesne-Kökenli Diller	30
4. Kuşak Dilleri	20
Kod Üreticiler	15

# Kuşak Dilleri

- **Birinci Kuşak Diller:** Makine Dilleri veri ve program komutları, 1 ve 0 'larla ifade edilir. Her bilgisayar türünün kendi makine dili vardır. Kodlamanın zor olduğu, hata yapmaya çok açık dillerdir. Bu durum daha kolay yazılabilir ve anlaşılabilir olan assembly dilinin geliştirilmesini sağlamıştır.
- **İkinci Kuşak Diller:** Assembly Dili, günümüzde bu dil çok alçak seviyeli olarak kabul edilir. İngilizce benzeri komutlardan oluşan bir dildir. Assembly dili ile yazılmış bir program, bir çeviriçi (assembler) ile makine diline çevrilerek çalıştırılır. Oldukça anlaşılabilir gözükmeye karşın çok basit bir iş için bile çok sayıda komut yazmak gerekmektedir. Kullanımı çok daha kolay olan dillerin ortaya çıkışması olması nedeniyle, sadece donanım tasarımcıları tarafından makine diline çevrim öncesi aşamada yoğunlukla kullanılmaktadır.

# Kuşak Dilleri

- **Üçüncü Kuşak Diller:** Bunlara Yüksek Seviyeli Diller'de denir. 1960'larda yaygınlaşmıştır. İngilizceye benzer ifadelerin kullanıldığı, aritmetik işlemlerde kullandığımız işaretlerin kullanıldığı, daha hızlı ve kolay program yazmayı sağlayan, programcının daha kolay program yazabildiği dillerdir.  
Çok sayıda dil geliştirilmiştir. Bunlar; Pascal, Fortran, Cobol, Algol, PL/I, C, C++, Java vb. Bu dillerin ifadeleri derleyici tarafından makine diline çevrilirler. Bu kuşak dillere yordamsal (procedural) diller denilmektedir. Bunun nedeni bu dillerle çözümün nasıl yapılacağıının ayrıntılı olarak belirtilmesidir.

# Kuşak Dilleri

- **Dördüncü Kuşak Diller:** Bu kuşak dillere 4GL (Fourth Generation Languages) denir. Çok Yüksek Seviyeli Programlama dilleridir. 3. kuşak dillerde yüzlerce satır halinde kodlanan bir program, 4.kuşak dillerle birkaç satırda ifade edilir. Çünkü bu dillerle sadece ne yapılmak istediği belirtilir, nasıl yapılacağı ile ilgilenilmez. Dolayısıyla bu dillere yordamsal olmayan diller (non-procedural) denir. Bu dillere örnek olarak; SQL, SPS, pHp, Asp, ABAP, PL/SQL, PostScript, Informix 4GL, Progress 4GL, Borland Delphi gibi dilleri verilebilir.
- **Beşinci Kuşak Diller:** Bu kuşak dillerde amaç tamamen doğal dillerin özelliklerini taşıyan programlama dilleri üretmektir. En önemli örnek PROLOG'dur. OPS5, Mercury de 5. kuşak dillere örnektir. Bu diller yapay zeka ve robotik çalışmalarında önemli bir yer almaktadır.

# İşlev Noktaları Yöntemi İle Kestirim

## Sistem Bilgi Ortamının İncelenmesi

Ölçüm Parametresi	Sayı	Ağırlık Faktörü			=	
		Yalın	Ortalama	Karmaşık		
Kullanıcı Girdi sayısı	X	3	4	6	=	
Kullanıcı Çıktı sayısı	X	4	5	7	=	
Kullanıcı Sorğu Sayısı	X	3	4	6	=	
Kütük Sayısı	X	7	10	15	=	
Dışsal Arayüz Sayısı	X	5	7	10	=	
Toplam Sayı (AİN)					=	XYZ

# İşlev Noktaları Yöntemi İle Kestirim

## Sistem Bilgi Ortamının İncelenmesi

- *Kullanıcı girdileri*

Yazılıma girdi olarak verilen her farklı uygulama bileşeni bir kullanıcı girdisi olarak sayılır.

Kullanıcı girdilerinin sayısı hesaplanırken **alan bazında değil daha genel olarak mantıksal kayıt bazında** uygulama yapılmalıdır.

