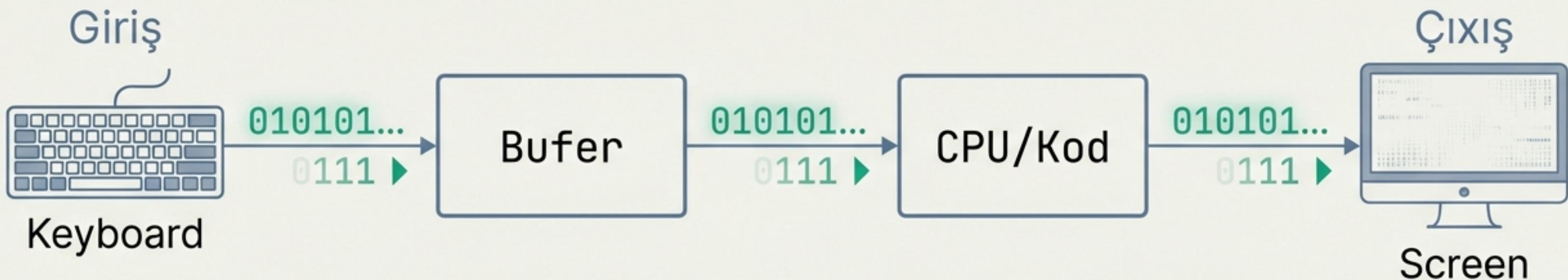


C DILINDƏ PROQRAMLASDIRMA

Mühazirə 3: Giriş və Çıxış Əməliyyatları

Məlumat axınının idarə edilməsi, təhlükəsizlik və standart funksiyalar



Standart Giriş-Çıxış Kitabxanası (stdio.h)



Çıxış (Danışmaq)

↗ `printf()`

Formatlı məlumat çıxışı (Universal)

↗ `puts()`

Sətir çapı (Avtomatik yeni sətir)

↗ `putchar()`

Tək simvol çapı



Giriş (Dinləmək)

↘ `scanf()`

Formatlı məlumat girişi

↘ `fgets()`

Təhlükəsiz sətir oxuma (Tövsiyə olunur)

↘ `getchar()`

Tək simvol oxuma

Məlumatın Ekrana Çıxarılması: printf və puts

```
int yas = 20;  
float gpa = 3.5;  
  
printf("Yaş: %d, Ortalaması: %.1f", yas, gpa);  
  
printf("Yaş: %d, Ortalaması: %.1f", yas, gpa);  
puts("\nTələbə məlumatı uğurla çap edildi.");
```

