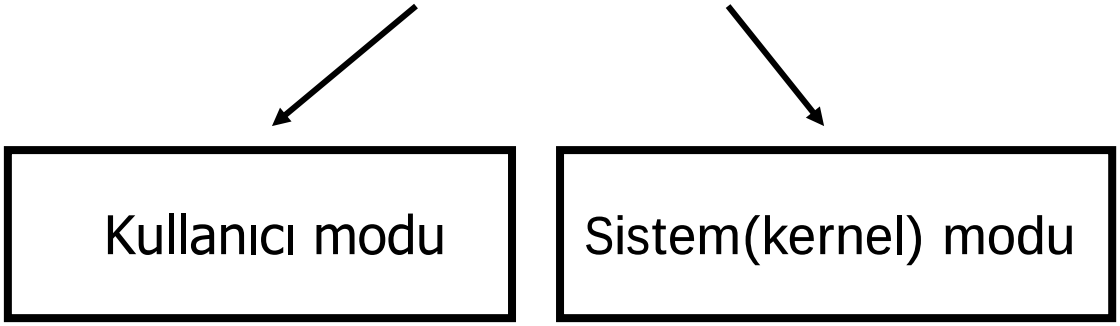
UNIX de iki moddaki operasyonlar(işlemler)

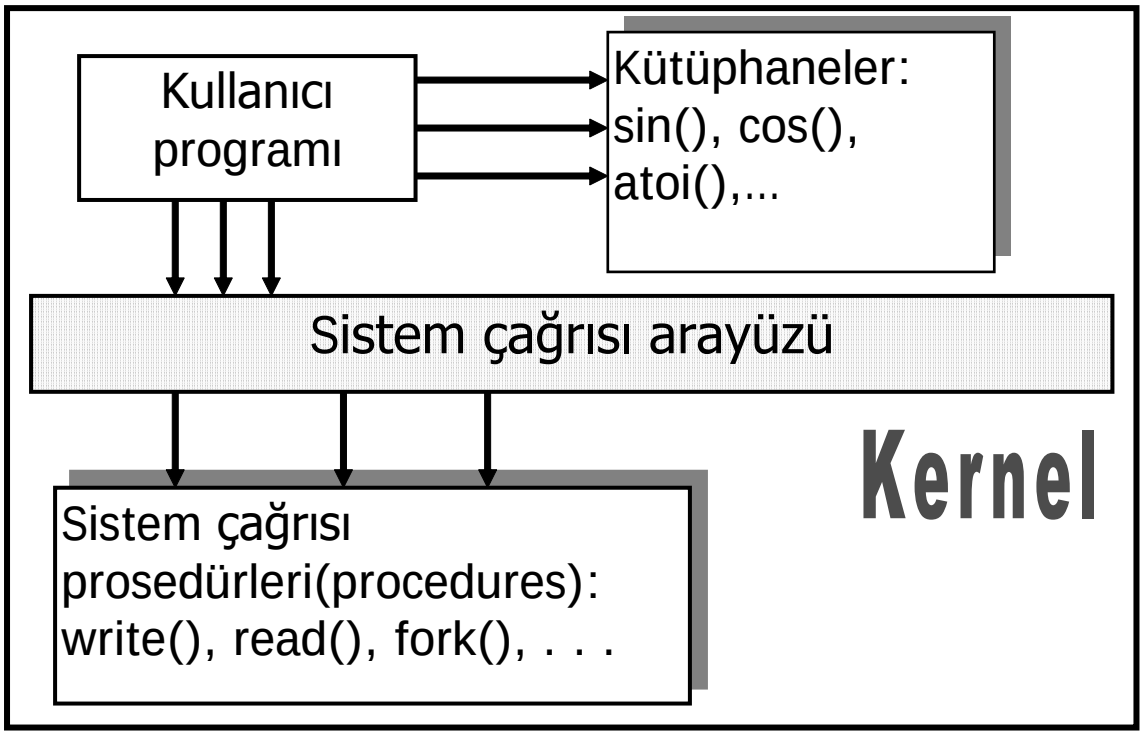


Bir gurup işlem (ayrıcalıklı işlem olarak da adlandırılır) yasaktır

Bütün işlemlere (ayrıcalıklı işlemler de dahil) izin verilir

- kesinti sağlama (interrupt enabling)
- kesinti kısıtlama (interrupt disabling)
- giriş/çıkış(I/O) cihazlarına doğrudan giriş
- v.b.