Örneğin, bir personel sisteminin geliştirilmesinden söz edildiğinde; personel sicil bilgileri, personel izin bilgileri gibi bilgiler ayrı kullanıcı girdileri olarak sayılabilir. Personel sicil no, personel adı soyadı türündeki **alan bilgileri kullanıcı girdisi olarak sayılmamalıdır.**

# İşlev Noktaları Yöntemi İle Kestirim

- *Kullanıcı çıktıları*

Kullanıcıyı ilgilendiren her tür mantıksal çıktı, kullanıcı çıktısı olarak sayılmalıdır. Örnek olarak; **raporlar, ekran çıktıları, hata iletileri** vb. verilebilir.

Girdiler gibi, **çıktılarda genel olarak sayılmalıdır**. Bir raporun içerisindeki bir alan kullanıcı çıktısı olarak sayılmamalıdır. Öte yandan, aynı bilgileri içeren ancak bilgi düzeni ya da sıralaması farklı olan raporlar da ayrı kullanıcı çıktısı olarak değerlendirilmemelidir. **Örneğin soyadına göre sıralanmış personel listesi ve sicil nosuna göre sıralanmış personel listesi raporları farklı kullanıcı çıktıları olarak sayılmamalıdır.**

# İşlev Noktaları Yöntemi İle Kestirim

- *Kullanıcı sorguları*

Kullanıcı sorgusu, çevrim içi olarak bilgisayara verilen bir girdi sonucu yine çevrim içi olarak bir kullanıcı çıktısı alınması biçiminde tanımlanmaktadır. **Her farklı sorgu ayrı olarak sayılmalıdır.** Örneğin; **personel sicil bilgilerinin sorgulanması, personel maaş bilgilerinin sorgulanması ayrı ayrı değerlendirilmelidir.**

- *Kütükler*

**Her mantıksal bilgi yiğini ya da kütük ayrı olarak sayılmalıdır. Personel sicil kütüğü, personel maaş kütüğü, personel hareket kütüğü vb.**

# İşlev Noktaları Yöntemi İle Kestirim

- *Dışsal arayüzler*

Geliştirilmesi öngörülen bilgi sisteminin, **gerek kurum içinde gerekse kurum dışında bir başka bilgi sistemi ile bilgisayar ortamında bir iletişimi söz konusu ise bu durum bir dışsal ara yüz olarak sayılmalıdır.**

Kağıt ya da rapor türündeki arayüzler bir dışsal arayüz sayılmazlar. Örneğin, personel bilgi sistemi, yine aynı kurumda olan muhasebe bilgi sistemine, personel maaşlarına ilişkin bilgiyi bilgisayar ortamında aktarması bir dışsal arayüz olarak sayılabilir.

Sistem bilgi ortamı için elde edilen değerler, karmaşıklık düzeylerine göre yukarıdaki tablodan seçilecek ağırlık faktörü ile çarpılır ve çarpımların sonucu elde edilen toplam işlev nokta sayısı, ayarlanmamış işlev nokta sayısı (AİN) olarak adlandırılır.

# İşlev Noktaları Yöntemi İle Kestirim

## Sistem Teknik Karmaşıklığının İncelenmesi

- Sistem teknik karmaşıklık faktörü (TKF) aşağıdaki tabloda belirtilen ve sistemin karmaşıklığına ilişkin sorulara verilecek cevapların toplamıyla elde edilir. Uygulama ortamının ve uygulamanın özellikleri dikkate alınarak her soruya 0 ile 5 arasında bir cevap verilir.
- Örneğin, sekizinci soru dikkate alındığında; eğer uygulama ana veri tabanının tümüyle çevrim içi olarak güncellenmesini gerektiriyorsa bu durumda cevap 5 olacaktır. Ancak, bazı veri tabanı işlemleri (örneğin %10 gibi) çalışma saatleri dışında çevrim dışı olarak yapılıyorsa bu durumda 4 gibi bir cevap verilebilir.

