



## Ana Başlıklar

- ❖ Dil ve dilin özellikleri;
- ❖ Kod Çevrim İşlemi ;
- ❖ Etkinlik ;
- ❖ Atık toplama ;
- ❖ Aykırı Durumların kotarılması ;
- ❖ Temel İlkeler;
- ❖ Kod Yazımı;
- ❖ Belgelendirme;

**Bir programlama dili, programcının bir bilgisayara ne yapmasını istediğini anlatmasını anlatan bir yoludur.**

---

**Programlama dilleri, programcının bilgisayara hangi veri üzerinde işlem yapacağını, verinin nasıl depolanıp iletileceğini, hangi koşullarda hangi işlemlerin yapılacağını tam olarak anlatmasını sağlar.**

## Dilllerin Tarihçesi :

1960 -> Yapısal Programlama

---

1980 -> Nesneye Yönelik Prog. Ve Tasarım

C -> etkinlik, Progl. Gücü,kullanım kolaylığı ...

Java -> İnternet Uygulamaları ve taşınabilir progl.

## Dillerin Gelişimi :

1. Nesil Diller (makine düzeyinde Kodlama)
2. Nesil Diller (modern dillerin Temeli fortran ,cobol ...)
3. Nesil Diller (Modern va yapısal diller, C , pascal ...)
4. Nesil Diller (SQL, C++,XML,UML ...)
5. Nesil Diller (Konuya yönelik,ardışık düşünmeden programlama)

**Dillerin Özellikleri :** Programlama dillerinin karşılaştırılmasında çeşitli özelliklerin ve yeteneklerin varlığı esas alınır.

### **Genel Özellikler**

- Tasarımda koda geçiş Kolaylığı
- Amaca Uygunluk
- Dilin Etkinliği
- Derleyici Etkinliği
- Taşınabilirlik
- Geliştirme Araçları
- Bakım
- Tip Kontrolü
- Denetim Yapıları

## Nesneye Yönelik Dillerin Özellikleri

1. Modülerlik
2. Otomatik Bellek Yönetimi
3. Sınıflar
4. Kalıtım
5. Çokeşlilik

## Gerçek zamanlı Dillerin Özellikleri

---

1. Kuvvetli tip kontrolü
2. Dinamik bellek yönetimi
3. Parametre geçirme teknikleri
4. Hata yakalama ve kotarma
5. Soyut veri tipleri
6. Zaman belirtimi
7. Kolay okunabilirlik ,taşınabilirlik, genişleme yeteneği



## Dil Seçimi :

Projenin genel başarısı için büyük önem taşır.

---

- **Uygulama alanının özellikleri**
- **Donanım özellikleri**
- **Başarım gereksinimleri (Dilin diğer özellikleri)**
- **Veri yapılarının özellikleri**
- **Personelin bilgi düzeyi**
- **İyi bir derleyici ve geliştirme ortamı**
- **Bakım ortamı**



## Dillerin Uygulama Alanları :

### ○ Karma dillerin Kullanımı

---

#### ●Kullanıcı Arayüz Yazılımları

- C , C++, VC++, Java
- Delphi
- Pascal

#### ●Algoritma Yazılımları

- C
- Ada
- Fortran

#### ●Alt Sistem Arayüz Yazılımları

- C , C++
- Assembly
- Delphi

## Yeni Diller :

---

Arayışlar internet teknolojileri de kullanarak ASCII metin üzerinde çok kuvvetli işler yapabilen diller geliştirilmektedir.

## Yeni dillerin arayış kriterleri

Performans,  
Taşınabilirlik,  
Kolay Geliştirilebilirlik,  
...

## Kod Çevrim İşlemi :

---

Fiziksel Prosedüre ait adımların mantıksal eşdeğeri üretilmesidir.

Üretilen eşdeğer  
Derleyici  
Yorumlayıcı

Profesyonel bir çalışma ortamında(Editör) geliştirilmekte olan projeyi etkileyecektir.