Legend

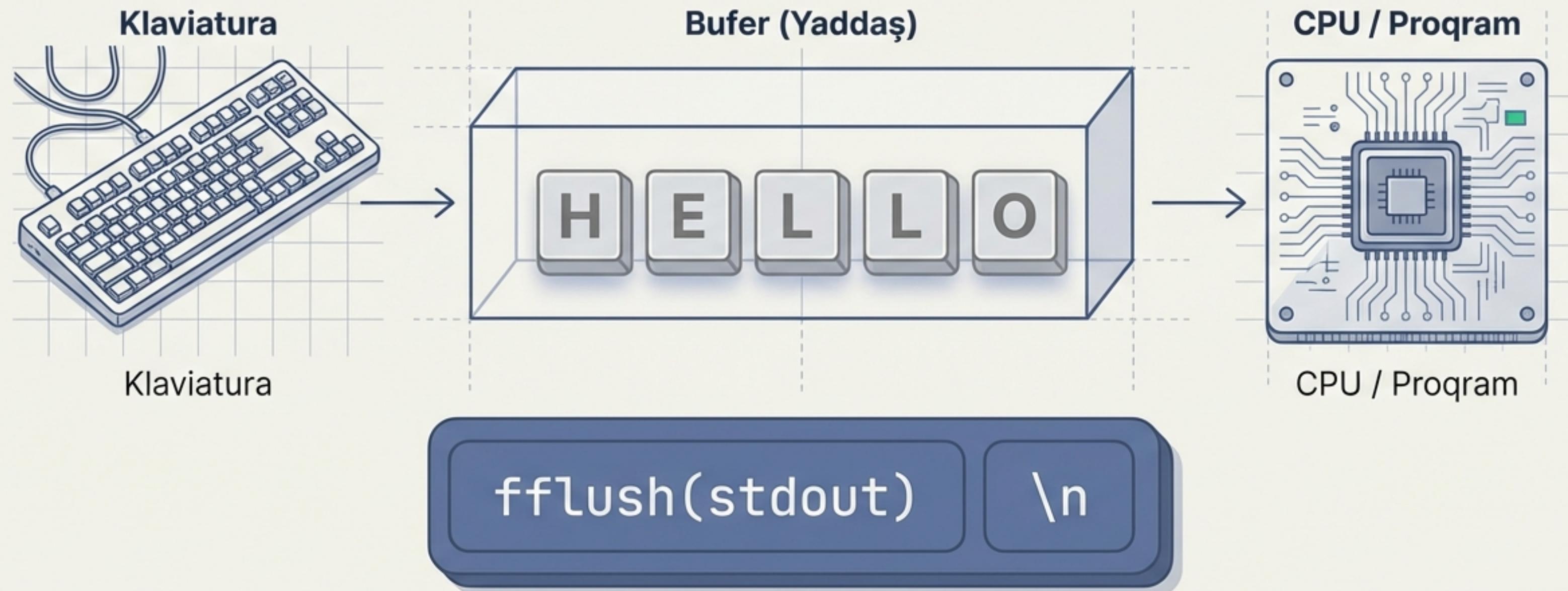
%d = Tam ədəd (integer) üçün yer tutucu.

%.1f = Kəsr ədəd (float) üçün 1 onluq dəqiqliklə yer tutucu.

Terminal Output

```
Yaş: 20, Ortalaması: 3.5  
Tələbə məlumatı uğurla çap edildi.
```

Gizli Mexanizm: Buferləşdirmə (Buffering)



- ✓ `setbuf()`: Buferi idarə etmək üçün.
- ! `fflush()`: Məlumatı dərhal emala ötürmək üçün.

Standart Giriş: scanf() və getchar()

```
int yas;  
scanf("%d", &yas);
```

QIZIL QAYDA:
Address Operator (&) mütləqdir!

```
double gpa;  
scanf("%lf", &gpa);
```