# İşlev Noktaları Yöntemi İle Kestirim

## TKF HESAPLAMAK İÇİN SORULAR

1. Uygulama, güvenilir yedekleme ve kurtarma gerektiriyor mu?
2. Veri iletişim gerekiyor mu?
3. Dağıtık işlem ve süreçler var mı?
4. Performans kritik mi?
5. Sistem mevcut ve fazla yüklü bir ortamda mı çalışacak?
6. Sistem çevrim içi veri girişi gerektiriyor mu?
7. Çevrim içi veri girişi, bir ara işlem için birden çok ekran gerektiriyor mu?
8. Ana kütükler çevrim içi olarak mı güncelleniyor?
9. Girdiler, çıktılar, kütükler ya da sorgular karmaşık mı?
10. İç süreç karmaşık mı?

# İşlev Noktaları Yöntemi İle Kestirim

11. Tasarlanacak kod yeniden kullanılabilir mi olacak?
12. Dönüştürme ve kurulum, tasarımında dikkate alınacak mı?
13. Sistem birden çok yerde yerleşik farklı kurumlar için mi geliştiriliyor?
14. Tasarlanan uygulama, kolay kullanılabilir ve kullanıcı tarafından kolayca değiştirilebilir mi olacak?

## Cevap kılavuzu:

- |                               |                             |   |
|-------------------------------|-----------------------------|---|
| <b>0:</b> Hiçbir etkisi yok   | <b>1:</b> Çok az etkisi var | <b>2:</b> Etkisi var                    |
| <b>3:</b> Ortalama etkisi var | <b>4:</b> Önemli etkisi var | <b>5:</b> Mutlaka olmalı,<br>kaçınılmaz |

# İşlev Noktaları Yöntemi İle Kestirim

## İşlev Nokta Sayısı Hesaplama

Gerek AİN (Ayarlanmamış İşlev Nokta Sayısı), gerekse TKF önceki bölümlerde açıklandığı gibi hesaplandıktan sonra, geliştirilecek bilgi sistemine ilişkin *işlev nokta sayısı* (İN) aşağıdaki formül ile hesaplanır.

$$\text{İN} = \text{AIN} \times (0.65 + 0.01 \times \text{TKF})$$

İşlev nokta sayısı çeşitli amaçlarla kullanılabilir.

$$\text{Üretkenlik} = \text{İN} / \text{Kişi-ay}$$

$$\text{Kalite} = \text{Hatalar} / \text{İN}$$

$$\text{Maliyet} = \$ / \text{İN}$$

# İşlev Noktaları Yöntemi İle Kestirim

İşlev noktaları, başka yöntemlerle birlikte kullanılmak istenildiğinde, kullanılacak yazılım geliştirme platformuna göre aşağıdaki değerler kullanılarak kolayca satır sayısı kestirimine dönüştürülebilir.

Örneğin, İN kestirimi olarak 300 İN (İşlev Nokta Sayısı) bulduğumuzu varsayıyalım. Eğer nesne kökenli bir dil kullanılarak yazılımı geliştirecek isek yaklaşık satır sayısı kestirimi: **300 x 30 = 9000 satır** olacaktır.

**NOT:** Aşağıdaki tablodan Nesne Kökenli Diller için ilgili değer 30 olarak alınmıştır.

# İşlev Noktaları Yöntemi İle Kestirim

Programlama Platformu	Ortalama Satır Sayısı / İN
Assembly dili	300
COBOL	100
FORTRAN	100
Pascal	90
C	90
Ada	70
Nesne-Kökenli Diller	30
4. Kuşak Dilleri	20
Kod Üreticiler	15