Kütüphane çağrıları ve sistem çağrıları

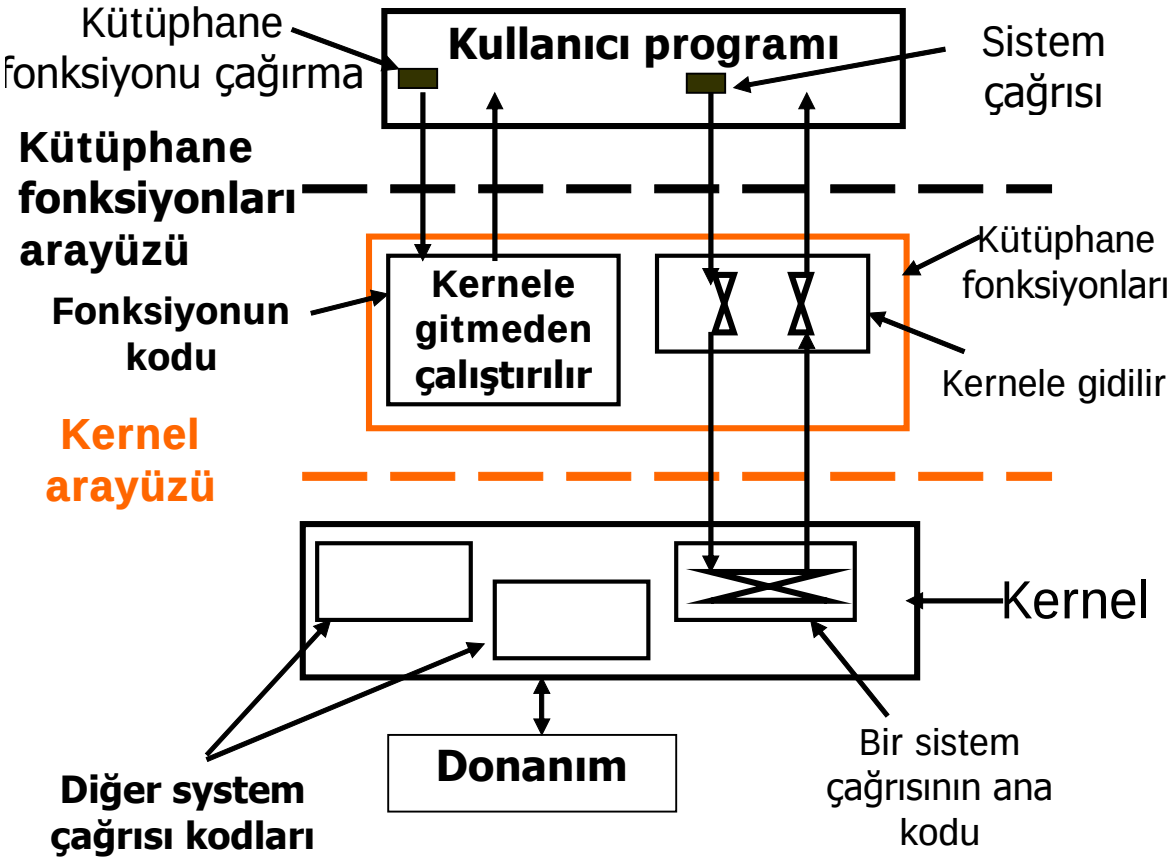


Notlar:

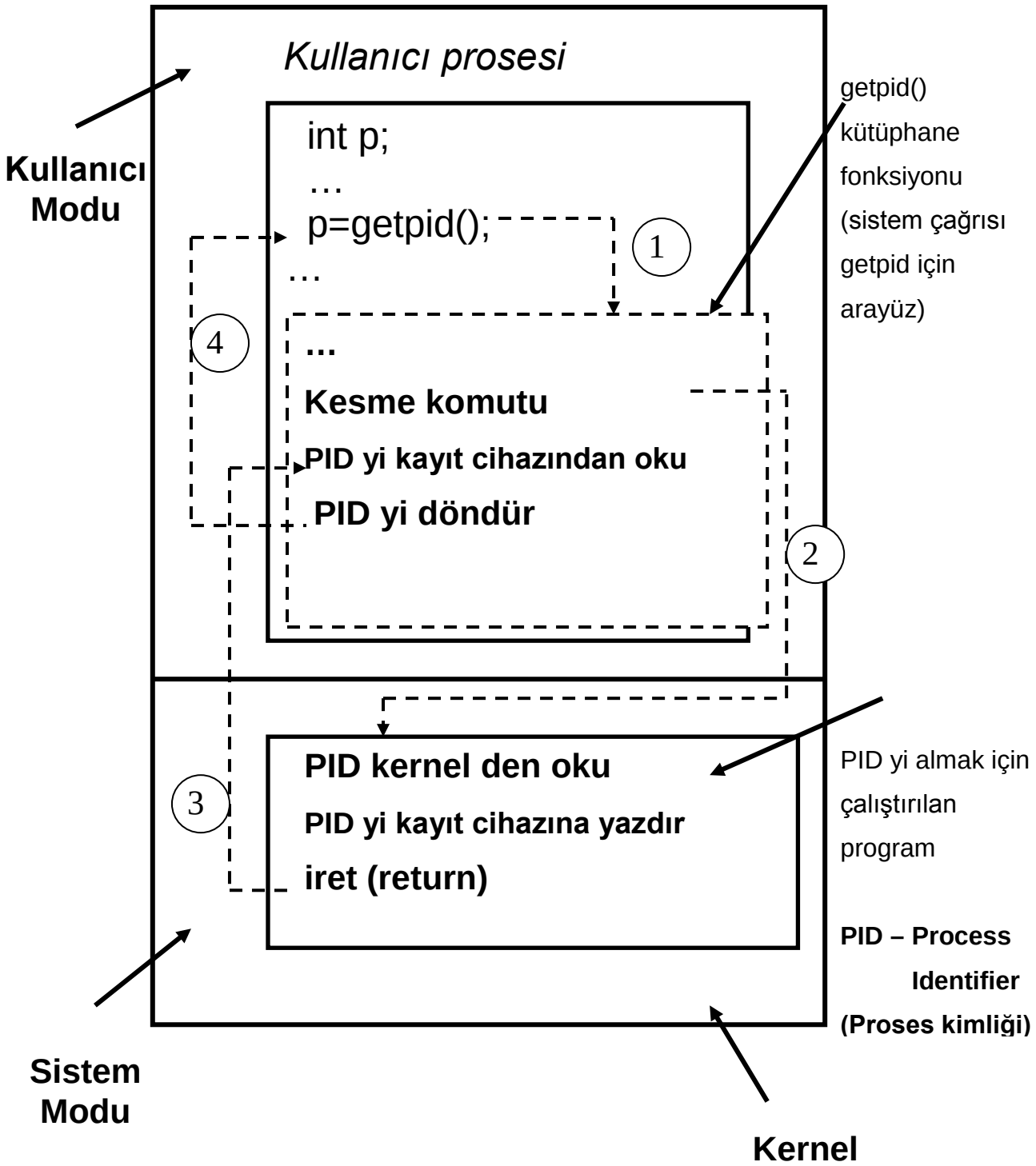
1. Kütüphane fonksiyonları çağrılırken sistem çağrısı arayüzüne gerek duyulmamaktadır

2.Bir system çağrısı her zaman sistem çağrısı arayüzünü ve işletim sisteminin çekirdeğini kullanmayı gerektirmektedir.

Bir kullanıcı programı, kütüphane fonksiyonları, sistem çağrıları



Bir sistem çağrısının Linux de gerçekleştirilmesi (getpid() system çağrısı için)



Bir sistem çağrısının genel biçimi

[dönen değer] = sistem_çağrısı_ismi (parametreler) ;
(eğer varsa)

dönen değer = { nenatif olmayan tamsayı , eğer tamamsa
-1, eğer hata oluşmuşsa
(bu durumda hata kodu *errno* dış değişkenine yerleştirilir)