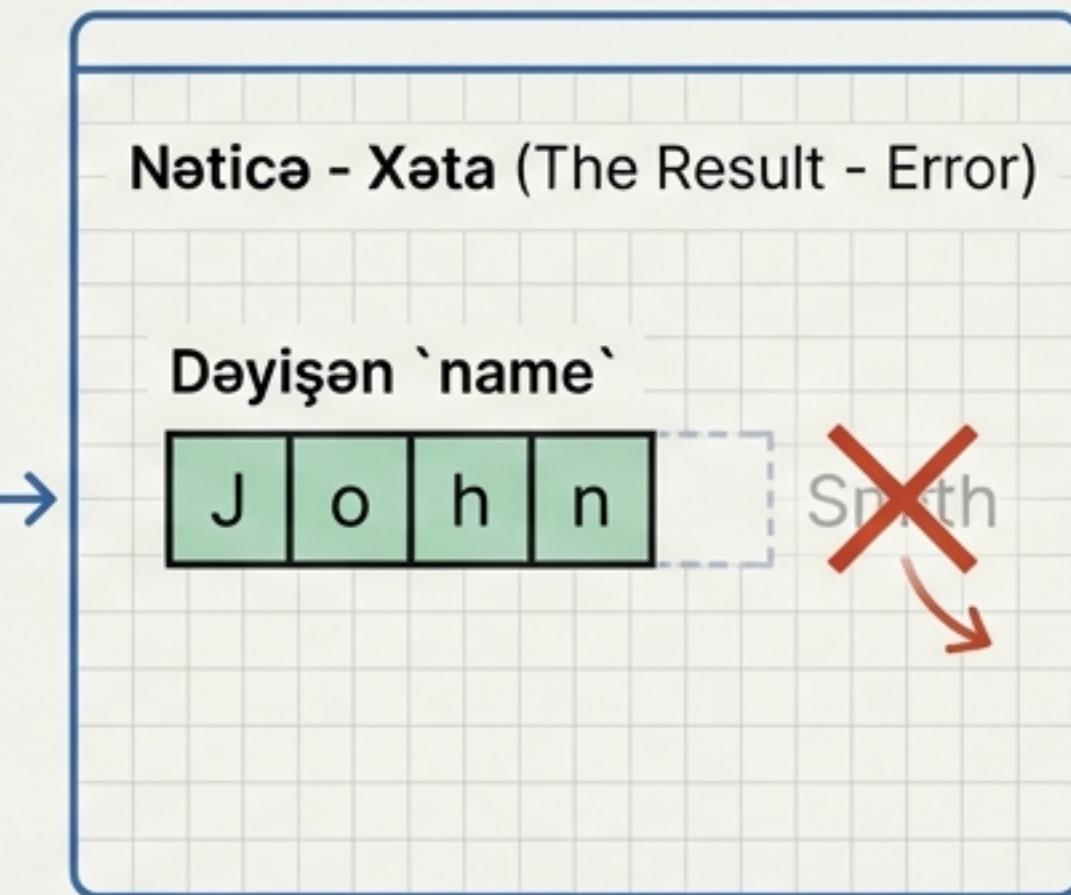
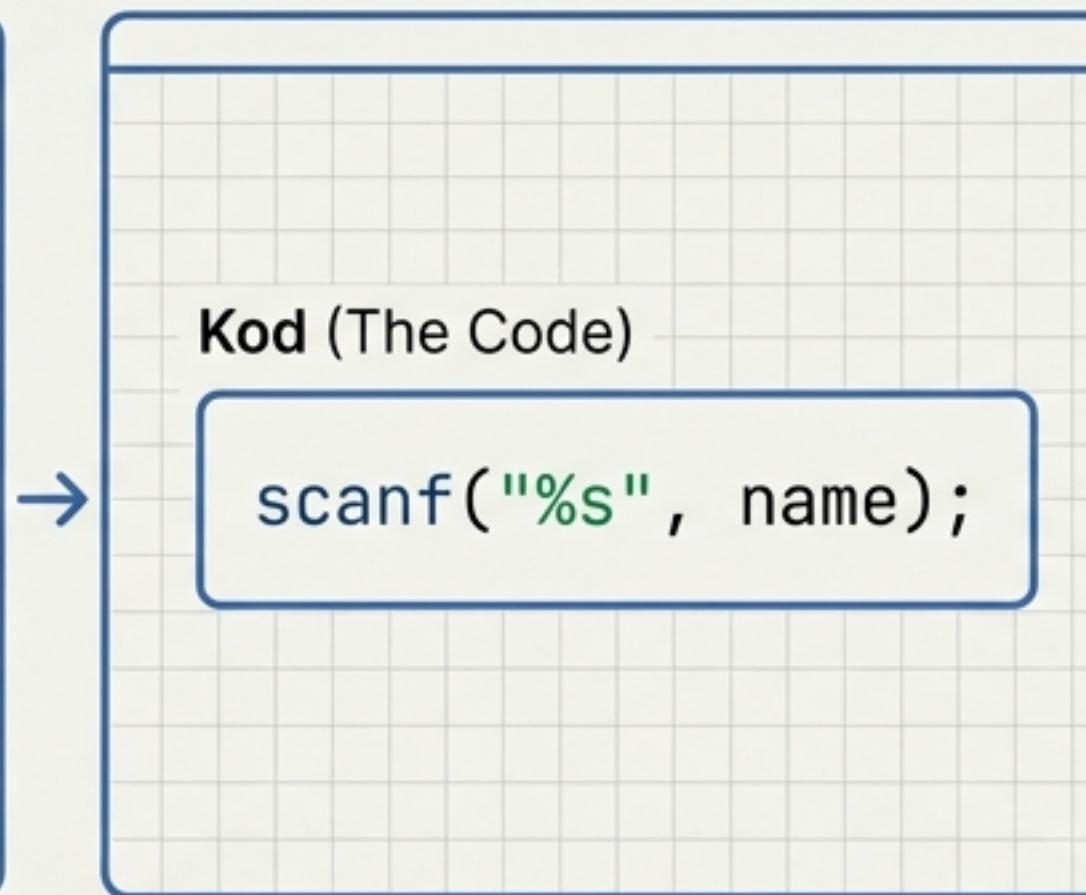
- Double üçün %lf istifadə olunur.

