

* Giriş Çıkış İşlemleri / Iostream:

* Program çalışırken dış dünyadan gelen Input / output : Giriş / Çıkış.

* Stream: bytes in flow

* Bu streami, ya okur / ya da salırız. Aynı byte stream'i getirmemiz. \rightarrow Bu byte stream'ı bufferlendirir.

* Fileter sistemini ve standard kütüphanenin içindeki de dahilinde \rightarrow input stream

\rightarrow output stream

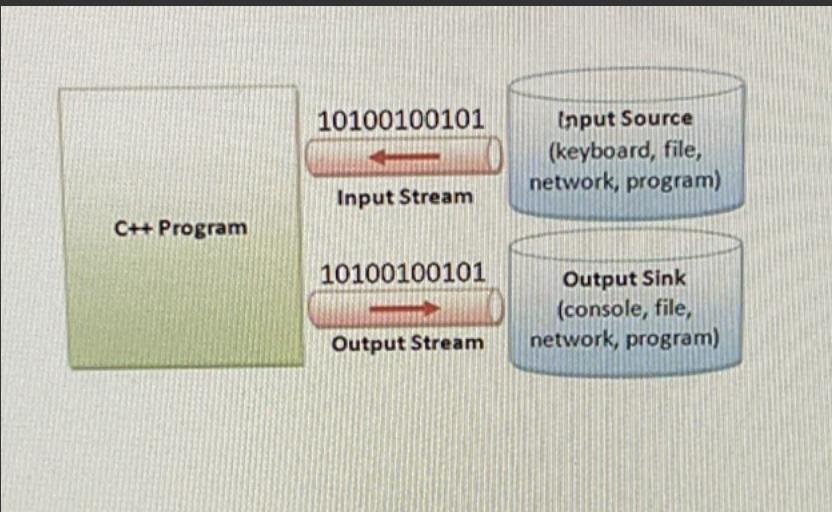
\rightarrow error stream

+ C'den farklı olarak, C++ da streamler bir object. \rightarrow Yani stream bir saytına ve bir interface'sı var.

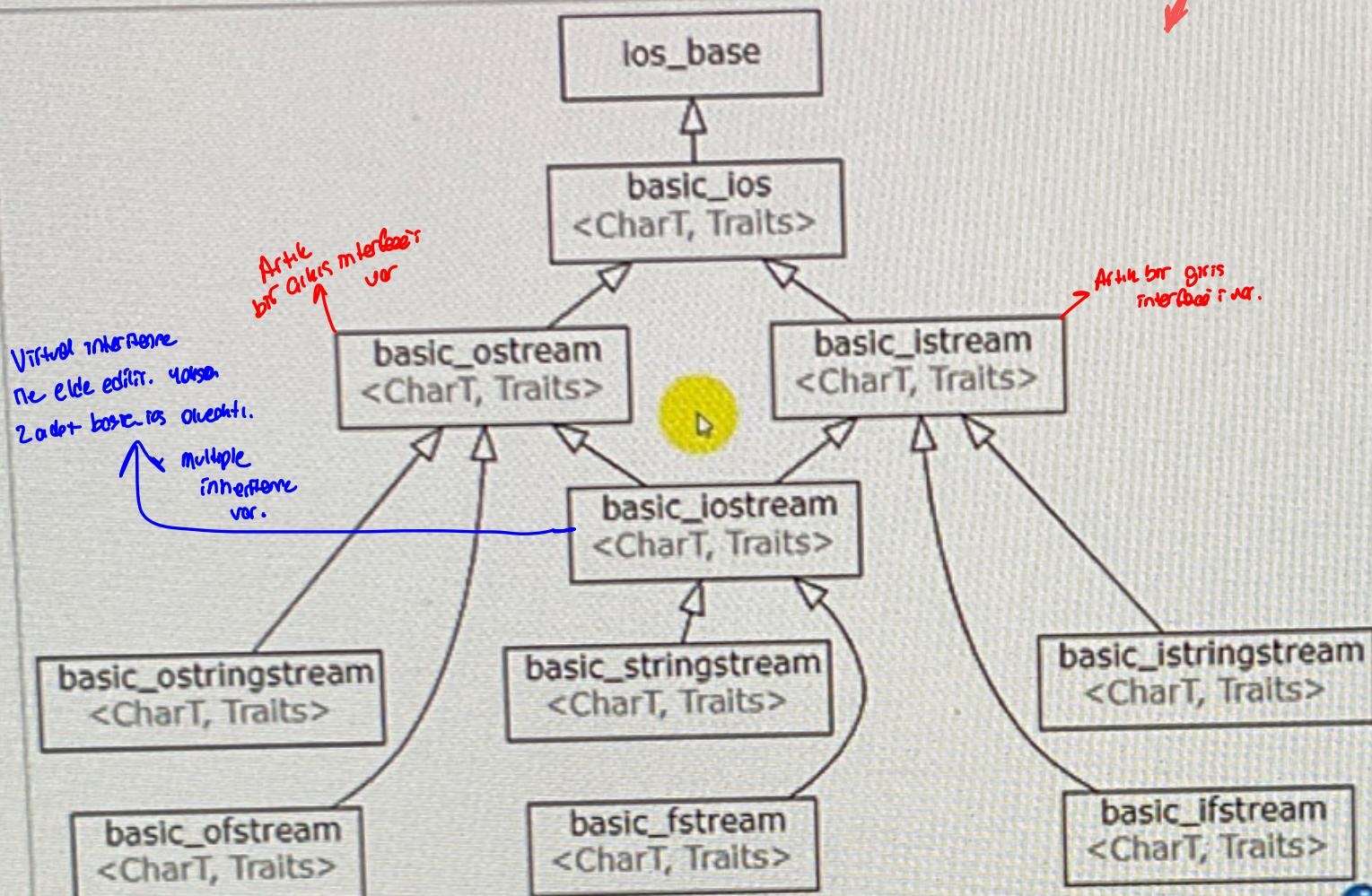
\rightarrow Object oriented olmanın yarısı, generic programming ve runtime polymorphism de var.