**Kodlama Biçimleri :** Tasarım ne kadar iyi yapılırsa yapılsın onu hayata geçiren işlem kodlamadır.

---

- **Kodlama dili**
- **Kod belgeleme**
  - **İsimlendirme**
  - **Açıklamalar**
  - **Hata ayıklama**
- **Veri bildirimi (Veri yapılarının düzeni)**
- **Deyim Yapıları (kodlama sıra ve düzeni)**

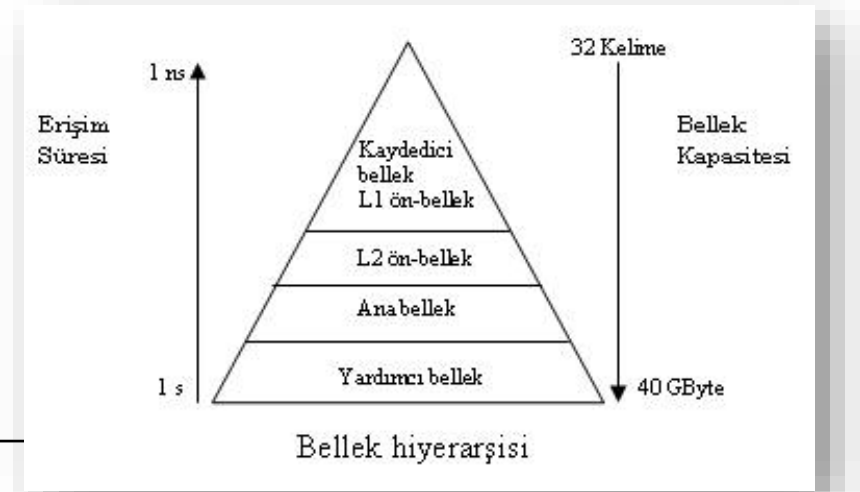
**Etkinlik** :Öz kaynakların etkin ve verimli bir şekilde kullanılarak amaçlara ulaşılmasıdır.

### Kod Etkinliği arttırma :

- Aritmetik işlem hazırlığı (anlaşılabilirlik , hız)
- Uzun yordamlar kullanılmamalıdır (Yeterince)
- Etkinliği düşürecek veri yapıları kullanımından kaçınılmalıdır.
- İşaretçi (pointer) kullanılmalıdır.
- Aritmetik işlemlerde makine mimarisi için en hızlı olan tercih edilmelidir.
- Uygunsuz atama ve mümkün olduğunca tipi uyumsuz veriler karşılaştırılmamalıdır.
- Veri tipi ve uzunluğu bellekteki durumu düşünülerek tasarlanmalıdır.



## Bellek Etkinliği artırma :



- Sık sık ekleme ve çıkarma yapılan , toplam eleman sayısı belirsiz olan veri yapılarında dinamik bellek kullanılmalı. Dilin özellikleri ön planda tutulmalı.
- Veri tipi tanımlanırken titizlikle uygun temel veri türü ve boyutu seçilmelidir.
- Kullanımı olmayan alanlar dahil edilmemelidir.
- Uygun değişkenler ve uzunluklar seçilmelidir.

## Giriş / Çıkış etkinliği :

1. Kullanıcı etkileşim
2. Çevre birimlerle etkileşim



## Kurallar :

- Giriş/çıkış istekleri mümkün olan en düşük düzeyde tutulmalıdır.
- İletişim yükünü azaltmak için tamponlanmalıdır.
- Veri türü ve büyüklüğü istek sıklığına göre en iyi hale getirilmelidir.
- Veri doğruluğu ve tutarlılığı kontrol edilmeli
- Dışarıdan gelen verilerin geçerliliği test edilmeli
- ...



## Atık toplama :

Kullanımı sona eren nesne ve veri yapılarının silinmesi.

*Atık toplama (garbage collection)*

---

Otomatik atık toplama için çeşitli algoritmalar toplanmıştır.