Sidebar Note:

getchar(): Yalnız bir düyməni oxuyur (Tez-tez ekranı saxlamaq üçün istifadə olunur).

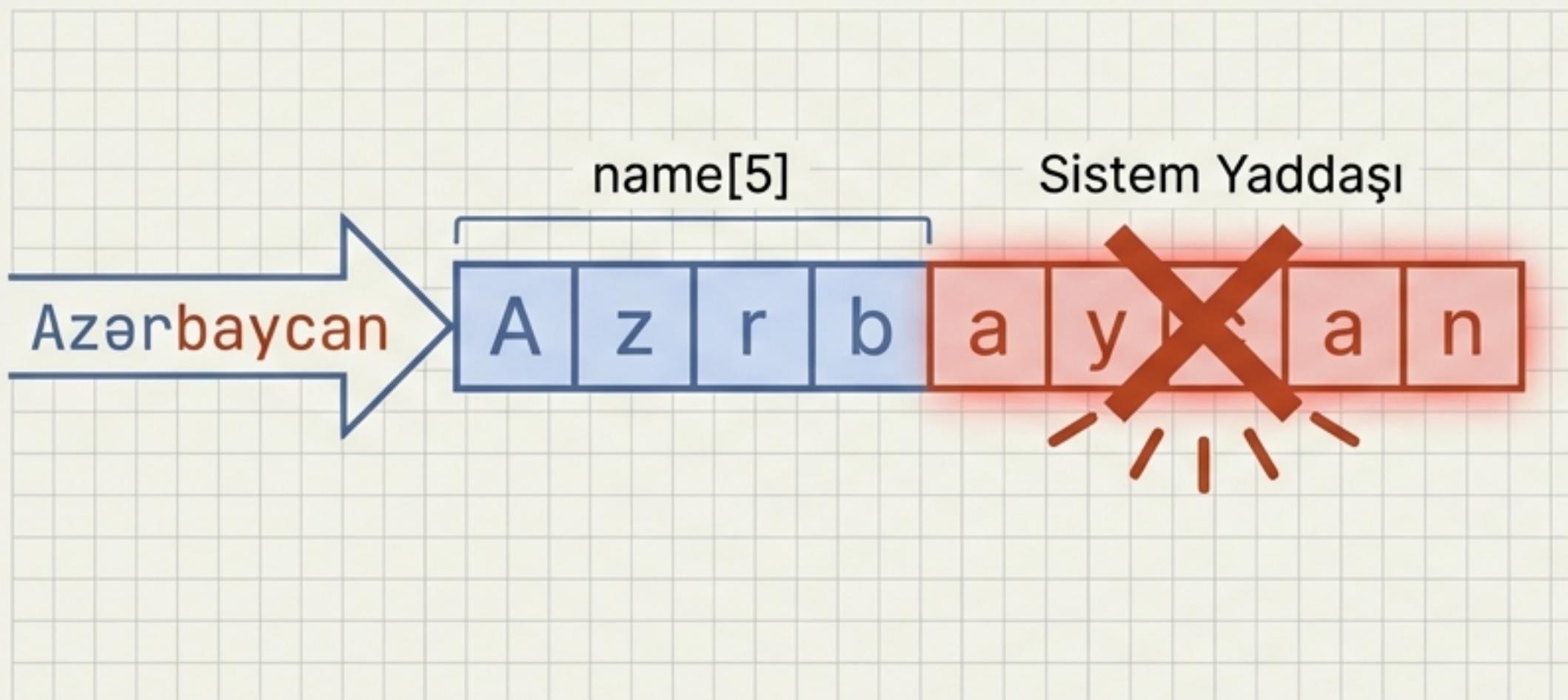


Problem 1: scanf və Boşluq Simvolu



- sccanf boşluq (space) gördükdə oxumağı dayandırır.
- ! Nəticə: Ad-soyadı bütöv oxumaq mümkün olmur.

Problem 2: Təhlükəsizlik və Giriş Daşmaları (Overflow)



- Yaddaşın **korlanması** (**Memory Corruption**)
- Programın **qəzası** (**Crash**)
- Hacker **hücumları** (**Buffer Overflow Exploits**)

scanf daxil edilən məlumatın ölçüsünü yoxlamır!

Həll Yolu: fgets() Funksiyası

Maksimal ölçü
(Təhlükəsizlik limiti)

Dəyişən adı ← **fgets(name, 20, stdin);** → Giriş mənbəyi