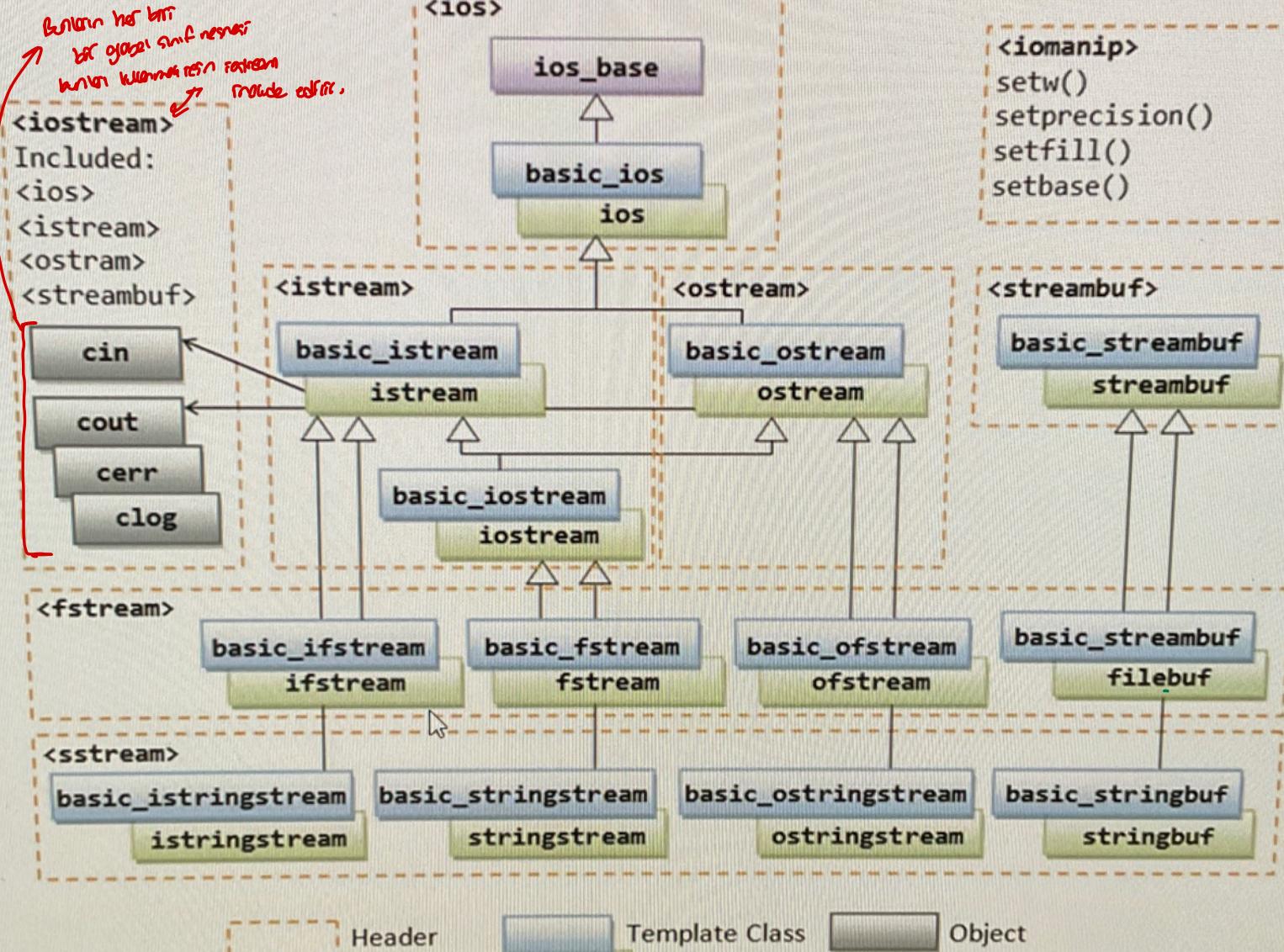
Ornekler cout / cin

bir nesne



Burada bertürmegen ve
Aşağı işaretler yapan: Stream'ün sınıfıdır. Filler nesnesi
bu buffer nesnesi yapsın.





* Bu sınıfların her biri bir özel sınıfının alt sınıfı.

```

int main()
{
    std::basic_ios<char>
    std::ios
}

```

Handwritten Notes:

- Annotation on std::basic_ios: "Bu sınıfların her biri bir özel sınıfının alt sınıfı" (These classes are each a subclass of a specific specialized class)
- Annotation on std::ios: "IOS UTEKLİĞİNİN HER BİRİ İSMİ." (Each of the IOSEKİLİĞİNİN HER BİRİ İSMİ.)

using namespace std;

```

auto n1 = ios_base::badbit; ✓
auto n2 = basic_ios<char>::badbit; ✓
auto n3 = ios::badbit; ✓
auto n4 = ostream::badbit; ✓

```

```
auto n5 = cout.badbit;
```

Fakat badbit, sınıfın static
veri elemanı, bu şekilde neme loçan
yapılır. Yoksası cout'un badbit'i gidi, int'de
etki etmeyecektir! Sınıfın var!

- Streamlerin **formatlama** (Formatting) ile formattača təqbiq etmək məqsədi. Bəzi birləşmələr.

```

int main()
{
    //cout << x;
    //mycout << x;
}

```

→ Földü: cout neme loçanın qədəhi formatları olubdır!!

format_state() → format state, state neme loçanın
encapsulate etmiş bir veri!!

→ Her stream neme loçanın bəzək format state təqbiq.