# **Yapısal/Etkin Maliyet Modeli - COCOMO**

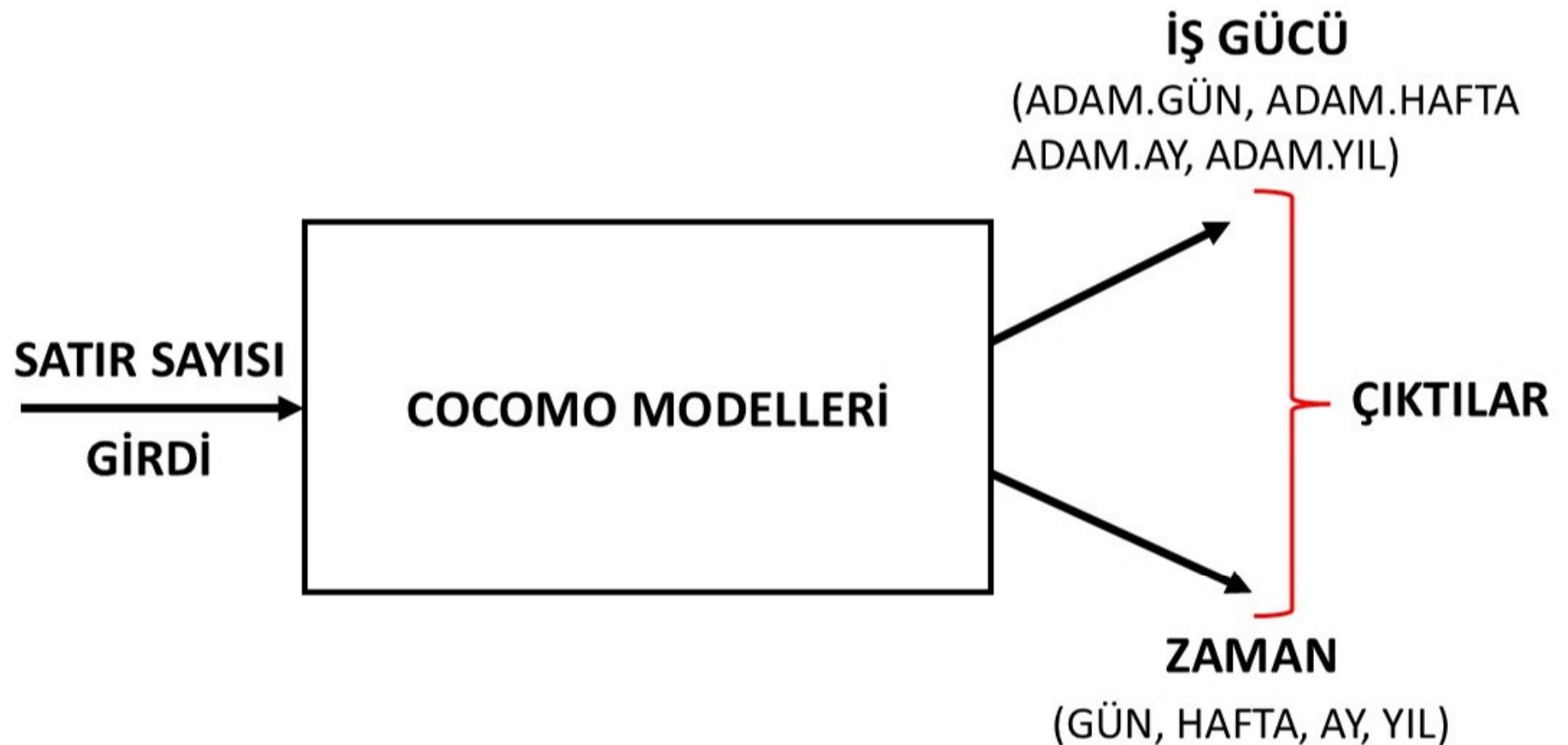
## **Etkin maliyet modeli - COCOMO**

COCOMO, 1981 yılında Boehm tarafından yayımlandıktan sonra oldukça ilgi görmüş bir maliyet kestirim modelidir. Uygulama, kullanılacak ayrıntı düzeyine göre üç ayrı model biçiminde yapılabilir:

- Temel model
- Ara model
- Ayrıntı model

Tüm COCOMO modelleri; temel girdi olarak satır sayısı kestirimini alır ve çıktı olarak iş gücü (Kişi-ay, kişi-yıl vb) ve zaman (gün, hafta, ay, yıl vb) çıktılarını verir.

# Yapısal/Etkin Maliyet Modeli - Cocomo



## Yapısal/Etkin Maliyet Modeli - Cocomo

İş gücü değerinin zaman değerine bölünmesiyle yaklaşık kişi sayısı kestirimi de elde edilmiş olur. Tüm COCOMO modelleri gerek iş gücü, gerekse zaman için **doğrusal olmayan üstel formüller** kullanılır.