Sistem Çağrılarındaki hataları elde etme

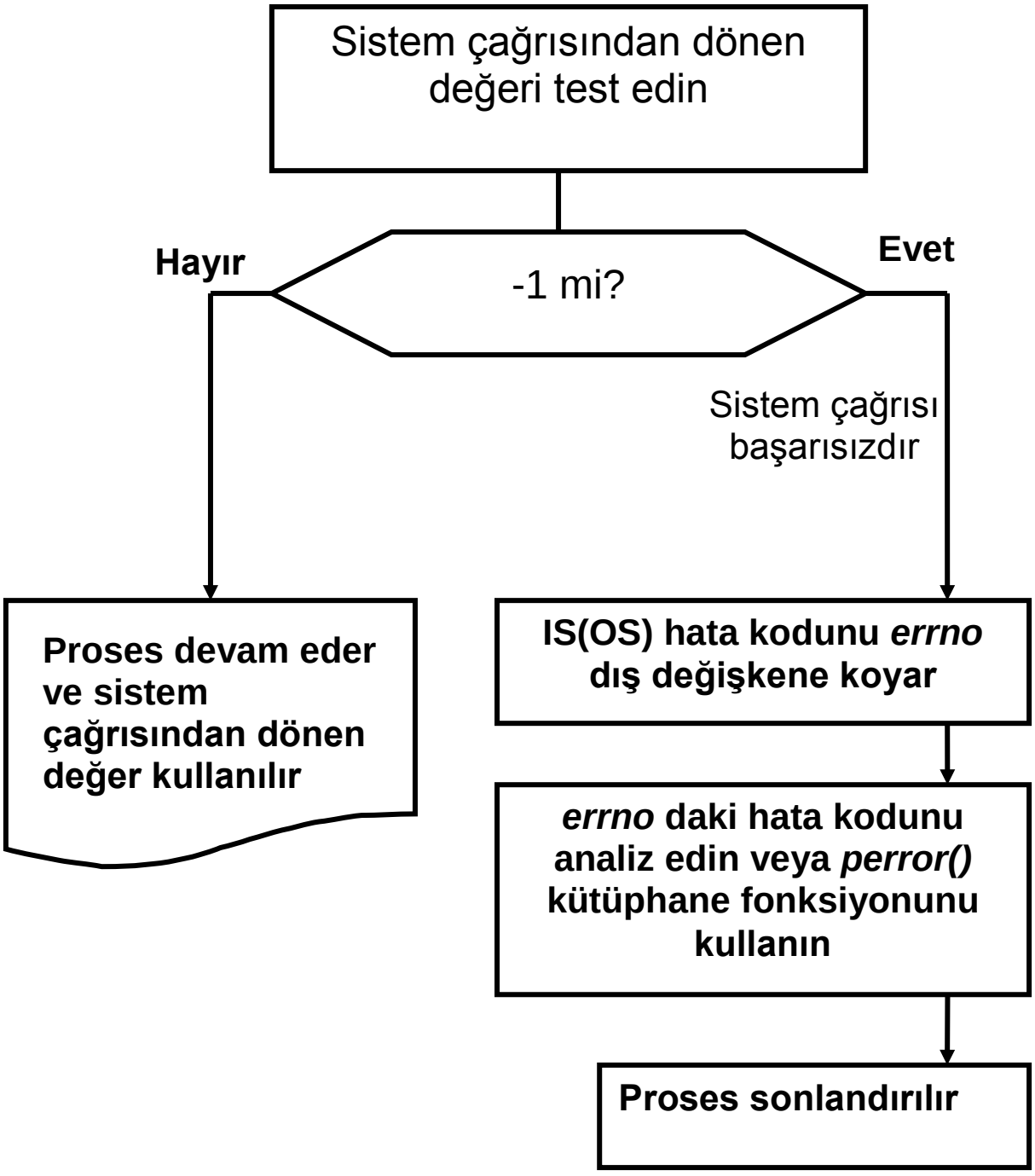
- Sistem çağrıları başarısız olduklarında genellikle -1, başarılı olduklarında sıfır veya sıfırdan büyük bir tam sayı döner.

- Programlama ipucusu:

  - (Hemen hemen) her zaman bir sistem çağrısının dönen değerini kontrol ediniz!!

  - Öte yandan *errno* yu eğer bir system çağrısı başarısız olursa kontrol ediniz...

Bir sistem çağrısı komutundan sonra program içindeki hareketler



Program içerisinde sistem çağrılarının çalıştırılmasında oluşabilecek hataları belirleme:

```
.....
.....
ret = sistem_çağrısı_ismi(...)
if(ret == -1)
   {  errno daki hata mesajını gör;
      hata mesajını yazdır;
     exit(1);
   }

/* Tamandır, devam edebilirsin*/
```

perror() (bir kütüphane fonksiyonudur) bir sistem çağrısının başarısız olma sebebini uygun bir hata mesajı yazdırarak açıklar.

```
#include<stdio.h>

void perror (const char *message);
```

Örnek 1: ***perror()*** **kütüphane fonksiyonunun kullanımı**

```
#include<fcntl.h>
#include...

int main(int argc, char *argv[])
{
   int fd;
   .....
   fd = open(argv[1], O_RDONLY); /* system çağrısı */
   if(fd = = -1)
   {
      perror(argv[1]); /* Hata mesajı yazdır ve çık */
      exit(1);
   }
.........
```

Programın kullanımı:

% progismi dosya_ismi

argv[1]