```
int main()
{
    using namespace std;

    ios_base x; // protected!
}
```

* Format operatorları:

→ Formatı gizli / gizli namespace gibi, formatı da bu türlerde kullanılır.

```
using namespace std;

cout.put('A');

friend std::ostream operator<<(std::ostream&, const Nec&);

};

inserter

cout << x
```

→ Ostream formatlı giriş fonksiyonları:

```
class ostream { // Normalde bu bir template

public:
    ostream& operator<<(int);
    ostream& operator<<(double);
    ostream& operator<<(long);
    ostream& operator<<(void *); adres yendirmek için
    ostream& operator<<(ostream& (*)(ostream &)); Finc pointer olur.
};

std::ostream& operator<<(std::ostream&, const char*); Cstring için
cinsinden değişken
```

```
using namespace std;

cout.operator<< ("sofia"); Bu "sofia" yordurur XX . Bu member function olan.
```

```
cout.operator<< ("sofia");
operator<<(cout, "sofia"); Yazının oblige adres
Yazının kendisini yazar.
```

```
using namespace std;
```

```
char c = 'A';
```

```
cout.operator<<(c);
```

int den dogru.

```
int main()
```

```
{
```

```
    using namespace std;
```

```
    int ival = 54807;
```

```
    cout.setf(ios::hex, ios::basefield);
```

→ Artık her cout kullanıldığında, hexabinary
olarak basıncaya oluyor!