İş gücü (K)       $K = a \times S^b$     MM (Man-Month)

Zaman (T)       $T = c \times K^d$     Month

a, b, c, d : her bir model için farklı olan katsayılar

S: Satır sayısı

## Yapısal/Etkin Maliyet Modeli - Cocomo

Diğer taraftan COCOMO modelleri farklı türdeki yazılım projelerinde farklı biçimde uygulanır. Projeler üç biçimde sınıflandırılmaktadır:

- Ayrık projeler
- Yarı gömülü projeler
- Gömülü projeler

**I) Ayrık projeler:** Deneyimli ve küçük proje ekipleri tarafından geliştirilecek, boyutları küçük (bir kaç bin satırdan, bir kaç on bin satıra kadar) bilgi sistemi projeleridir. Bu tür projelerde, bir donanımın yazılım tarafından sürülmesi ya da yönetilmesi söz konusu değildir. Genellikle, yerel ağ üzerinde çalışan girdi ve çıktıları tümüyle veri bilgi tabanlı olan projelerdir. Veri işleme / güncelleme uygulamaları, bilimsel projeler, bir kurumun yerel ağ üzerinde çalışan çeşitli yazılımlar gibi yazılımlardır.

## **Yapısal/Etkin Maliyet Modeli - Cocomo**

- II) Yarı-gömülü projeler:** Hem bilgi boyutu hem de donanım sürme boyutu olan yazılım projeleridir.
- III) Gömülü projeler:** Donanım sürmeyi hedefleyen (sureç denetimi, plotsuz uçağı yönlendiren yazılım projeleri vb.) ve donanım kısıtları yüksek projelerdir.

# **Yapısal/Etkin Maliyet Modeli - Cocomo**

## **A) Temel model**

Temel model, küçük ve orta boy projeler için hızlı kestirim yapmak amacıyla kullanılır. Yalnızca, iki COCOMO formülünün kullanılmasından ibarettir.

K= İş gücü, T= Zaman olmak üzere;

# Yapısal/Etkin Maliyet Modeli - Cocomo

*Ayrik projeler için:*

$$K = 2.4 \times S^{1.05}$$

$$T = 2.5 \times K^{0.38}$$

*Yarı gömülü projeler için:*

$$K = 3.0 \times S^{1.12}$$

$$T = 2.5 \times K^{0.35}$$

*Gömülü projeler için:*

$$K = 3.6 \times S^{1.20}$$

$$T = 2.5 \times K^{0.32}$$

Yandaki formüllerden de görüleceği gibi, proje karmaşıklığı arttıkça, gerekecek iş gücü de artmaktadır. Temel modelin aksak yönlerinden biri, yazılım projesinin geliştirileceği ortam ve yazılımı geliştirecek ekibin özelliklerini dikkate almayışıdır. Bununla birlikte, bir hesap makinesi yardımıyla kolayca uygulanabilme özelliğine sahiptir.

# **Yapısal/Etkin Maliyet Modeli - Cocomo**

## **B) Ara model**

Ara model, temel modelin eksikliğini gidermek üzere oluşturulmuştur. Bir yazılım projesinin zaman ve iş gücü maliyetlerinin kestiriminde, proje ekibinin özellikleri, projenin geliştirilmesinde kullanılacak araç, yöntem ve ortamı dikkate almak gerekliliği üzerine kuruludur.

Örneğin, aynı yazılımı yeni mezun ve deneyimsiz bir ekip ile geliştirmekle, deneyimli bir proje ekibi ile geliştirmek arasında zaman ve iş gücü açısından oldukça farklılıklar vardır.

Ara model üç aşamalı olarak uygulanır:

# Yapısal/Etkin Maliyet Modeli - Cocomo

- İş gücü hesaplama
- Maliyet çarpanı hesaplama
- İlk iş gücü değerlerini düzeltme

## İş gücü hesaplama:

Ara modelde bu aşamada kullanılacak iş gücü hesaplama formülleri yanda verilmiştir.

*Ayrik projeler:*

$$K = 3.2 \times S^{1.05}$$

*Yarı gömülü projeler:*

$$K = 3.0 \times S^{1.12}$$

*Gömülü projeler:*

$$K = 2.8 \times S^{1.20}$$

# Yapısal/Etkin Maliyet Modeli - Cocomo

## Maliyet çarpanı hesaplama:

Maliyet çarpanı ( $C$ ), aşağıdaki tabloda belirtilen değerlerden seçilecek 15 maliyet etmeninin çarpımı sonucu elde edilir.

$$C = \prod_{i=1}^{15} c_i = C_1 \times C_2 \times C_3 \times \dots \times C_{15}$$

**MALİYET ÇARPANI** → **MALİYET ETMENLERİ**

Herbir maliyet etmeninin seçilmesinde yardımcı olacak bilgiler ve açıklamalar aşağıda verilmiştir.