(Klaviatura)



Bosluqları oxuyur + Yaddasdan daşmanın qarşısını alır.

fgets() ilə işləməyin incəlikləri

Yeni sətir simvolunun (\n) təmizlənməsi

S	a		a	m	\n	\0
0	1	2	3	4	5	6



S	a		a	m	\0	✓
0	1	2	3	4	6	

```
size_t len = strlen(text);  
  
// Yeni sətir simvolunun \n  
if (text[len-1] == '\n') {  
    text[len-1] = '\0';  
}
```

Ədəd oxumaq üçün:
Öncə fgets ilə sətri oxuyun, sonra
sscanf və ya atoi ilə çevirin.

Praktika: Təhlükəsizlik Müqayisəsi

QADAĞAN (Unsafe)

```
char name[20];
scanf("%s", name);
```

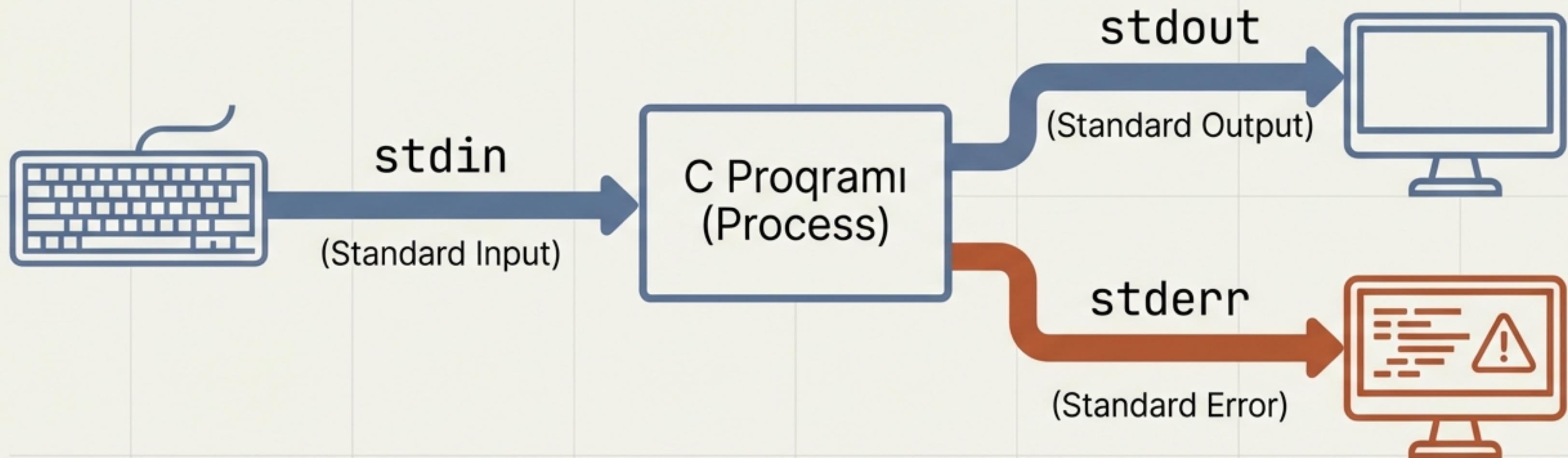
- Boşluqları oxumur
- Daşma (Overflow) riski var
- Sistem təhlükəsizliyini pozur