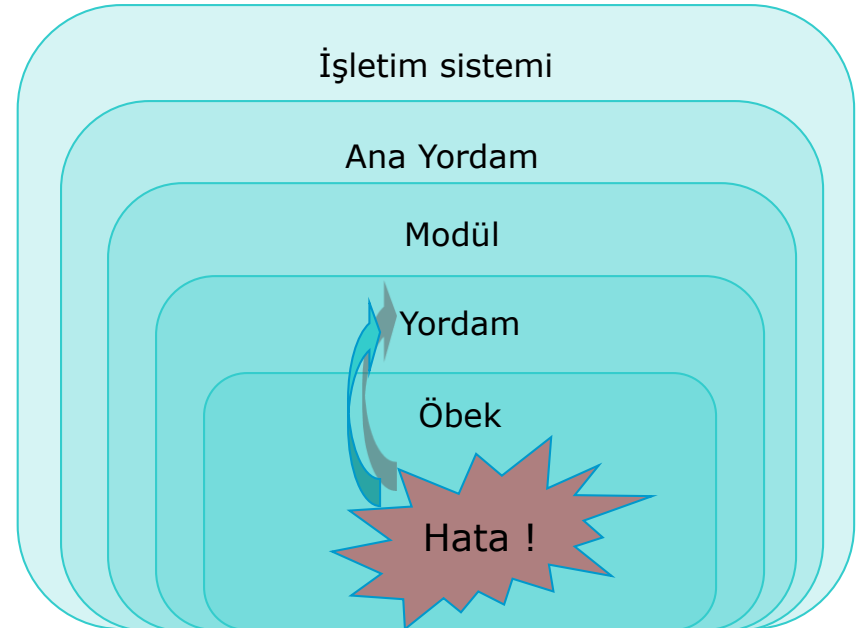
1. İşaretle ve temizle yöntemi
2. Referans sayma yöntemi
3. Yarılan kopyalama tekniği

## Aykırı Durumların kotarılması :

Programın denetim dışında sonlanmasına neden olan durumdur.

Yürütme sırasında herhangi bir olağan durum oluştuğunda sıraya göre şu işlemler yapılabilir.

1. Kod öbeği içinde kotarma
2. Yordam içinde kotarma
3. Modül içinde kotarma
4. Ana yordam içinde kotarma
5. Durdurma



## Temel İlkeler :

İyi bir yazılım geliştirmek; kişinin aklını kullanma yeteneğine , yeterli beğeni duygusuna ve sabıra sahip olmasını gerektirir.

### Kodlamada niteliksel özellikler :

---

1. **Soyutlama** (tekrardan kaçınılmalıdır)
2. **Bilgi Gizleme** (gerekli olan saklanmalıdır)
3. **Otomasyon** (Çalışm zamanı belirli olan ,önceden çalıştırılabilir)
4. **Çok Düzeyli Korunma** (seviyeli önlem)
5. **Etiketleme** (anlam kazandırma)
6. **Belirgin arayüz** (açık ve anlaşılır)
7. **Taşınabilirlik** (donanım ayrıntısı gözetilmemesi)
8. **Güvenlik** (veri ve erişim güvenliği sağlanmalı)
9. **Basitlik** (karmaşıklık uzak olunmalıdır)
10. **Genel Yapı** (metinsel biçim ve kurallara uyum...)
11. **Sözdizimsel Tutarlılık** (anlamsal benzerlikler , isimlendirme )
12. **Sıfır-bir-sonsuz** (kısıtlama yok, sabit sayıya göre tasarım yapılmamalı)