# Yapısal/Etkin Maliyet Modeli - Cocomo

MALİYET ETMENLERİ	SEÇENEKLER					
	Çok düşük	Düşük	Normal	Yüksek	Çok yüksek	Oldukça yüksek
<b>Ürün özellikleri</b>						
RELY	0.75	0.88	1.00	1.15	1.40	-
DATA	-	0.94	1.00	1.08	1.16	-
CPLX	0.70	0.85	1.00	1.15	1.30	1.65
<b>Bilgisayar Özellikleri</b>						
TIME	-	-	1.00	1.11	1.30	1.66
STOR	-	-	1.00	1.06	1.21	1.56
VIRT	-	0.87	1.00	1.15	1.30	-
TURN	-	0.87	1.00	1.07	1.15	-
<b>Personel Özellikleri</b>						
ACAP	1.46	1.19	1.00	0.86	0.71	-
AEXP	1.29	1.13	1.00	0.91	0.82	-
PCAP	1.42	1.17	1.00	0.86	0.70	-
VEXP	1.21	1.10	1.00	0.90	-	-
LEXP	1.14	1.07	1.00	0.95	-	-
<b>Proje Özellikleri</b>						
MODP	1.24	1.10	1.00	0.91	0.82	-
TOOL	1.24	1.10	1.00	0.91	0.83	-
SCED	1.23	1.08	1.00	1.04	1.10	-

Dr. Öğretim Üyesi Yüksel BAL

## **Yapısal/Etkin Maliyet Modeli - Cocomo**

**RELY:** Gereken yazılım güvenilirliği, yazılımın işletimi sırasında ortaya çıkabilecek bir hatanın kullanıcıya olası etkileri dikkate alınarak seçilir. Örneğin, yazılım hatası geliştiriciler tarafından kısa sürede giderilebilecek bir hata ise “çok düşük”, insan yaşamına mal olacak kadar etkili ise “çok yüksek” seçeneği seçilir.

**DATA:** Veri tabanı büyülüğündür. Data maliyet etmeninin seçiminde, yazılım tarafından oluşturulacak veri tabanı büyülüğünün program büyülüğüne oranı dikkate alınır.  $10/100/1000/10000$  kat oranlarına göre “düşük” ile “çok yüksek” arasında seçim yapılır.

**CPLX:** Ürün karmaşıklığıdır. Geliştirilecek yazılım ürününün karmaşıklığı genel olarak dikkate alınarak seçim yapılır.

## **Yapısal/Etkin Maliyet Modeli - Cocomo**

**TIME:** İşletim zamanı kısıtıdır. Geliştirilecek yazılımın işletileceği bilgisayarın zaman kaynaklarının oranına göre seçim yapılır. Örneğin sözkonusu oran %50 ise “normal” , %5 ise “oldukça yüksek” seçeneği seçilir.

**STOR:** Ana bellek kısıtıdır. TIME maliyet etmenindeki örneklemeye yazılımın işletileceği bilgisayarın ana bellek kullanım kısıtları dikkate alınarak yapılır.

**VIRT:** Bilgisayar platform değişim olasılığıdır. Yazılımın geliştirildiği bilgisayar platformu ile ilgili olası değişiklikler dikkate alınarak seçim yapılır. Ana bellek ya da disk kapasitelerinin arttırımı, bilgisayarın bir üst modele yükseltimi, kapalı sistemlerden açık sistemlere geçiş gibi durumlar dikkate alınarak “düşük” ile “çok yüksek” arasında seçim yapılır.

# Yapısal/Etkin Maliyet Modeli - Cocomo

**TURN:** Bilgisayar iş geri dönüş zamanıdır. Programcıya yönelik olarak işin geri dönüş zamanı dikkate alınarak seçim yapılır. Örneğin, etkileşimli sistemler için “düşük” geri dönüş süresi 12 saatten fazla olan sistemler için “çok yüksek” seçenekleri seçilebilir.