TÖVSIYƏ OLUNAN (Safe)

```
char name[20];
fgets(name, 20, stdin);
```

- Ad və Soyadı tam oxuyur
- Maksimum limiti qoruyur
- Yaddaş təhlükəsizliyi təmin edilir

Unix/Linux Mühiti: Standart Axınlar

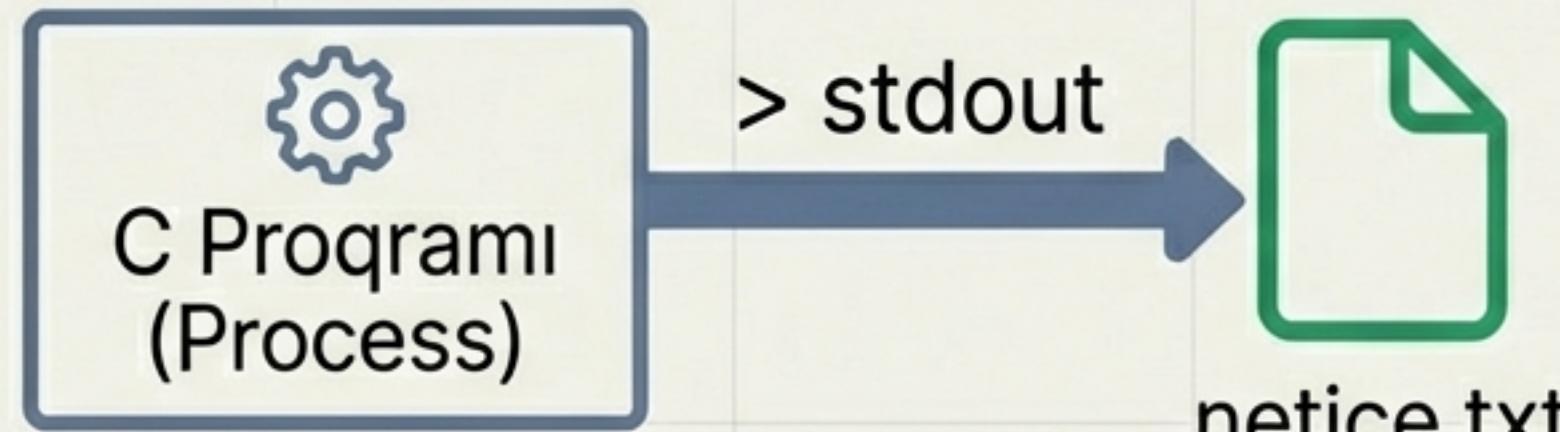


Unix sistemlərində program işə düşərkən
bu 3 kanal avtomatik açılır.