## Temel İlkeler

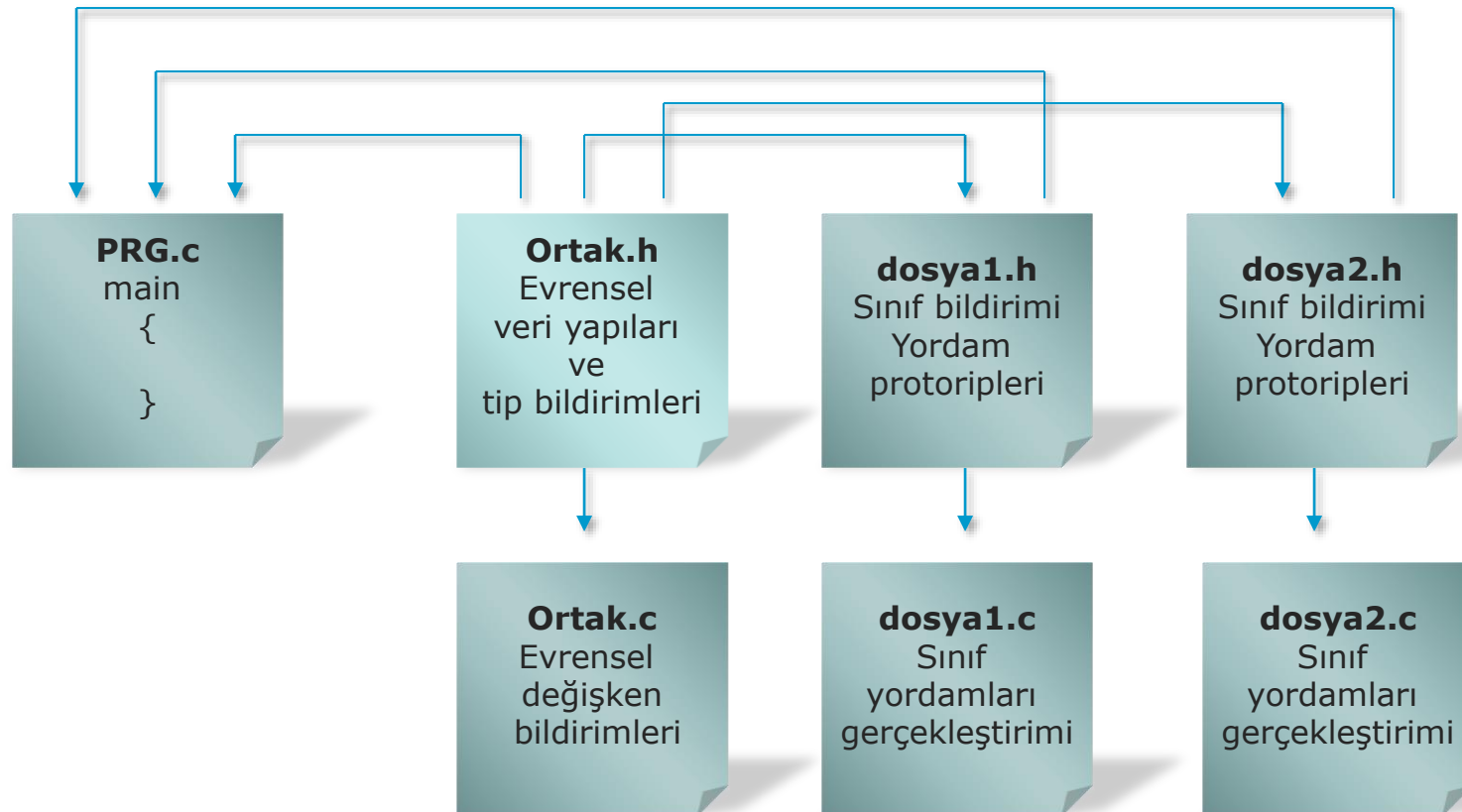
**Modül Oluşturma** :Birbirleriyle ilişkili yordamlarla bunlara ait verileri bir dosya içine koyarak bir modül oluşturulabilir.

---

### Dikkat edilmesi gereken özellikler:

- Aynı tip işlevlere sahip yordamlar belirlenmeli
- Modülün anlaşılabilir bir arayüzü olmalıdır
- Veri yapısı ve değişkenler modülün başında belirtilmelidir.
- Program için evrensel veri tipleri , makrolar oluşturulmalıdır.
- ...