**ACAP:** Çözümleyici yeteneğidir. Çözümleyici ekibin deneyimi, birlikte çalışabilirliği iş gücü maliyetlerini oldukça etkilemektedir. Yeni mezun kişilerden oluşan bir ekip için “düşük”, ortalama deneyimleri 5 yıldan fazla olan bir ekip için “çok yüksek” seçeneği seçilebilir.

**AEXP:** Uygulama deneyimidir. Proje ekibinin ortalama uygulama deneyimi dikkate alınarak seçim yapılır. 4 aydan daha az süreli deneyimler için “çok düşük”, 12 yıldan fazla deneyimler için “çok yüksek” seçenekleri seçilebilir.

# Yapısal/Etkin Maliyet Modeli - Cocomo

**PCAP:** Programcı yeteneğidir. ACAP maliyet etmenindeki açıklamalar programcılar için dikkate alınarak seçim yapılır. Sözkonusu seçim genel yapılmalı, tek tek programcılar için yapılmamalıdır.

**VEXP:** Bilgisayar platformu deneyimidir. Proje ekibinin geliştirme için kullanılacak bilgisayar platformunu tanıma oranı ile ilgilidir. Bir aydan daha az deneyimler için “çok düşük” , üç yıldan daha fazla deneyimler için “yüksek” seçenekleri seçilebilir.

**LEXP:** Programlama dili deneyimidir. Programlama ekibinin, kullanılacak yazılım geliştirme platformuna ilişkin deneyimi dikkate alınarak seçim yapılır. Bir aydan daha az deneyimler için “çok düşük” , üç yıldan daha fazla deneyimler için “yüksek” seçenekleri seçilebilir.

# Yapısal/Etkin Maliyet Modeli - Cocomo

**MODP:** Modern programlama teknikleridir. Proje ekibi tarafından modern programlama tekniklerinin (yapısal programlama, yukarıdan aşağıya geliştirme, yapısal metodolojiler vb.) kullanım oranı dikkate alınarak, “çok düşük” ile “çok yüksek” arasında seçim yapılır.

**TOOL:** Yazılım geliştirme araçları kullanımıdır. Proje ekibi tarafından bilgisayar destekli yazılım geliştirme araçlarının (CASE araçları, metin düzenleyiciler, ortam yönetim araçları vb.) kullanım oranı dikkate alınarak “çok düşük” ile “çok yüksek” arasında seçim yapılır.

**SCED:** Zaman kısıtıdır. Projedeki geliştirme zamanı kısıtı dikkate alınarak seçim yapılır. Zaman kısıtının çok az ya da çok fazla olmasının, proje maliyetlerini olumsuz olarak etkilediğine dikkat edilmelidir.

# **Yapısal/Etkin Maliyet Modeli - Cocomo**

## **İlk iş gücünü değerlerini düzeltme:**

Maliyet çarpanı ( $C$ ), ikinci aşamada elde edildikten sonra, ilk aşamada elde edilen iş gücü değeri ( $K$ ) ile çarpılarak düzeltilmiş iş gücü değeri  $K_d$  elde edilir.

Elde edilen düzeltilmiş iş gücü değeri, temel modelde kullanılan zaman formüllerinden ilgili olanı kullanılarak zaman maliyetleri hesaplanır.

$$K_d = K \times C$$

# **Yapısal/Etkin Maliyet Modeli - Cocomo**

## **C) Ayrıntı modeli**

Ayrıntı modeli, temel ve ara modellere ek olarak iki ek özellik taşımaktadır.

- 1) Aşama ile ilgili iş gücü katsayıları: Yazılım üretiminde her aşama (planlama, çözümleme, tasarım, geliştirme, test, kurulum) eşdeğer karmaşıklık düzeyine sahip değildir. Ayrıntı modelde her aşama farklı olarak ağırlıklandırılmıştır.
- 2) Üç düzey ürün sıra düzeni: Ayrıntı model, yazılım maliyetlerinin kestiriminde modül-alt modül-sistem sıra düzenini dikkate almayı gerektirir. Yani kestirimler, önce modül bazında, daha sonra alt sistem bazında ve en sonunda da tüm sistem bazında yapılır.