Faylların Yönlendirilməsi (Redirection)

Çıxış Yönlendirilməsi (Output)

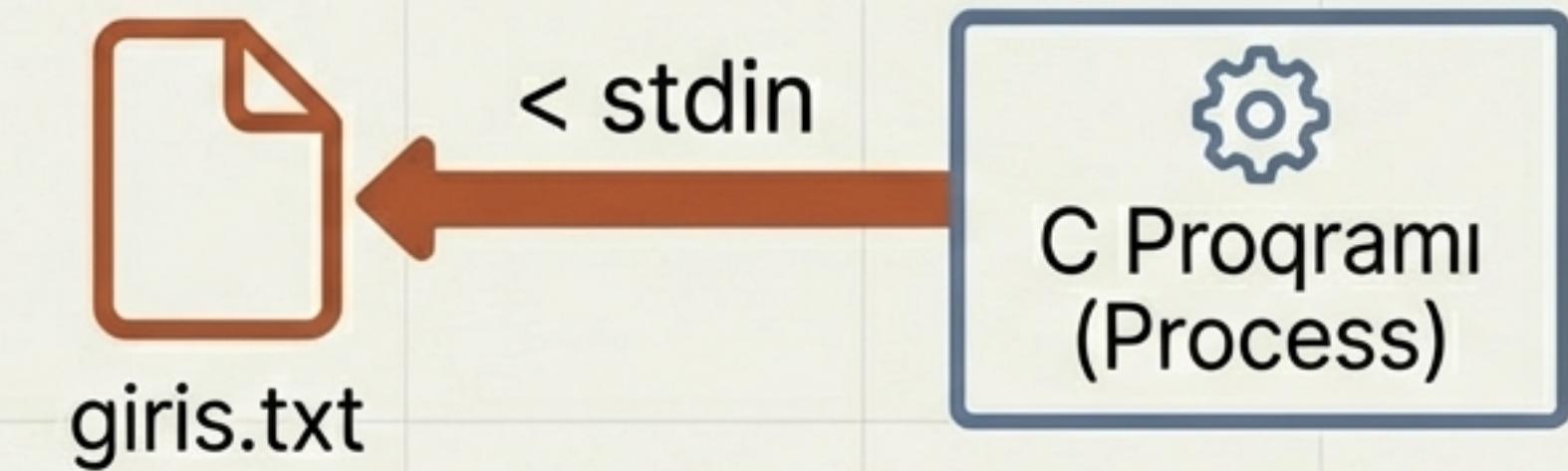
```
Mac OS X Terminal  
JetBrains Mono  
./app > netice.txt
```



Nəticə fayla yazılır.

Giriş Yönlendirilməsi (Input)

```
Mac OS X Terminal  
JetBrains Mono  
./app < giris.txt
```



Giriş fayldan oxunur.

Kod dəyişikliyi etmədən məlumat mənbəyini dəyişmək mümkündür.

Yönləndirmənin Praktiki Əhəmiyyəti Əhəmiyyəti

Avtomatlaşdırma



Test məlumatlarını əllə yazmağa ehtiyac yoxdur.

Loglama (Logging)



Nəticələrin faylda saxlanması və analizi.

Kombinasiya (Piping)



Bir programın çıkışını digərinə ötürmək (`|` operatoru).

Məlumatın Sətirə Yazılması: sprintf

```
printf("Koor: %d, %d", x, y);
```

Koor: 10, 20

```
sprintf(buffer, "Koor: %d, %d", x, y);
```

buffer[50]

```
char buffer[50];
int x = 10, y = 20;
sprintf(buffer, "Koor: %d, %d", x, y);
```

Koor: 10, 20\0

Ekrana yox, dəyişənə yazır. Mürəkkəb sətirləri formalasdırmaq üçün idealdır.

Xülasə: Məlumat Axınını İdarə Etmək

- ✓ Çıxış üçün `printf` və `puts` istifadə edin.
- ✓ Təhlükəsiz giriş üçün `scanf` əvəzinə `fgets` seçin.
- ✓ Giriş daşmalarından (Overflow) qorunmaq üçün həmişə ölçü limiti qoyun.
- ✓ Testləri sürətləndirmək üçün Yönləndirmədən (<, >) istifadə edin.

Təhlükəsiz kod yazmaq, işlək kod yazmaq qədər vacibdir.