→ Bir sonrakı unde e hatalı olduğu biliniyor.

```
int main()
```

```
{
```

```
    cout << true ? 10 : 20;
```

Syntax hatası yok. Çünkü ostream'in operator bool()
fonksiyonu var.

```
int main()
```

```
{
```

```
    cout << true ? 10 : 20;
```

```
    cout.operator<<(true).operator bool();
```

Dolayısıyla bu soruluk yorum!

```
int main()
```

artık tern boolean doğruları true false
gönderebilir.

```
{  
    cout.setf(ios::boolalpha);  
    cout << (20 > 10);  
}
```

Bu bildirim armsaşdı. true 0 yadırmalı.

```
cout.setf(ios::boolalpha);
```

```
cout << (20 > 10);
```

```
cout.unsetf(ios::boolalpha);
```

```
cout << (20 > 10);
```

```
///
```

- Agni flag'i kullanarak unset
edilebilir!

```
ostream os{ cout.rdbuf() };  
os << hex << showbase << uppercase << boolalpha;  
hex format, 0x eker  
hepsи büyük.
```

```
int x = 47804, y = 54807;
```

```
cout << x << " " << y << " " << true << "\n";  
os << x << " " << y << " " << true << "\n";
```

Microsoft Visual Studio Debug Console

47804 54807 1

0XBABC 0XD617 true

D:\CONCURRENCY\PACA_2022\Releases
Press any key to close this window

* İstediğiniz gibiostream'ı, istenilen formatla print edebiliyor.

* Format State:

→ büyük bir yapıda boolean olarak tutulur.

boolean değerler

true false

0 - 1

uppercase lowercase

showbase noshowbase

showpos → positive sayıları basma + eklenen mi eklememem mi?

showpoint → noktayı göster mi göstermem mi?

skip whitespace

unit buffering → buffer obucusu mi flush edilir yoksa, her aktış sırasında

zorla mi flush edilir?

* Flags Fonksiyonu

· fmtflags'i get eder. (Fmt+Flags, Stream State'i tutan bir tür)

```
using namespace std;
```

```
cout << typeid(ios::fmtflags).name() << "\n";
```

→ Neconun compileri, fmtflags türün int olduğunu.

* fmtflags: Değer int'e转化为 eden

bir bitmask. Yani her bir flag

aslında 2'nin bir katı olarak tutulur.

(Düzenlenen değiştirgeye degerlendir)

```
int main()
{
    using namespace std;

    cout.setf(ios::boolalpha);
    cout.flags(cout.flags() | ios::boolalpha);
```

set etme alternatif!

get eden fonksiyonu organın direkt bu
sentilde değiştire, set eder!

```
using namespace std;
```

```
cout.setf(ios::boolalpha);
cout.flags(cout.flags() | ios::boolalpha);
cout.flags(cout.flags() & ~ios::boolalpha);
```

unsetf alternatif olarak kullanılabilir.

* Cool 1 yöntem:

```
int main()
{
    using namespace std;

    cout << (cout.flags() & ios::showbase ? "gosterir" : "gostermey");
```

Bir önceki örneğin daha da iyi hali:

```
void display_on_of_flags(std::ios_base& ib)
{
    if (ib.flags() & std::ios::boolalpha)
```

Flags'ı bu sınıftan elde edildiğinde, bu sınıfın türünden tüm
Flags'ı göstermeye imkan sunar. Fonksiyon:

* Setf Overload:

→ setf ile hex, dec, oct gibi tabanı markeleri tek bir ile gösterilmesi. Çünkü bunlar aynı anda set edilmemeli.
Bu yüzden setf'nin şöyle bir overload'u var.