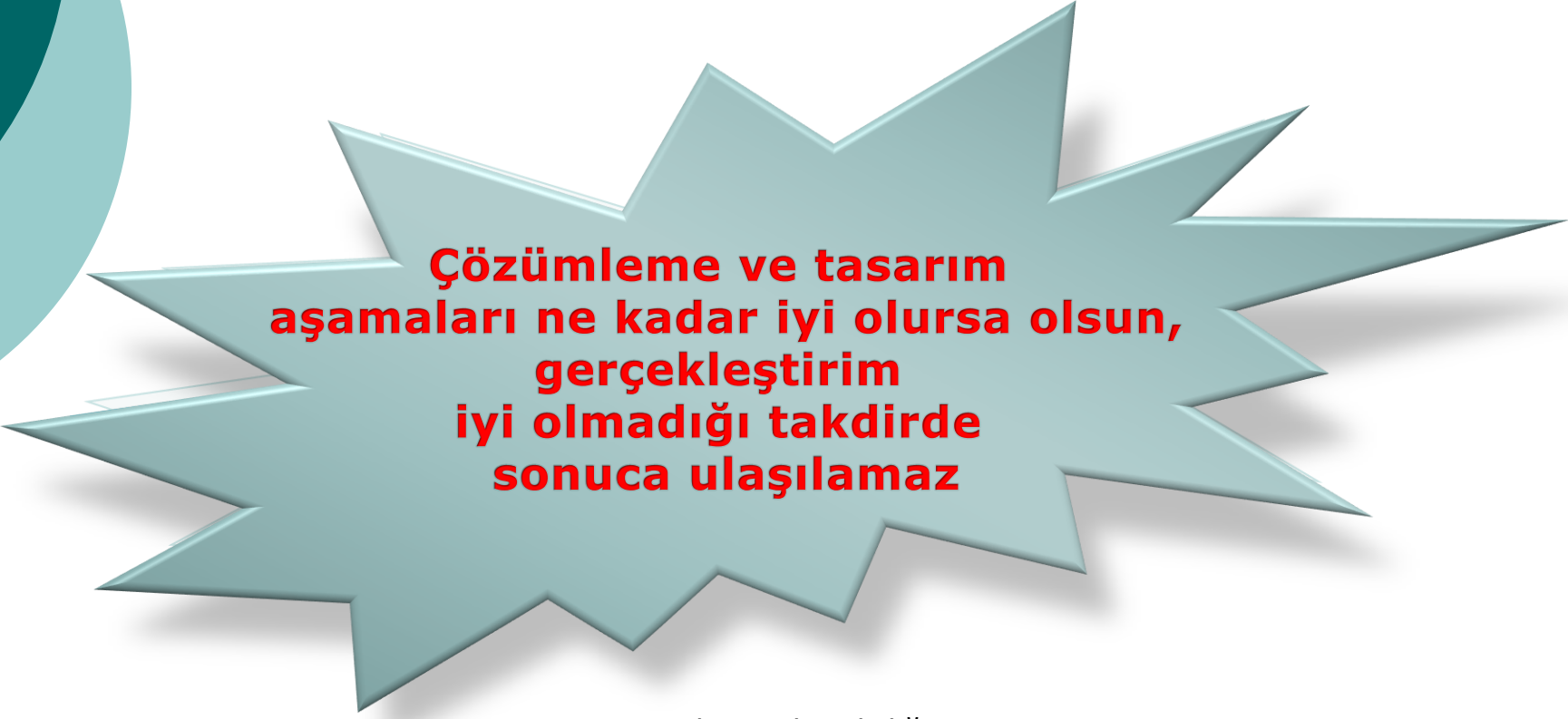
## Modüler Program Yapısı



## Kod Yazımı :

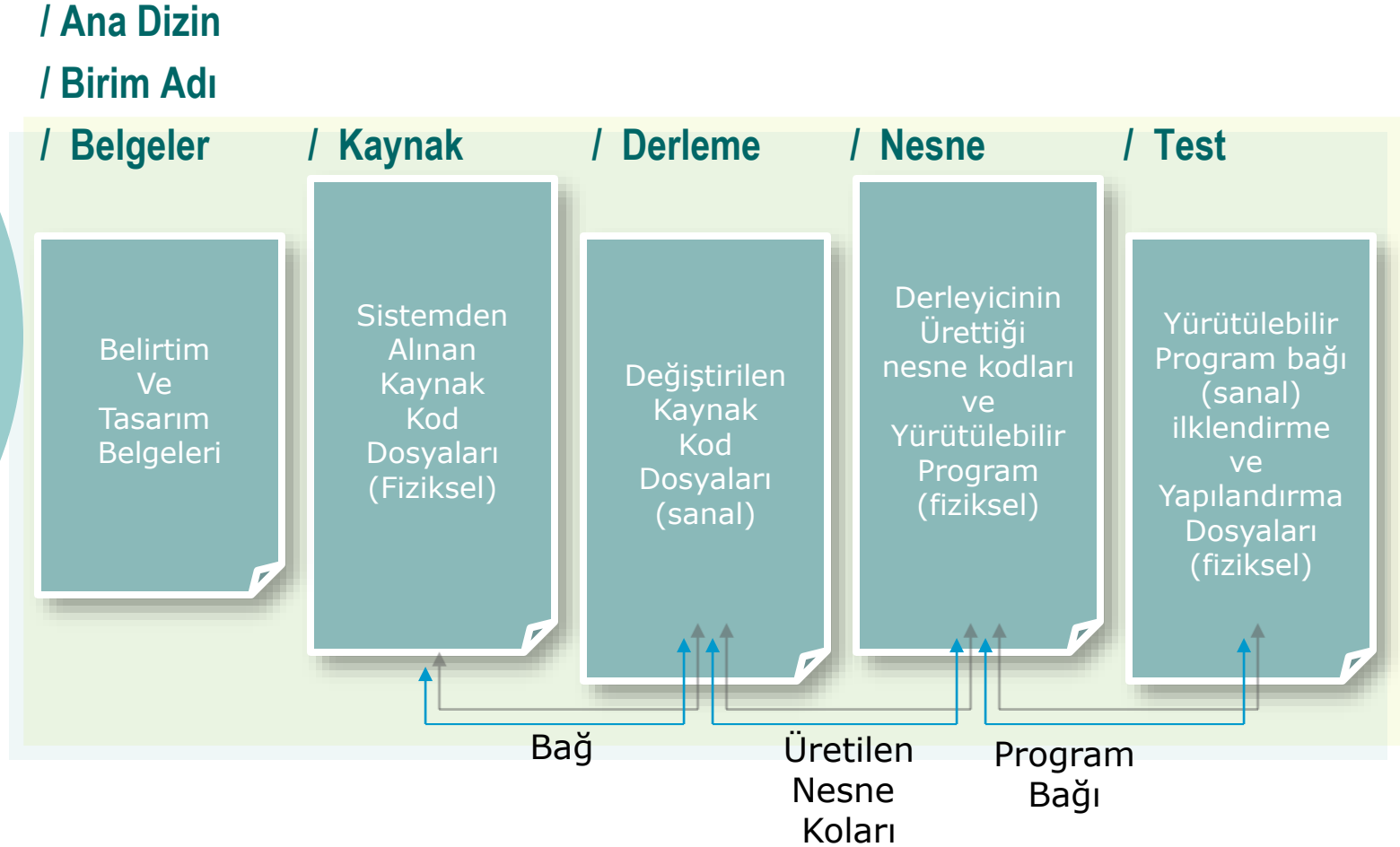
Kullanılan dilin özelliklerine göre kodlama yapılır.

---



**Çözümleme ve tasarım  
aşamaları ne kadar iyi olursa olsun,  
gerçekleştirim  
iyi olmadığı takdirde  
sonuca ulaşamaz**

## Çalışma Ortamı Dizin Yapısı



**Belgelendirme:** Kodlama sırasında elde edilen bazı bilgilerin yönetim sisteminde saklamak gerekebilir.

---

## Riskler :

- Dilin etkin kullanılmaması
- Geliştirme ortamının kısıtları
- Tasarımın tamamının koda dönüştürülememesi
- Yanlış kodlama
- Mantık hataları
- Eksik hata yakalama düzenekleri
- Düşük okunabilirlik
- Gereksiz kod parçaları
- Kod inceleme yapılmaması