Örnek2:

```
#include<fcntl.h>
#include...

int main(int argc, char *argv[])
{
   int fd;
   .....
   fd = open("dosya_ismi", O_RDONLY);

   if(fd = = -1)
   {
      perror("dosya_ismi"); /* Hata mesajı yazdır ve çık */
      exit(1);
   }
.........
```

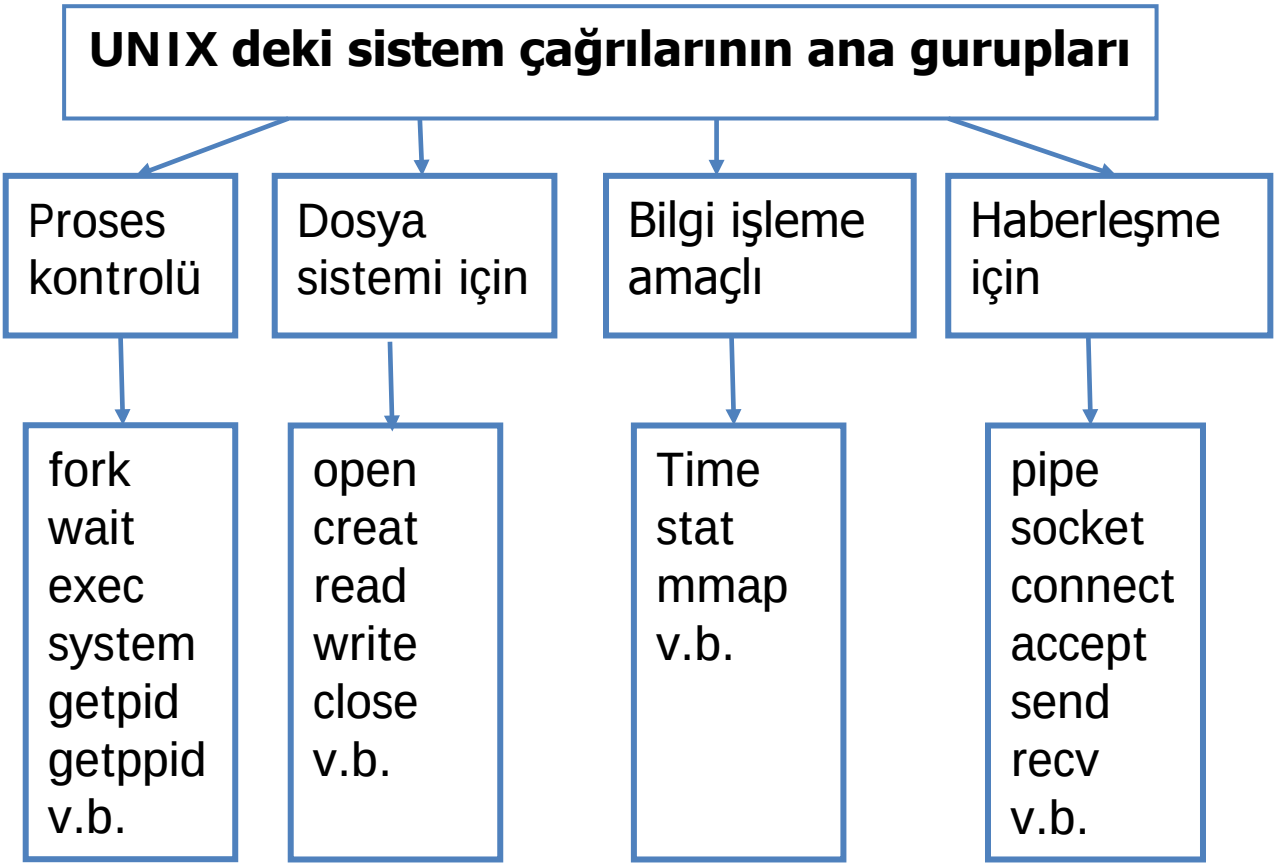
karakter dizgisi veya dizgi değişkeni

Eğer dosya açılmazsa:

Çıktı:

dosya_ismi: sistemin gönderdiği hata mesajı

örneğin – böyle bir dosya yoktur


UNIX deki sistem çağrılarının ana gurupları
Proses kontrolü
Dosya sistemi için
Bilgi işleme amaçlı
Haberleşme için
fork
wait
exec
system
getpid
getppid
v.b.
open
creat
read
write
close
v.b.
Time
stat
mmap
v.b.
pipe
socket
connect
accept
send
recv
v.b.