010
100
001
8bit,

```
int main()
{
    using namespace std;

    cout.setf(ios::hex, ios::basefield);
```

→ Default onlu sayı sistemi

```
int main()
{
    using namespace std;

    cout << 0xBABA << "\n"; //dec
    cout.setf(ios::hex, ios::basefield);
    cout << 0xBABA << "\n"; //hex
    cout.setf(ios::oct, ios::basefield);
    cout << 0xBABA << "\n"; //oct
    cout.setf(ios::dec, ios::basefield);
    cout << 0xBABA << "\n"; //dec
```

```
int main()
{
    using namespace std;

    cout << "dec :" << (cout.flags() & ios::dec ? "set" : "unset") << '\n';
    cout << "oct :" << (cout.flags() & ios::oct ? "set" : "unset") << '\n';
    cout << "hex :" << (cout.flags() & ios::hex ? "set" : "unset") << '\n';
}
```

→ Bu ne flags'ı ne sayıları
yapıyor??.

* Output width: → Output streaminde, yani sağa/sola dayalı olarak icon

output width

```
int ival = 456
-----456
+      7234
-      786343
ios::adjustfield
ios::left
ios::right
ios::internal
```

```
int main()
{
    using namespace std;

    cout.setf(ios::left, ios::adjustfield);
    cout.width(20);
    cout << 127 << "eray";
}
```

width yarınca ilerken ayarlanır. Sonra geri 0

Microsoft Visual Studio Debug Console
127 eray
D:\CONCURRENCY\PACA_2022\Release\PACA...
Press any key to close this window...

* Fixed / Scientific Output:

The screenshot shows a Microsoft Visual Studio code editor and a debug console window. The code in the editor is as follows:

```
#include <iostream>
#include <bitset>

int main()
{
    using namespace std;

    cout << 76234.87435 << "\n"; → Fixed
    cout << 8973459872435.7829345 << "\n"; → Scientific
}
```

The debug console window shows the output:

```
Microsoft Visual Studio Debug Console
76234.9
8.97346e+12
D:\CONCURRENCY\PACA_2022\Release\PACA_2022.exe (process...
Press any key to close this window...
```

* Get / Set Funktionen:

The screenshot shows a Microsoft Visual Studio code editor with a tooltip for the `ios::fixed` member function.

```
int main()
{
    using namespace std;

    cout << 76234.87435 << "\n";
    cout << 8973459872435.7829345 << "\n";
    cout.setf(ios::fixed, ios::floatfield);
    cout << 8973459872435.7829345 << "\n";
    cout.setf(ios::scientific, ios::floatfield);
    cout << 76234.87435 << "\n";
    cout.setf(ios::fixed);
    cout << 76234.87435 << "\n";
}
```

The tooltip content is:

- static constexpr inline std::ios::floatfield
- scientific | fixed
- Search Online

* Get / Set Funktionen:

The screenshot shows a Microsoft Visual Studio code editor with a yellow circle highlighting the `cout.precision(6);` line.

```
cout.width()
cout.width(12)

cout.fill()
cout.fill('-')

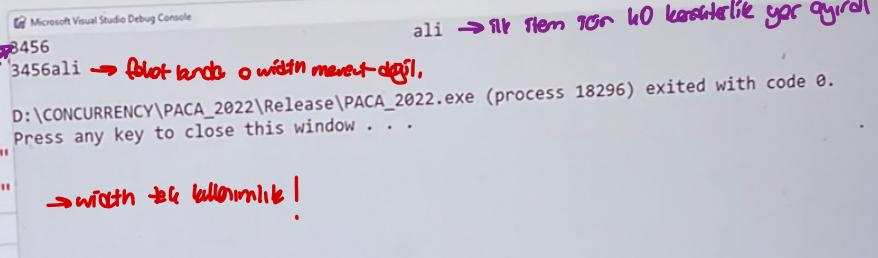
cout.precision()
cout.precision(6);
```

* parameter `char` → set function
der return → get function

* width: Format sıteci's kolcu amagın tek özellih. Her item için tekrar tekrar set edilmeli!

```
{ using namespace std;
```

```
cout.setf(ios::left, ios::zfill);
[ cout.width(40);
  cout << 3456 << "ali" << " "
  cout << 3456 << "ali" << "
```



```
int main()
```

```
{ using namespace std;
  cout.width(3);
  cout << 987123761;
```

→ width truncle etmem ✓