# ÖRNEK-1

Dr. Öğretim Üyesi Yüksel BAL

## ÖRNEK-1:

Bir yazılım projesinde (ayrık proje), “müşteri talepleri analizi” sonrasında planlama aşamasında aşağıdaki analizler yapılmış olup, yazılımlar nesne kökenli diller ile geliştirilecektir. Maliyet etmenleri aşağıda Tablo-1 de verilmiştir ve bu yazılım projesinde  $TKF=42$  olarak tespit edilmiştir.

- a) Yazılım satır sayısı kestirimini yapınız
- b) COCOMO yöntemi ile ara modeli kullanarak; düzeltilmiş iş gücünü,
- c) Projenin / Faaliyetin süresini ve ortalama personel sayısını hesaplayınız.

	Sayı	Ağırlık Faktörü
Kullanıcı girdi sayısı	:10	4
Kullanıcı çıktı sayısı	:20	4
Kullanıcı sorgu sayısı	:50	3
Kütük sayısı	:5	15
Dışsal arayüz sayısı	:3	5

Tablo-1

MALİYET ETMENLERİ	ÇOK DÜŞÜK	DÜŞÜK	NORMAL	YÜKSEK	ÇOK YÜKSEK	OLDUKÇA YÜKSEK
<b>Ürün Özellikleri</b>						
RELY		0,88				
CPLX				1,15		
<b>Personel Özellikleri</b>						
AEXP		1,13				
LEXP				0,95		

**NOT: Diğer maliyet etmenleri “normal” yani 1 değerindedir.**

## ÇÖZÜM:

### a) İşlev noktaları yöntemi ile satır sayısı kestirimi:

$$AİN = 40+80+150+75+15 = 360,$$

TKF = 42 (14 Sorunun cevabına göre elde edilmiş Teknik Karmaşıklık Faktörü)

$$İN = AİN \times (0.65 + 0.01 \times TKF)$$

$$İN = 360 \times (0.65 + 0.01 \times 42) = 385 \text{ (İşlev Nokta Sayısı)}$$

$$SS = 385 \times 30 = 11550 \text{ (Satır Sayısı)}$$

NESNE KÖKENLİ DİLLER İÇİN 'İN'  
BAŞINA ORTALAMA SATIR SAYISI

### b) COCOMO ile İşgücü, maliyet çarpanı ve düzeltilmiş işgücünün bulunması:

#### İşgücü:

$$K = 3,2 \times S^{1.05}$$

SS; FORMÜLDE (SS/1000) ALINIR.

$$K = 3,2 \times (11,55)^{1.05} = \mathbf{41,77 \text{ MM (Adam.Ay)}}$$

## Maliyet Çarpanı;

$$C = \prod_{i=1}^{15} C_i = C_1 \times C_2 \times C_3 \times \dots \times C_{15} \text{ ve Tablo-1 den;}$$

$$C = 0.88 \times 1.15 \times 1.13 \times 0.95 = \mathbf{1.086}$$

NOT: Diğer maliyet etmenleri “normal=1” alınmıştır.)

Düzeltilmiş iş gücü;  $K_d = K \times C$  den;

$$K_d = 41,77 \times 1,086 = \mathbf{45,36 \text{ MM}}$$

## c) COCOMO'da projenin süresi ve ortalama personel sayısı:

$$\text{Proje Süresi } (T) = 2,5 \times (K_d)^{0.38} = 2,5 \times (45,36)^{0.38} = \mathbf{10,65 \text{ M (Ay)}}$$

$$\begin{aligned} \text{Ortalama Personel Sayısı} &= N = K_d / T = 45,36 \text{ MM} / 10,65 \text{ M} \\ &= \mathbf{4,26 \text{ M (Adam)} = 5 \text{ Adam}} \end{aligned}$$

# Kaynaklar

- David Gustafson, ‘Software Engineering’
- Ali Arifoğlu, Ali Doğru, ‘Yazılım Mühendisliği’
- Yüksel Bal, ‘Yazılım Projesi Geliştirme’ Ders Notları,
- Yüksel Bal, ‘Yazılım Mühendisliği ve Sistem Analizi’ Ders Notları,
- M. Erhan Sarıdoğan, ‘Yazılım Mühendisliği’,
- Oya Kalıpsız, Ayşe Buharalı, Ayşe Biricik, ‘Sistem Analizi ve Tasarımı’
- Çeşitli İnternet Kaynakları,